TUGAS AKHIR – KI141502

RANCANG BANGUN KOMPUTASI PERVASIF PADA SISTEM MONITORING DAN FORECASTING CHARGING STATION MOBIL LISTRIK DENGAN ARDUINO BERBASIS ANDROID

SANINDYA LESARIO
NRP 5111100018

Dosen Pembimbing
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
Dr. Dimas Anton Asfani, ST., MT.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

[Halaman ini sengaja dikosongkan]

UNDERGRADUATE THESES – KI141502

PERVASIVE COMPUTING IN ELECTRIC CAR CHARGING STATION MONITORING SYSTEM AND FORECASTING WITH ARDUINO AND ANDROID BASED

SANINDYA LESARIO
NRP 5112100141

Supervisor
Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.
Dr. Dimas Anton Asfani, ST., MT.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
SURABAYA2016

[Halaman ini sengaja dikosongkan]

LEMBAR PENGESAHAN

RANCANG BANGUN KOMPUTASI PERVASIF PADA SISTEM MONITORING DAN FORECASTING CHARGING STATION MOBIL LISTRIK DENGAN ARDUINO BERBASIS ANDROID

TUGAS AKHIR

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Bidang Studi Komputasi Berbasis Jaringan
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh
SANINDYA LESARIO
NRP : 5112 100 141

Disetujui oleh Dosen Pembimbing Tugas Akhir:

1. Dr. Eng. Radityo Anggoro, S.Kom., M.Sc. .....
   NIP:198410162008121002 (Pembimbing 1)
2. Dr. Dimas Anton Asfani, ST., MT. .....
   NIP:198109052005011002 (Pembimbing 2)

SURABAYA
JUNI, 2016

[Halaman ini sengaja dikosongkan]

RANCANG BANGUN KOMPUTASI PERVASIF PADA SISTEM MONITORING dan FORECASTING CHARGING STATION MOBIL LISTRIK DENGAN ARDUINO BERBASIS ANDROID

Nama Mahasiswa	**: SANINDYA LESARIO**
NRP	**: 5112100141**
Jurusan	**: Teknik Informatika FTIF-ITS**
Dosen Pembimbing 1	**: Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.**
Dosen Pembimbing 2	**: Dr. Dimas Anton Asfani, ST., MT.**

Abstrak

Keberadaan mobil listrik di Indonesia telah menunjukkan suatu kemajuan dalam bidang teknologi khususnya dalam inovasi bahan bakar baru yaitu listrik. Untuk dapat terus digunakan dalam berbagai keperluan, mobil listrik membutuhkan stasiun isi ulang atau charging station sebagai penyuplai energi listrik. Untuk pengisian mobil listrik, dibutuhkan pengukuran daya listrik yang keluar dari charging station dengan tepat dan akurat. Hal ini merupakan langkah penting untuk menghindari terjadinya kelebihan beban arus yang diterima mobil listrik. Beban arus yang berlebih dapat memicu terjadinya hubungan antar penghantar bertegangan secara langsung yang tidak melalui media yang semestinya sehingga menyebabkan terjadinya aliran arus yang tidak normal (korsleting).

Oleh karena itu dalam tugas akhir ini, saya membuat sebuah sistem yang dapat melakukan monitor kondisi daya listrik yang mengalir dari charging station ke mobil listrik yang ingin melakukan isi ulang. Aliran perubahan arus listrik yang keluar dideteksi oleh sensor arus, kemudian diproses menggunakan mikrokontroler Arduino untuk dilakukan pemrosesan data arus listrik yang akan ditampilkan di perangkat bergerak (mobile). Untuk dapat menghitung pemakaian daya yang akan datang

(menghitung pemakaian daya bulan depan), digunakan algoritma forecasting / peramalan. Data yang akan ditampilkan pada sistem monitoring meliputi daya listrik yang keluar dari charging station, laporan peramalan daya.

Aplikasi ini diharapkan dapat menyelesaikan permasalahan charging station untuk mobil listrik. Dengan adanya sistem monitoring yang akan dibuat diharapkan dapat membantu proses pemantauan daya yang keluar dari charging station, dapat membantu perhitungan pemakaian setiap daya yang keluar, memudahkan proses pengaksesan monitoring dengan aplikasi yang sudah menggunakan teknologi perangkat bergerak (mobile).

Kata kunci: Stasiun Isi Ulang, Mobil Listrik, Arduino, Perangkat Bergerak, Peramalan

PERVASIVE COMPUTING IN MONITORING SYSTEM and FORECASTING ELECTRIC CAR CHARGING STATION WITH ARDUINO BASED ON ANDROID

Student's Name	**: SANINDYA LESARIO**
Student's ID	**: 5112100141**
Department	**: Teknik Informatika FTIF-ITS**
First Advisor	**: Dr. Eng. Radityo Anggoro, S.Kom., M.Sc.**
Second Advisor	**: Dr. Dimas Anton Asfani, ST., MT.**

Abstract

The existence of electric cars in Indonesia has shown some progress in the field of technological innovation, especially in the new fuel is electricity. To be used continously in a variety of purposes, electric car needs charging station as a supplier of electrical energy. In terms to charge electric cars, electric power measurement that comes out from the charging station with precise and accurate is needed. This is an important step to avoid overload currents received electric cars. Excessive load currents can trigger direct contact between voltage conductors that are not directly through the appropriate media, thus causing the current flow is not normal (short circuit).

Therefore, in this undergraduate thesis, i built a system that can monitor the condition of the power supplied from the charging station for electric cars who want to recharge. The flow of electric current out changes detected by the current sensor, and then processed using the Arduino microcontroller to do the data processing of electric current to be displayed on mobile devices. To be able to calculate the power consumption that would come (calculate the power consumption of next month), used forecasting algorithms. The data will be displayed on the monitoring system includes the electric power that comes out from the charging station, power forecasting reports.

This application is expected to solve the problems of charging station for electric cars. With the monitoring system to be built is expected to assist in the monitoring of the power coming out of the charging station, could help the power consumption's calculations, facilitates the process of power monitoring with mobile devices based application.

Keyword: Charging Station, Electric Car, Arduino, Mobile Device, Forecasting.

KATA PENGANTAR

بِسۡمِ ٱللَّهِ ٱلرَّحۡمَٰنِ ٱلرَّحِيمِ

Segala puji bagi Allah SWT, Tuhan semesta alam yang telah melimpahkan rahmat dan hidayah-Nya kepada penulis, sehingga tugas akhir berjudul **"Rancang Bangun Komputasi Pervasif pada Sistem Monitoring dan Forecasting Charging Station Mobil Listrik Dengan Arduino Berbasis Android"** ini dapat selesai tepat waktu.

Pengerjaan tugas akhir ini merupakan salah satu dari sekian banyak kesempatan yang saya dapatkan, untuk mendapatkan ilmu dan pengalaman berharga selama saya berada di kampus Teknik Informatika ITS ini. Dengan pengerjaan tugas akhir ini, saya menjadi semakin bisa untuk memanajemen waktu dan memanajemen diri sendiri, sehingga tugas akhir ini dapat selesai tepat waktu.

Selesainya tugas akhir ini tidak lepas dari bantuan dan dukungan beberapa pihak. Sehingga pada kesempatan ini penulis mengucapkan syukur dan terima kasih kepada:

1. Allah SWT dan Nabi Muhammad SAW.
2. Orang Tua dan keluarga penulis yang telah memberikan dukungan doa, moral, yang selalu memotivasi penulis untuk menyelesaikan tugas akhir ini.
3. Bapak Dr. Eng. Radityo Anggoro, S.Kom., M.Sc., selaku koordinator TA dan pembimbing I yang telah banyak membantu, membimbing, dan memberikan banyak motivasi kepada penulis dalam menyelesaikan tugas akhir ini.
4. Bapak Dr. Dimas Anton Asfani, ST., MT., selaku pembimbing II yang telah banyak membimbing dan memberikan arahan kepada penulis dalam menyelesaikan Tugas Akhir ini.
5. Dr. Ir. RV. Hari Ginardi, M.Kom., selaku dosen wali yang telah membimbing saya dari awal kuliah hingga saat ini, serta Bapak

Ibu Dosen lainnya yang telah memberikan ilmu - ilmu yang sangat bermanfaat .

6. Teman - teman *TC ROLAS*, yang selalu memberi doa san semangat kepada penulis untuk menyelesaikan tugas akhir ini.
7. Ubed, Ruli, Farid selaku rekan kerja tim tugas akhir yang ikut berkontribusi dalam penyelesaian tugas akhir ini.
8. Serta semua pihak yang yang telah turut membantu penulis dalam menyelesaikan tugas akhir ini

Penulis menyadari bahwa Tugas Akhir ini masih memiliki banyak kekurangan. Dengan kerendahan hati, penulis mengharapkan kritik dan saran dari pembaca untuk perbaikan ke depan.

Surabaya, Juni 2016

DAFTAR ISI

DAFTAR GAMBAR

[Halaman ini sengaja dikosongkan]

DAFTAR TABEL

[Halaman ini sengaja dikosongkan]

DAFTAR KODE SUMBER

[Halaman ini sengaja dikosongkan]

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan, batasan permasalahan, metodologi pembuatan tugas akhir, dan sistematika penulisan.

1.1 Latar Belakang

Keberadaan mobil listrik di Indonesia telah menunjukkan suatu kemajuan dalam bidang teknologi khususnya dalam inovasi bahan bakar baru yaitu listrik. Untuk dapat terus digunakan dalam berbagai keperluan, mobil listrik membutuhkan stasiun isi ulang atau *charging station* sebagai penyuplai energi listrik. Untuk pengisian mobil listrik, dibutuhkan pengukuran daya listrik yang keluar dari *charging station* dengan tepat dan akurat. Hal ini merupakan langkah penting untuk menghindari terjadinya kelebihan beban arus yang diterima mobil listrik. Beban arus yang berlebih dapat memicu terjadinya hubungan antar penghantar bertegangan secara langsung yang tidak melalui media yang semestinya sehingga menyebabkan terjadinya aliran arus yang tidak normal (korsleting).

Oleh karena itu dalam tugas akhir ini, saya membuat sebuah sistem yang dapat melakukan monitor kondisi daya listrik yang mengalir dari *charging station* ke mobil listrik yang ingin melakukan isi ulang. Aliran perubahan arus listrik yang keluar dideteksi oleh sensor arus, kemudian diproses menggunakan mikrokontroler *Arduino* untuk dilakukan pemrosesan. Data arus listrik yang telah didapat akan ditampilkan di perangkat bergerak (*mobile*) dalam bentuk aplikasi. Untuk dapat menghitung pemakaian daya yang akan datang, digunakan algoritma *forecasting* / peramalan. Data yang akan ditampilkan pada aplikasi sistem monitoring meliputi daya listrik yang keluar dari *charging station*, laporan peramalan daya.

Aplikasi ini diharapkan dapat menyelesaikan permasalahan *charging station* untuk mobil listrik. Dengan adanya sistem monitoring yang akan dibuat diharapkan dapat membantu proses

pemantauan daya yang keluar dari *charging station*, dapat membantu perhitungan pemakaian setiap daya yang keluar, memudahkan proses pengaksesan monitoring dengan aplikasi yang sudah menggunakan teknologi perangkat bergerak (*mobile*).

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam tugas akhir ini dapat dipaparkan sebagai berikut:

1. Bagaimana rancangan sistem monitoring *charging station* berbasis teknologi perangkat bergerak (mobile)?
2. Bagaimana melakukan pengambilan data dari sumber daya di *charging station*, pengolahan, hingga proses menampilkan di perangkat bergerak (mobile)?
3. Bagaimana mengimplementasikan metode peramalan *moving average* di perangkat bergerak (mobile)?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam tugas akhir ini memiliki beberapa batasan, diantaranya sebagai berikut:

1. Implementasi dilakukan dengan menggunakan bahasa pemrograman *Arduino*, *Java*, *PHP*, dan *HTML*.
2. Sensor arus AC yang digunakan adalah *LEM LA 55 - P*.
3. Sensor tegangan AC yang digunakan adalah *ZMPT101B*.
4. Teknologi perangkat bergerak (*mobile*) yang digunakan berbasis *Android*.
5. Mikrokontroler yang digunakan ialah *Arduino Mega 2560* dengan *Ethernet Shield* sebagai modul untuk terhubung dengan jaringan internet.
6. Router yang digunakan untuk membuat jaringan lokal ialah *TP-LINK*.

1.4 Tujuan

Tujuan dari tugas akhir ini adalah sebagai berikut:

1. Mengetahui rancangan sistem monitoring daya *charging station* pada perangkat bergerak (*mobile*).
2. Melakukan pengambilan data dari sumber daya di *charging station*, pengolahan, hingga proses menampilkan di perangkat bergerak (*mobile*).
3. Mengimplementasikan metode peramalan *moving average* di perangkat bergerak (*mobile*).

1.5 Manfaat

Dengan tugas akhir ini diharapkan mendapatkan manfaat antara lain:

1. Mendapatkan kemudahan dalam mengakses sistem monitoring *charging station* dengan menggunakan perangkat bergerak (*mobile*).
2. Mendapatkan hasil prediksi perhitungan daya listrik dari *charging station* dengan menggunakan metode peramalan *moving average*.

1.6 Metodologi

Berikut adalah metodologi pembuatan tugas akhir:

a. Penyusunan Proposal Tugas Akhir

Tahap awal untuk memulai pengerjaan tugas akhir adalah penyusunan proposal tugas akhir. Proposal tugas akhir yang diajukan memiliki gagasan untuk mengimplementasikan rancang bangun sistem monitoring pada tugas akhir ini.

b. Studi Literatur

Pada studi literatur ini, akan dipelajari sejumlah referensi yang diperlukan dalam pembuatan aplikasi yaitu

mengenai sensor arus, sensor tegangan, algoritma *forecasting* atau peramalan, mikrokontroler *Arduino*, kemudian *Android* sebagai *mobile platform* dari aplikasi.

c. Perancangan Perangkat Lunak

Tahap ini meliputi perancangan perangkat lunak yang akan dikembangkan dan langkah-langkah yang dikerjakan. Pada tahapan ini dibuat prototipe aplikasi sistem, metode yang digunakan, serta dilakukan desain sistem dan desain proses-proses yang ada.

d. Implementasi Perangkat Lunak

Pada tahap ini dilakukan implementasi perangkat lunak. Perangkat lunak dibuat berdasarkan rancangan yang telah dibuat sebelumnya sehingga menjadi sebuah aplikasi sistem yang sesuai dengan apa yang telah direncanakan.

e. Pengujian dan Evaluasi

Tahapan ini dimaksudkan untuk melakukan uji coba pada fungsionalitas aplikasi sistem dan metode yang telah dirancang. Tahapan ini dimaksudkan untuk mengevaluasi fungsionalitas dan tingkat akurasi dari metode aplikasi sistem tersebut, serta mencari masalah yang mungkin timbul dan mengadakan perbaikan jika terdapat kesalahan.

f. Penyusunan Buku Tugas Akhir

Tahap ini merupakan tahap penyusunan dari buku tugas akhir. Dalam buku ini dijelaskan dasar teori, metode, perancangan, implementasi, pengujian, dan kesimpulan dari pengerjaan tugas akhir.

1.7 Sistematika Penulisan

Buku tugas akhir ini bertujuan untuk mendapatkan gambaran dari pengerjaan tugas akhir ini. Selain itu, diharapkan dapat berguna untuk pembaca yang tertarik untuk melakukan pengembangan lebih lanjut. Secara garis besar, buku tugas akhir terdiri atas beberapa bagian seperti berikut ini:

Bab I Pendahuluan

Bab ini berisi mengenai latar belakang masalah, tujuan, dan manfaat dari pembuatan tugas akhir, permasalahan, batasan masalah, metodologi yang digunakan, dan sistematika penyusunan tugas akhir.

Bab II Dasar Teori

Bab ini membahas beberapa teori penunjang yang berhubungan dengan pokok pembahasan dan mendasari pembuatan tugas akhir ini.

Bab III Perancangan Perangkat Lunak

Bab ini berisi tentang desain aplikasi sistem dan metode yang disajikan dalam bentuk *flowchart*.

Bab IV Implementasi

Bab ini berisi implementasi dari perancangan yang telah dibuat sebelumnya. Penjelasan berupa *pseudocode* yang digunakan untuk proses implementasi.

Bab V Uji Coba Dan Evaluasi

Bab ini menjelaskan kemampuan aplikasi sistem dengan melakukan pengujian kebutuhan fungsional dan pengujian kinerja dari sistem yang telah dibuat.

Bab VI Kesimpulan Dan Saran

Bab ini merupakan bab terakhir yang menyampaikan kesimpulan dari hasil uji coba yang dilakukan dan saran untuk pengembangan perangkat lunak kedepannya.

BAB II
TINJAUAN PUSTAKA

Pada bab ini akan dijelaskan mengenai landasan teori yang berkaitan tentang implementasi aplikasi sistem. Penjelasan ini bertujuan untuk memberikan gambaran secara umum terhadap aplikasi yang dibuat dan berguna sebagai penunjang dalam pengembangan aplikasi.

2.1 Stasiun Pengisian Kendaraan Listrik

Charging station atau biasa disebut dengan *EV (Electric Vehicle) charging station* adalah sebuah stasiun pengisian yang menyuplai energi listrik untuk pengisian kembali kendaraan listrik, seperti mobil listrik, motor listrik, dan sejenisnya.

Komponen *charging station* seperti pada Gambar 2.1 tidak terlalu berbeda dengan stasiun pengisian bahan bakar pada umumnya. Terdapat panel yang mengatur pengisian daya listrik, konektor berbentuk seperti *jack* / *plug* yang dihubungkan ke kendaraan listrik untuk mengalirkan daya. Beberapa mempunyai fitur tambahan yaitu sistem pembayaran elektronik, akses sistem dengan menggunakan kartu, dan lainnya.

Gambar 2.1 Charging Station [1]

Terdapat 2 standar yang digunakan untuk komponen *charging station*. Yang pertama standar yang dipublikasikan oleh *SAE International* yaitu *SAE J1772*. Standar ini mendukung tipe isi ulang dengan arus AC maupun DC. Seluruh kendaraan listrik memakai socket standar *J1772* kecuali *Tesla Motors* yang mengembangkan sendiri sistem isi ulang cepat dengan arus DC yang dinamai *Supercharger*. Standar yang kedua yaitu *CHAdeMO* yang hanya mendukung jenis isi ulang cepat dengan arus DC.

Ada 3 level dalam pengisian ulang pada charging station berdasarkan karakteristiknya. Berikut perbandingannya pada Tabel 2.1 di bawah ini.

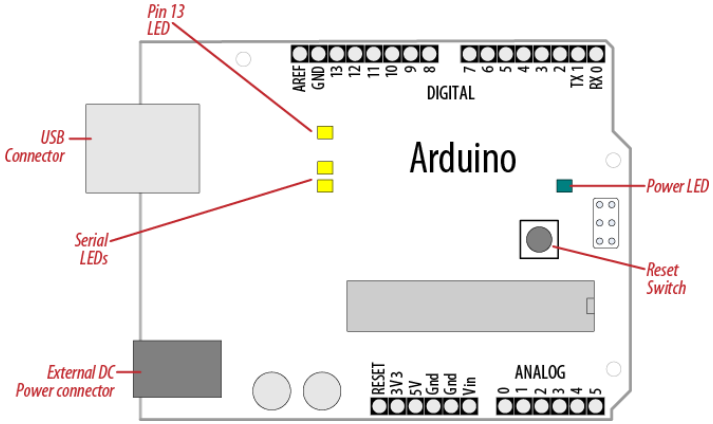
Tabel 2.1 Perbedaan Level Pengisian Ulang

	Level 1	Level 2	Level 3
Tegangan	120 V	208 or 240 V	200 to 450 V
Tipe arus	AC	AC	DC
Daya yang digunakan	1.4 kW	7.2 kW	50 kW
Keluaran maksimal	1.9 kW	19.2 kW	150 kW
Waktu isi ulang	12 jam	3 jam	20 menit
Konektor	J1772	J1772	J1772 Combo, *CHAdeMO*, dan *Supercharger*

2.2 Mikrokontroler Arduino

Arduino merupakan mikrokontroler yang terdiri dari perangkat keras dan perangkat lunak. Perangkat keras Arduino yaitu sebuah papan atau board yang sudah dilengkapi oleh komponen - komponen yang membuat perangkat ini dapat berinteraksi dengan

sensor, lampu LED, pengeras suara. Untuk berinteraksi dengan komputer pada papan *Arduino* dilengkapi dengan penghubung USB. Gambaran mengenai skema papan *Arduino* dapat dilihat pada Gambar 2.2 dibawah ini. Perangkat lunak *Arduino* yaitu berupa *integrated development environment* (IDE). Perangkat lunak ini gratis dan bersifat *open source*. Dengan menggunakan IDE pengguna dapat membuat perintah program yang akan dieksekusi oleh perangkat *keras Arduino*. Proses transfer kode ke perangkat keras *Arduino* disebut *uploading*.



Gambar 2.2 Skema papan *ArduinoUno* [2]

2.3 Sensor Arus LA 55- P

Sensor *LA 55-P* adalah sensor yang bagus untuk mendeteksi adanya arus baik itu arus DC maupun arus AC. Sensor ini memiliki output arus yang mana perbandingannya adalah 1 : 1000. Jika input tegangan 1 ampere maka outputnya adalah 0,001 A. Mikrokontroler tidak bisa mendeteksi arus, maka arus tersebut haruslah dikonversi menjadi tegangan dengan menggunakan resistor, Selain untuk mengubah tegangan, resistor juga diposisikan untuk penguat yang ditambahkan pada rangkaian *non-inverting amplifier* agar tegangan

yang terbaca bisa terlihat oleh mikrokontroler. Bentuk sensor arus yang digunakan dapat dilihat pada Gambar 2.3

Gambar 2.3 Sensor Arus *LA 55 - P* [3]

2.4 Sensor Tegangan ZMPT101B

Sensor tegangan dapat mengukur tegangan AC maupun DC tergantung penggunaan algoritma pengukuran. Perbedaan antara kedua jenis tegangan tersebut yaitu tegangan DC bernilai konstan, sedangkan tegangan AC berubah - ubah mengikuti gelombang sinus. Sensor tegangan ZMPT101B yang dirancang menggunakan transformator seperti pada Gambar 2.4 hanya dapat digunakan untuk membaca tegangan AC. Jika sebuah sensor tegangan ingin digunakan untuk membaca tegangan AC dan DC, maka sensor tegangan harus dirancang dengan sirkuit pembagi tegangan. Sensor ZMPT101B memiliki perbandingan output tegangan yaitu 1000 : 1000. Jika input tegangan 1000 volt maka outputnya yaitu 1000 volt. Sensor ZMPT101B dapat digunakan untuk pengukuran energi kelistrikan, peralatan listrik rumah tangga.peralatan industri, pengujian peralatan listrik, proteksi relay, dan lain - lain.

Gambar 2.4 Sensor Tegangan *ZMPT101B* [4]

2.5 Ethernet Shield

Arduino Ethernet Shield yaitu merupakan perangkat tambahan yang memungkinkan *Arduino* yang kita gunakan dapat terhubung ke dengan jaringan internet. Yang membuat *Ethernet Shield* dapat berfungsi seperti ini adalah dengan adanya keping pengatur *ethernet* yaitu *W5100*. Keping ini mendukung jaringan *TCP* dan *UDP* dan 4 koneksi soket secara bersamaan. Terdapat suatu mekanisme yang dapat mencegah serangan jaringan seperti: *flooding*, *spoofing*, *injection*. Mempunyai memori internal 16 KB untuk memproses paket *TCP/IP*. Gambar 2.5 menunjukkan keping pengatur *ethernet*.

Gambar 2.5 Keping *W5100* [5]

Untuk dapat menggunakan *shield* ini, cukup dengan menancapkan di atas *Arduino* dengan memasukkannya ke *SPI port*. Untuk *Arduino Uno* ditancapkan ke pin 10, 11, 12, 13 dan pin 50, 51, 52 untuk *Arduino Mega*.

Gambar 2.6 *Arduino Mega* dengan *Ethernet Shield* [6]

Perangkat tambahan ini menggunakan standar *RJ-45* sebagai koneksi dan sebagai sumber daya. Terdapat tempat *micro-SD* yang menjadi tempat penyimpanan. Fitur ini dapat diakses dengan menggunakan *SD library* pada *IDE Arduino*. Tombol *reset* pada *shield* dapat mengatur ulang *Arduino* dan *Ethernet Shield*. Gambar 2.6 menunjukkan *Ethernet Shield* yang dipasang dengan *Arduino Mega*.

2.6 Android

Android merupakan sebuah sistem operasi yang bersifat *open source*. Teknologi *Android* dikembangkan dari kernel Linux. Dengan ini memungkinkan komunikasi tingkat bawah dengan perangkat keras, manajemen memori, kontrol proses. Teknologi ini juga mendukung library yang bersifat *open source* untuk menunjang pengembangan aplikasi seperti *SQLite*, *WebKit*, *OpenGL*. Aplikasi yang dapat dikembangkan dengan menggunakan sistem operasi *Android* yaitu aplikasi pemutar musik, kalkulator, manajemen informasi pribadi. [7]

2.7 Peramalan *Moving Average*

Forecasting atau peramalan adalah tentang memprediksi masa depan seakurat mungkin, mengingat semua informasi yang tersedia, termasuk data historis dan pengetahuan dari setiap peristiwa masa depan yang mungkin berdampak pada perkiraan. Adapun metode - metode yang digunakan untuk melakukan forecasting atau peramalan:

- Metode Kualitatif:

  Metode yang digunakan ketika data yang akan digunakan untuk peramalan tidak relevan. Metode ini cenderung objektif dan hasilnya akan tergantung oleh orang yang menyusunnya. Metode ini meliputi metode delphi, analogi, survey pasar, dan lain - lain.

- Metode Kuantitatif:

  Metode ini digunakan apabila tersedia data - data numerik yang valid dari masa lalu, dapat diasumsikan akan terus berlanjut ke masa yang akan datang.

Ada berbagai metode peramalan kuantitatif, sering dikembangkan dalam disiplin ilmu tertentu untuk tujuan tertentu. Setiap metode memiliki sifat sendiri, akurasi, dan biaya yang harus diperhatikan ketika memilih metode tertentu. Sebagian besar masalah peramalan kuantitatif menggunakan baik data *time series* (dikumpulkan secara berkala dari waktu ke waktu) atau data *cross-sectional* (dikumpulkan pada satu titik dalam waktu).

Moving Averages (rata-rata bergerak) adalah metode peramalan perataan nilai dengan mengambil sekelompok nilai pengamatan yang kemudian dicari rata - ratanya, lalu menggunakan rata - rata tersebut sebagai ramalan untuk periode berikutnya. Istilah rata - rata bergerak digunakan, karena setiap kali data observasi baru tersedia, maka angka rata-rata yang baru dihitung dan dipergunakan sebagai ramalan. [8]

Untuk menentukan ramalan pada periode yang akan datang memerlukan data historis selama jangka waktu tertentu. Misalnya, dengan 3 bulan *moving average*, maka ramalan bulan ke - 5 baru dibuat setelah bulan ke - 4 selesai/berakhir. Jika bulan *moving averages* bulan ke - 7 baru bisa dibuat setelah bulan ke - 6 berakhir. Semakin panjang jangka waktu *moving average*, efek pelicinan semakin terlihat dalam ramalan atau menghasilakan *moving average* yang semakin halus.

Persamaan matematis single moving averages adalah sebagai berikut:

$$\hat{Y}_{t+1} = \frac{Y_t + Y_{t-1} + \cdots + Y_{t-n+1}}{m}$$

Dimana:

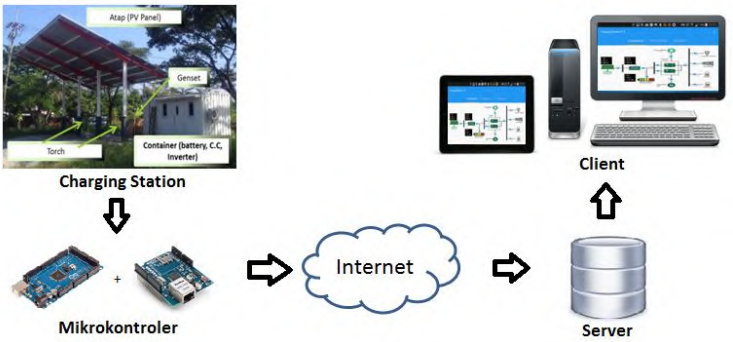
Y_{t+1} = Nilai ramalan untuk periode (t + 1)
Y_t = Nilai sebenarnya untuk periode ke - t
m = Jumlah batas dalam *moving average*

BAB III
DESAIN PERANGKAT

Bab ini membahas tahap perancangan aplikasi yang dibuat pada tugas akhir ini. Perancangan akan dibagi menjadi dua proses utama, yaitu perancangan perangkat keras dan perancangan perangkat lunak.

3.1 Deskripsi Umum Sistem

Perangkat lunak yang akan dibangun pada pengerjaan tugas akhir ini yaitu sebuah aplikasi perangkat bergerak berbasis *Android* yang dapat memantau kondisi daya listrik yang keluar dari komponen penyuplai daya di stasiun pengisian isi ulang kendaraan listrik / *charging station*. Komponen penyuplai daya yang dipilih yaitu panel *Photovoltaics (PV)*, baterai, dan *charge controller* (*MPPT*). Selain hasil keluaran daya dari masing - masing komponen penyuplai, aplikasi ini juga akan menampilkan prediksi daya yang akan keluar. Metode yang digunakan untuk implementasi prediksi daya yaitu *moving average*. Pengguna juga dapat melihat keseluruhan data daya keluaran yang telah dipantau dengan fitur melihat database pada aplikasi.

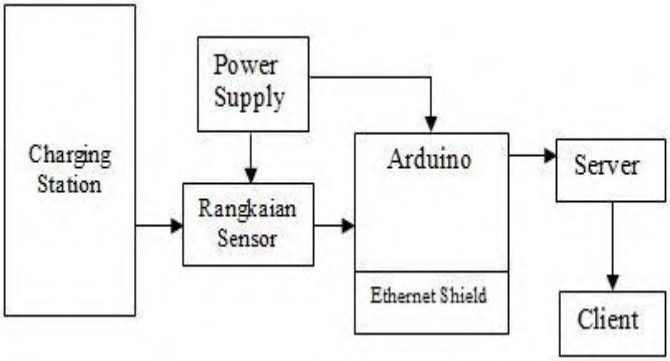


Gambar 3.1 Alur Kerja Sistem

Pada Gambar 3.1 ditunjukkan gambaran umum mengenai alur kerja sistem. Untuk membaca daya yang keluar, digunakan sensor arus dan sensor tegangan yang diletakkan pada masing - masing komponen penyuplai daya. Sensor tersebut dihubungkan dengan mikrokontroler *Arduino Mega 2560* yang telah dipasangi *Ethernet Shield* agar dapat terhubung ke jaringan internet. Data analog yang dibaca oleh sensor kemudian dilakukan linierisasi. Hal ini dilakukan agar data yang ditampilkan merupakan data dengan nilai pembacaan yang sebenarnya.

Setelah nilai sensor dapat dibaca, kemudian mikrokontroler memasukkan data tersebut ke *database server*. Untuk dapat ditampilkan pada perangkat bergerak, data yang ingin ditampilkan diubah ke dalam format *JSON*. Format *JSON* merupakan layanan web yang memungkinkan data dapat diakses oleh perangkat jenis lain (dalam hal ini perangkat bergerak).

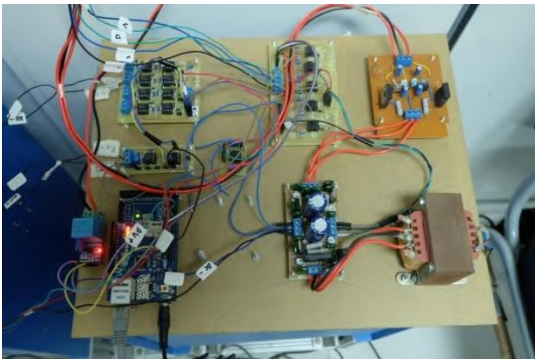
3.2 Perancangan Perangkat Keras



Gambar 3.2 Diagram blok sistem

Dalam perancangan perangkat keras dibutuhkan alat - alat sebagai berikut: mikrokontroler *Arduino Mega 2560*. *Ethernet Shield W5100*, sensor arus *LA 55- P*, sensor tegangan *ZMPT101B*, *power supply*, dan router. Berdasarkan alur diagram blok sistem pada Gambar 3.2, data arus dan tegangan akan dibaca oleh rangkaian

sensor yang telah dipasang pada setiap komponen penyuplai daya. Rangkaian sensor dan mikrokontroler dihubungkan dengan *power supply*. Dengan adanya komponen dasar penyusun power supply yaitu *IC 7812* dan *IC 7912*, sehingga membuatnya dapat menghasilkan tegangan stabil +12V. Tegangan ini dibutuhkan untuk menyuplai sensor arus *LA 55-P*. Untuk memberi suplai ke sensor tegangan *ZMPT101B* sebanyak + 5V, pada *power supply* ditambahkan komponen IC 7805T. Mikrokontroler dan modul *ethernet* menggunakan +5V dan +12V sebagai suplai tenaga. Untuk lebih jelasnya dapat dilihat pada Gambar 3.3 di bawah ini.



Gambar 3.3 Rancangan Perangkat Keras Sistem

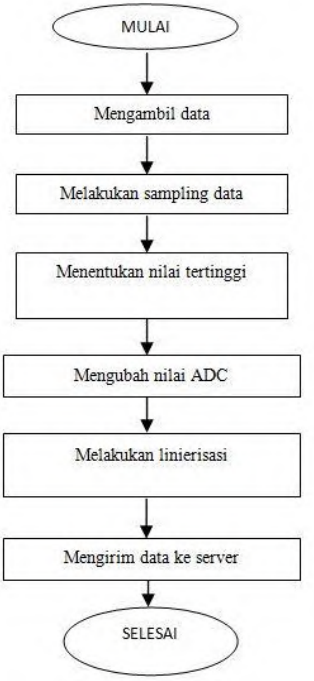
3.3 Perancangan Perangkat Lunak

Pada bagian ini, akan dibahas secara keseluruhan mengenai rancangan untuk menampilkan data - data yang telah dibaca oleh sensor, diproses di mikrokontroler hingga ditampilkan di aplikasi perangkat bergerak (*mobile*). Perancangan yang dilakukan meliputi pembacaan nilai dari sensor, fungsional aplikasi sistem, hingga rancangan desain antarmuka aplikasi sistem.

3.3.1 Metode Pembacaan Nilai Arus dan Tegangan AC

Setelah sensor arus LA 55-P dan sensor tegangan ZMPT101B dihubungkan dengan *Arduino* melalui pin ADC, data

yang didapat terlebih dahulu masuk ke rangkaian *summing amplifier*. Rangkaian ini dibuat untuk mengubah nilai yang dibaca sensor AC menjadi positif. Kemudian dilakukan sampling sebanyak 300 data. Oleh karena hasil keluaran dari rangkaian *summing amplifier* berupa gelombang sinusoidal, maka untuk mendapat nilai tertinggi yaitu dengan mengambil titik puncak. Setelah itu, nilai yang didapat dalam bentuk ADC dikembalikan seperti semula. Resolusi ADC bernilai 10 bit, maka skala penuhnya bernilai 1023. Setiap nilai yang dibaca oleh sensor akan dibagi dengan jumlah bit yaitu 1024 . Untuk mendapat nilai bacaan sensor arus dan tegangan yang sebenarnya dilakukan proses linierisasi dengan menggunakan metode regresi linier. Setelah itu baru data bacaan sensor dimasukkan ke *database server*. Alur proses pembacaan sensor arus dan tegangan AC dapat dilihat pada Gambar 3.4 dibawah ini.



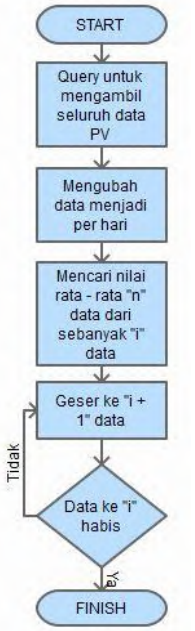
Gambar 3.4 *Flowchart* membaca arus dan tegangan AC

3.2.2 Metode Peramalan Moving Average

Untuk mendapatkan prediksi data dalam aplikasi sistem monitoring *charging station* ini, penulis menggunakan metode peramalan *moving average*. Data yang akan diramal diambil dari salah satu komponen penyuplai daya untuk charging station yaitu *PV*. Data yang akan diramal berupa arus, tegangan, dan daya.

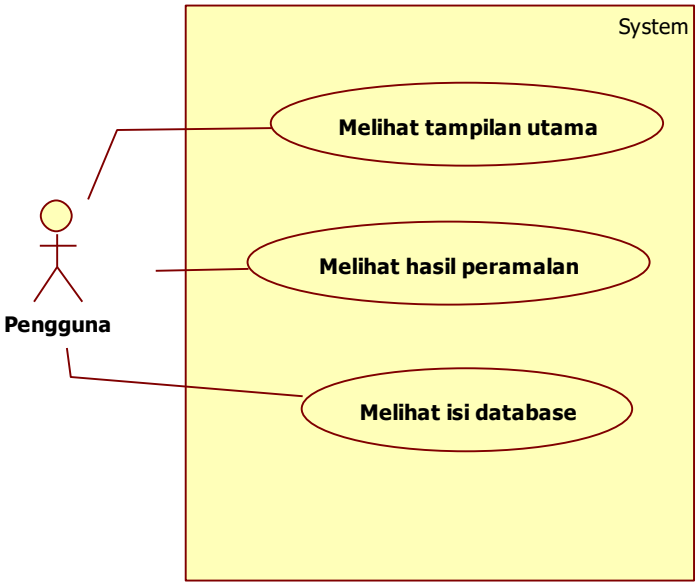
Seperti bab sebelumnya, proses meramal data dengan *moving average* yaitu dengan mencari nilai rata - rata dari *n* data (dimana n = periode moving average yang ditentukan), kemudian dihitung kembali dengan bergeser ke data selanjutnya sebanyak periode *n*, dan begitu seterusnya sampai data habis.

Pada kasus ini,data yang masuk ke database diatur oleh mikrokontroler setiap 5 menit sekali. Kemudian data - data tersebutakan dicari nilai rata - rata untuk per harinya. Setelah itu diambil data sebanyak 20 hari untuk dilakukan peramalan.



Gambar 3.5 *Flowchart* peramalan *moving average*

3.3.2 Diagram Kasus Pengguna

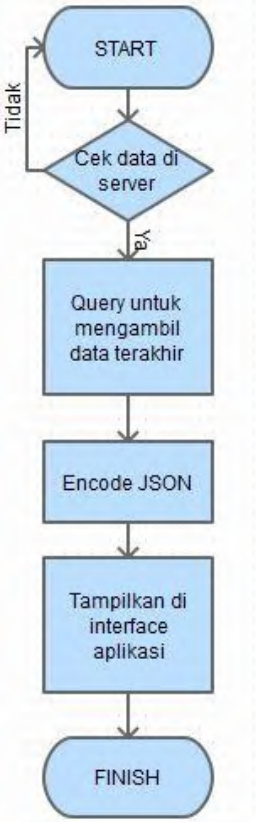


Gambar 3.6 Diagram kasus pengguna

Berdasarkan Gambar 3.6 terdapat 3 fungsi utama dari aplikasi yaitu melihat tampilan utama, melihat hasil peramalan, melihat isi database. Fungsi yang pertama pengguna dapat *melihat tampilan utama*. Pada menu ini, pengguna dapat melihat data yang dibaca oleh sensor dan terakhir diperbarui dari database.Fungsi yang kedua yaitu, pengguna dapat *melihat hasil ramalan*. Hasil ramalan yang ditampilkan yaitu data dari komponen *charging station* yang telah masuk *database* akan dilakukan peramalan sesuai metode yang ditentukan. Setelah itu untuk fungsi ketiga yaitu pengguna dapat *melihat isi database*. Fitur ini dibuat untuk mempermudah dalam membandingkan tingkat keakuratan hasil peramalan dengan data asli.

3.3.3 Fitur Melihat Tampilan Utama

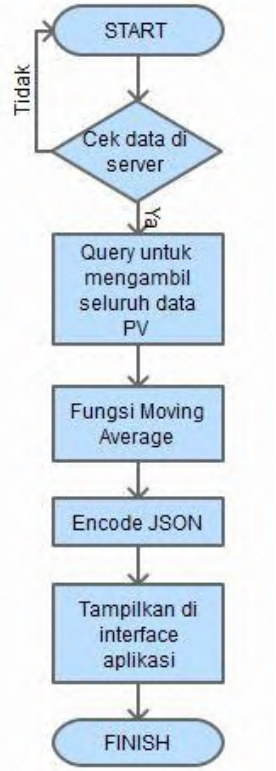
Pada Gambar 3.7 proses dalam menampilkan data dari mikrokontroler diawali dari mengecek data yang akan ditampilkan. Kemudian pilih isi data dari *PV* yaitu arus, tegangan, daya, yang terakhir masuk ke dalam *database*. Setelah itu ubah dalam bentuk *JSON* agar dapat dibaca oleh aplikasi lain. Setelah itu pada aplikasi perangkat bergerak, data yang berbentuk *JSON* diubah menjadi bentuk *String* untuk ditampilkan di antarmuka perangkat bergerak.



Gambar 3.7 *Flowchart* melihat tampilan utama

3.3.4 Fitur Melihat Hasil Peramalan

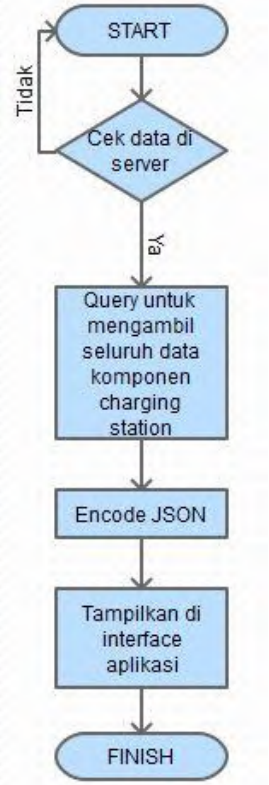
Pada Gambar 3.8 proses melihat hasil peramalan diawali dari mengecek data yang akan ditampilkan sudah ada di *database server* atau belum. Kemudian pilih keseluruhan isi data dari *PV* yaitu arus, tegangan, daya, yang terakhir masuk ke dalam *database*. Setelah semuanya dipilih, kemudian masukkan kedalam array untuk diproses di fungsi peramalan dengan metode *moving average*. Setelah itu ubah dalam bentuk *JSON* agar dapat dibaca oleh aplikasi lain. Setelah itu pada aplikasi perangkat bergerak, data yang berbentuk *JSON* diubah menjadi bentuk *String* untuk ditampilkan di antarmuka perangkat bergerak.



Gambar 3.8 *Flowchart* melihat hasil ramalan

3.3.5 Fitur Melihat Isi Database

Pada Gambar 3.9 proses dalam menampilkan data dari mikrokontroler diawali dari mengecek data yang akan ditampilkan sudah ada di *database server* atau belum. Kemudian pilih keseluruhan isi data dari *PV* yaitu arus, tegangan, daya, yang terakhir masuk ke dalam *database*. Setelah itu ubah dalam bentuk *JSON* agar dapat dibaca oleh aplikasi lain. Setelah itu pada aplikasi perangkat bergerak, data yang berbentuk *JSON* diubah menjadi bentuk *String* untuk ditampilkan di antarmuka perangkat bergerak.

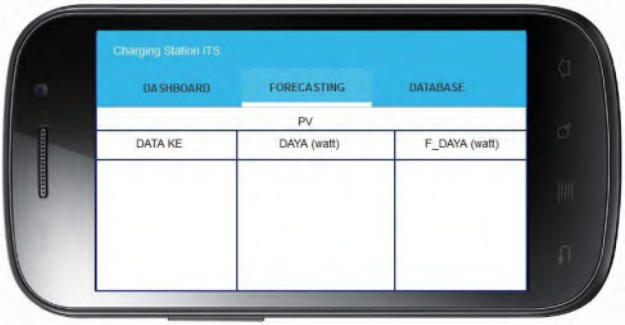


Gambar 3.9 *Flowchart* melihat isi *database*

3.3.6 Antarmuka Sistem

Gambar 3.10 Tampilan dashboard pada aplikasi monitoring

Seperti pada rancangan diagram kasus pengguna, fitur melihat tampilan utama akan ditampilkan dengan menu *dashboard* pada aplikasi sistem monitoring *charging station*. Pada Gambar 3.10 dapat dilihat untuk nilai yang dibaca oleh sensor dan sudah masuk di *database* akan ditampilkan pada kotak berwarna kuning, kemudian untuk gambar aliran daya dari setiap komponen di *charging station* hingga ke mobil listrik akan diletakkan di kotak berwarna biru.



Gambar 3.11 Tampilan peramalan pada aplikasi monitoring *charging station*

Data hasil pembacaan sensor yang telah masuk *database* akan dilakukan peramalan dan akan dimasukkan ke menu *forecasting* pada aplikasi sistem monitoring *charging station*. Berdasarkan

Gambar 3.11 data yang akan ditampilkan yaitu berasal dari *PV*. Parameter yang akan ditampilkan yaitu adalah arus, tegangan, dan daya. Kolom *data ke* digunakan untuk menampilkan hasil peramalan dalam hitungan hari.

Gambar 3.12 Tampilan database pada aplikasi monitoring *charging station*

Menu *database* ini dirancang untuk memudahkan pengguna untuk membandingkan data yang telah diramal dengan data yang telah diolah oleh sensor dan mikrokontroler. Pada Gambar 3.12 tampilan menu ini berupa tabel dengan data yang ditampilkan berasal dari *PV*. Parameter yang akan ditampilkan yaitu adalah arus, tegangan, dan daya. Kolom *data ke* digunakan untuk menampilkan hasil peramalan dalam hitungan hari.

[Halaman ini sengaja dikosongkan]

BAB IV
IMPLEMENTASI

Pada bab ini akan dijelaskan implementasi dari perancangan untuk aplikasi sistem monitoring *charging station* yang telah dijelaskan pada bab sebelumnya. Sebelum masuk ke implementasi, terlebih dahulu penulis memberikan informasi lingkungan implementasi itu sendiri.

4.1 Lingkungan Implementasi

Pada implementasi sistem, digunakan beberapa perangkat pendukung sebagai berikut:

1. Arduino Mega 2560 dan *Ethernet Shield*.
2. Sensor Arus AC LA 55 - P dan Sensor Tegangan AC ZMPT101B.
3. Laptop HP Pavilion g4 i7-3612QM @ 2.10Ghz 4GB RAM.
4. Smartphone Samsung Galaxy Note II GT - N7100.
5. Arduino *Development Kit* versi 1.6.3 sebagai IDE untuk mengimplementasikan kode program arduino
6. Sublime sebagai *text editor* untuk mengimplementasikan kode program aplikasi monitoring
7. Router yang digunakan untuk membuat jaringan lokal ialah TP-LINK.
8. Modem smartfren 3G.

4.2 Implementasi

Pada bagian ini penulis membagi pembahasan implementasi menjadi dua bagian yaitu implementasi perangkat keras dan perangkat lunak. Untuk implementasi perangkat keras akan dijelaskan implementasi sensor dan mikrokontrolernya, kemudian untuk implementasi perangkat lunak akan dijelaskan mengenai kebutuhan fungsional dari aplikasi sistem monitoring *charging station* itu sendiri.

4.2.1 Implementasi Perangkat Keras

Pada bagian ini akan dijelaskan implementasi rangkaian sensor dan mikrokontroler dari perancangan pada bab sebelumnya. Pembahasan dimulai dari pembacaan sensor hingga mengirimkan data ke *server*.

4.2.1.1 Implementasi Pembacaan Nilai Arus dan Tegangan AC

Data hasil bacaan sensor masuk melalui pin ADC pada mikrokontroler. Kemudian dilakukan pengambilan data sebanyak 300 untuk dijadikan sampel. Setelah itu diambil beberapa data yang menempati titik puncak, karena data berupa sinyal sinusoidal. Setelah itu, nilai yang didapat dalam bentuk ADC dikembalikan seperti semula. Resolusi ADC bernilai 10 bit, maka skala penuhnya bernilai 1023. Setiap nilai yang dibaca oleh sensor akan dibagi dengan jumlah bit yaitu 1024 . Untuk mendapat nilai bacaan sensor arus dan tegangan yang sebenarnya dilakukan proses linierisasi dengan menggunakan metode *regresi linier*.

```
1  for i = 0 to 300
2    set start_times[i] to micros()
3    set values[i] to analogRead(A0)
4    if values[i] >= peak0
5      then set peak0 to values[i]
6    set stop_times[i] to micros()
7  set adc0 to (peak0 * 5) / 1023
8  set linear0 to(adc0 - 2.5221) / 0.0103
```

Kode Sumber 4.1 *Pseudeocode* implementasi pembacaan nilai arus dan tegangan AC

4.2.1.3 Implementasi Web Service pada Mikrokontroler

Dengan menggunakan *Ethernet Shield*, mikrokontroler dapat terhubung ke jaringan internet. Untuk menjalankannya dari Arduino gunakan *library Ethernet Shield*. Setelah itu atur *port* dan *host* ke lokasi *server*. Data hasil bacaan sensor dimasukkan ke dalam satu variable yaitu *txtData3*. Gunakan metode *HTTP POST* untuk

mengirimkan data dari mikrokontroler *Arduino* ke *server*. Data yang telah dikirim akan ditangani oleh *web service*.

```
initialize client to EthernetClient
if client.connect(server, 80)
    then
    set   txData3   to   "pv_arus="+   (String
(pv_arus))    +    "&pv_tegangan="+    (String
(pv_tegangan))    +    "&pv_daya="+    (String
(pv_daya))    +    "&mppt_arus="+    (String
(mppt_arus))  +  "&mppt_tegangan="+  (String
(mppt_tegangan))+   "&mppt_daya="+   (String
(mppt_daya))+      "&bat_arus="+      (String
(bat_arus));
print "connected"
print "POST/insert2.php    HTTP/1.1"
print "Host: www.chargingstation.its.ac.id"
print "Connection: close"
print  "Content-Type:application/x-www-form-
urlencoded\n"
print"Content-Length: "
print txData2.length()
print"\n\n"
print txData2
set delay to 1000

end if
  print "Connection Failed”
set delay to 1000
```

Kode Sumber 4.2 *Pseudeocode* implementasi web service

4.2.2.1 Implementasi Pengiriman Data ke Database

Data yang telah dikirim oleh mikrokontroler Arduino akan masuk ke *server* akan diterima oleh *web service*. Untuk memasukkan data tersebut ke *database*, dibutuhkan *query* atau komunikasi dengan *database server* dengan menggunakan perintah *insert*. Kemudian untuk method yang digunakan yaitu *POST* karena ingin melakukan pengiriman data ke *database server*.

```
1   load config3.php
2
3   set query to "INSERT INTO tbcharging_2 (
4                                 pv_arus,
5                                 pv_tegangan,
6                                 pv_daya,
7                                 mppt_arus,
8                                 mppt_tegangan,
9                                 mppt_daya,
10                                bat_arus)
11  VALUES
12  (
13                  '$_POST[pv_arus]',
14                  '$_POST[pv_tegangan]',
15                  '$_POST[pv_daya]',
16                  '$_POST[mppt_arus]',
17                  '$_POST[mppt_tegangan]',
18                  '$_POST[mppt_daya]',
19                  '$_POST[bat_arus]'
20  )"
```

Kode Sumber 4.3 *Pseudocode* memasukkan data ke server

4.2.2 Implementasi Perangkat Lunak

Pada bagian ini akan dibahas mengenai implementasi kebutuhan fungsional dari aplikasi sistem monitoring yang telah direncanakan pada bab sebelumnya.

4.2.2.2 Implementasi Fitur Melihat Tampilan Utama

Pada fitur ini, data yang akan ditampilkan akan dimuat dalam bentuk *textview* untuk setiap komponen penyuplai daya di *charging station*. Kemudian terdapat skema aliran daya berupa *ImageView*. Pada Kode Sumber 4.4 berisi kode dengan bahasa *PHP* untuk mengambil data dari *database* dan mengubahnya ke format *JSON*. Format *JSON* yang digunakan yaitu *JSON array*. Hal ini dilakukan untuk mempermudah proses pemanggilan data yang akan ditampilkan di perangkat bergerak, karena hanya memanggil *JSON array* saja.

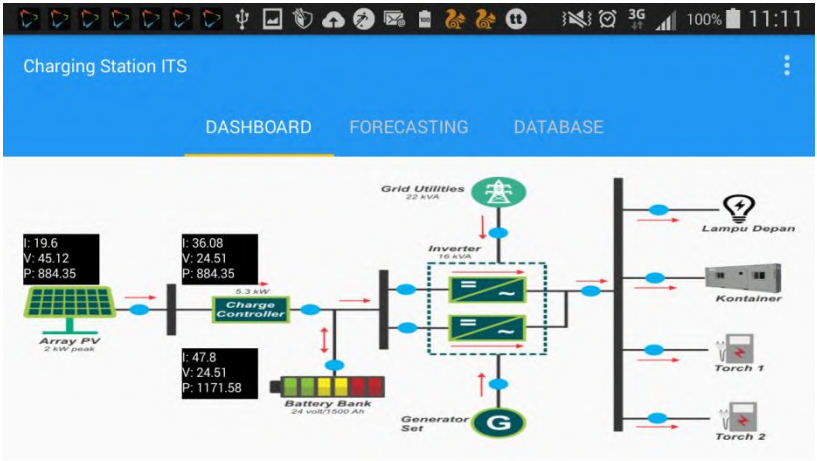
```
1   void function showData(){
2   initialize conn
3
4   set query to "SELECT id, waktu, pv_arus,
5   pv_tegangan, mppt_arus, mppt_tegangan,
6   mppt_daya, bat_arus, bat_tegangan, bat_daya,
7   round(pv_arus * pv_tegangan,2) AS pv_daya FROM
8   tbcharging_2 ORDER BY id DESC LIMIT 1;"
9
10  set result to mysqli_query(conn, query)
11  set number_of_rows to  mysqli_num_rows(result)
12
13  set temp_array to array()
14  if number_of_rows > 0
15    then
16    while set row to mysqli_fetch_assoc(result))
17     do
18     set temp_array[] to row
19
20  echo
21  json_encode(array("tbcharging"=>temp_array))
22  mysqli_close($conn)
```

Kode Sumber 4.4 *Pseudocode* mengambil data untuk tampilan utama

Setelah berformat *JSON*, data tersebut kemudian diproses di perangkat bergerak dengan mengubah kembali format *JSON* menjadi bentuk *String*. Kemudian dimasukkan ke dalam *textview* untuk ditampilkan di perangkat bergerak seperti pada Kode Sumber 4.5. Proses yang dilakukan yaitu dengan mengambil nama *JSON* yang mempunyai format *array*, karena *JSON* yang telah dibentuk pada Kode Sumber 4.4 mempunyai format *array*. Setelah itu setiap isi dari *JSON array* yaitu *JSON object* akan diambil sebanyak isi *array* tersebut. Setiap isi yang telah diambil dijadikan format *String*. Selanjutnya, setiap isi yang mempunyai format *String* dimasukkan ke dalam format *textView*.

```
void dashboard(){
initialize jsonObjectRequest
set jsonObjectRequest = new
JsonObjectRequest(Request.Method.POST,
"http://chargingstation.its.ac.id/android_dashboard.php"
, null, new Response.Listener<JSONObject>() {
try
   set jsonArray = create new JSONArray
   set jsonArray to tbcharging
   for i = 0 to tbcharging
      set jsonObject = create new JSONObject
      set jsonObject to isi
      set isi to jsonObject(i)

          set isi.pv_arus to pv_arus
          set isi.pv_tegangan to pv_tegangan
          set isi.pv_daya to pv_daya

          set isi.mppt_arus to mppt_arus
          set isi.mppt_arus to mppt_arus
          set isi.mppt_arus to mppt_arus

          set isi.bat_arus to bat_arus
          set isi.bat_arus to bat_tegangan
          set isi.bat_daya to bat_daya

          add pv_arus to isiPV
          add pv_tegangan to isiPV
          add pv_daya to isiPV

          add mppt_arus to isiMPPT
          add mppt_tegangan to isiMPPT
          add mppt_daya to isiMPPT

          add bat_arus to isiBaterai
          add bat_tegangan to isiBaterai
          add bat_daya to isiBaterai

catch JSONException e
   e.printStackTrace()

add jsonObjectRequest to requestQueue
```

Kode Sumber 4.5 *Pseudocode* menampilkan tampilan utama di aplikasi



Gambar 4.1 *Pseudocode* tampilan fitur melihat tampilan utama

4.2.2.3 Implementasi Fitur Melihat Hasil Peramalan

Pada fitur ini, data yang telah diramal akan ditampilkan akan dimuat dalam bentuk *textview*. Data yang akan ditampilkan yaitu arus, tegangan, dan daya dari *PV*. Pada Kode Sumber 4.6 berisi kode dengan bahasa *PHP* untuk mengambil data dari database, memprosesnya pada fungsi peramalan dan mengubahnya ke format *JSON*.

```
1   void function movingAverage
2   initialize conn
3
4   set sql to "SELECT id, waktu, pv_arus,
5   pv_tegangan, mppt_arus, mppt_tegangan,
6   mppt_daya, bat_arus, bat_tegangan,
7   bat_daya, round(pv_arus * pv_tegangan,2)
8   AS pv_asli FROM tbcharging_2 WHERE id < 26
9
10  set hasil to mysqli_query(conn, sql)
11  set number_of_rows to
12  mysqli_num_rows(hasil)
13
14  set data_asli to array()
```

```
set data1 to array()

if number_of_rows > 0
   then
   while row = mysqli_fetch_assoc(hasil))
   do
       set data_asli[] to row
       set data1[] to row['pv_asli']

set periode to 3
set jumlah1 to 0

for i=0 to periode
   set jumlah1 to jumlah1 + data1[i]

set data_f to array()
set data_terakhir to count(data1)

for i=periode to data_terakhir
   set data_f[] to array(
          "id_f" => i,
          "pv_f"     =>round((jumlah1     /
(periode)),2),)

set   jumlah1   to   jumlah1   -   data1[i   -
periode] + data1[i]

echo
json_encode(array("tbcharging"=>data_asli,
"tbcharging_f"=>data_f))

mysqli_close(conn)
```

Kode Sumber 4.6 Pseudocode mengambil data untuk hasil peramalan

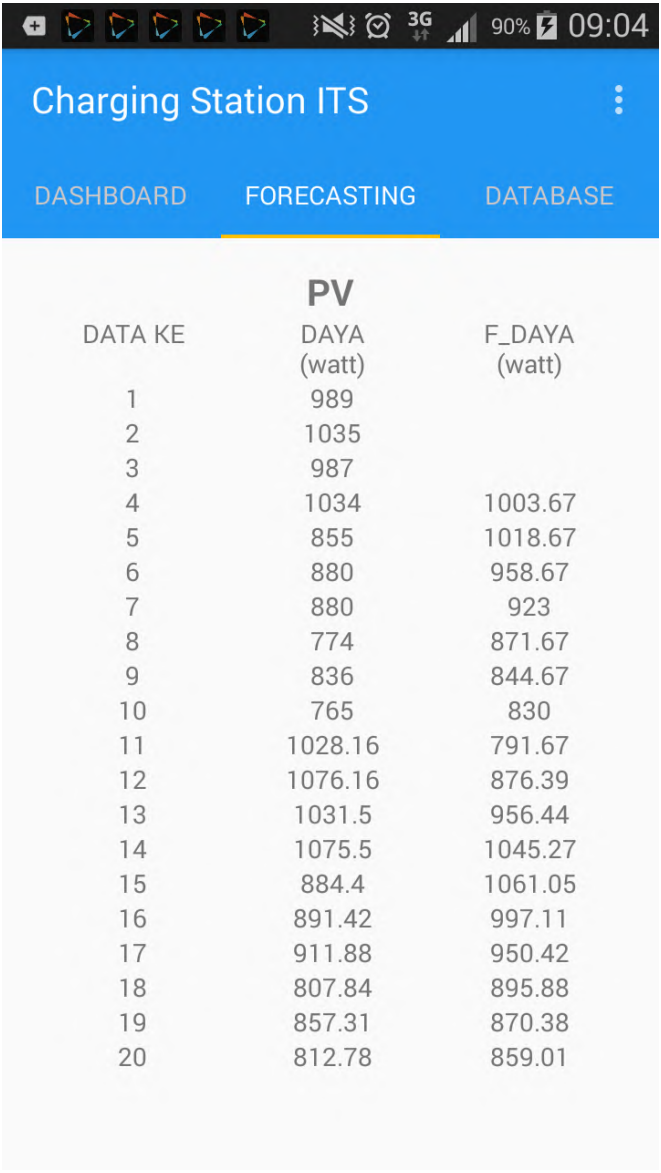
Setelah berformat JSON, data tersebut kemudian diproses di perangkat bergerak dengan mengubah kembali format *JSON* menjadi bentuk *string*. Kemudian akan dimasukkan ke dalam *textview* untuk ditampilkan di perangkat bergerak seperti pada Kode Sumber 4.7.

Proses yang dilakukan yaitu dengan mengambil nama *JSON* yang mempunyai format *array*, karena *JSON* yang telah dibentuk pada Kode Sumber 4.6 mempunyai format *array*. Setelah itu setiap

isi dari *JSON array* yaitu *JSON object* akan diambil sebanyak isi *array* tersebut. Setiap isi yang telah diambil dijadikan format *String*. Selanjutnya, setiap isi yang mempunyai format String dimasukkan ke dalam format *textView*. Masing - masing *textView* dimasukkan ke variabel *tableRow3* untuk dapat ditampilkan dalam bentuk tabel.

```
void addDataForecasting(){
initialize jsonObjectRequest
set jsonObjectRequest = new
JsonObjectRequest(Request.Method.POST,
"http://chargingstation.its.ac.id/android_forecasting.php"
, null, new Response.Listener<JSONObject>() {
try
   set jsonArray = create new JSONArray
   set jsonArray to tbcharging_f
   for i = 0 to tbcharging_f
      set jsonObject = create new JSONObject
      set jsonObject to isi
      set isi to jsonObject(i)

          set isi.id to id
          set isi.pv_asli to pv_asli
          set isi.pv_f to pv_f

          add id to txtView1
          add pv_asli to txtView2
          add pv_f to txtView3

          add txtView1 to tableRow3
          add txtView2 to tableRow3
          add txtView3 to tableRow3

          add tableRow3 to tableMain

catch JSONException e
   e.printStackTrace()

add jsonObjectRequest to requestQueue
```

Kode Sumber 4.7 Pseudocode menampilkan hasil ramalan di aplikasi



Gambar 4.2 *Pseudocode* tampilan fitur melihat hasil peramalan

4.2.2.4 Implementasi Fitur Melihat Isi Database

Pada fitur ini, data dari PV yang sudah tersimpan dalam database akan ditampilkan dalam bentuk *textView*. Data yang akan ditampilkan yaitu arus, tegangan, dan daya dari *PV*. Pada Kode Sumber 4.8 berisi kode dengan bahasa *PHP* untuk mengambil data dari database dan mengubahnya ke format *JSON*. Format JSON yang digunakan yaitu *JSON array*. Hal ini dilakukan untuk mempermudah proses pemanggilan data yang akan ditampilkan di perangkat bergerak, karena hanya memanggil *JSON array* saja.

```
1   void function showData(){
2   initialize conn
3
4   set query to "SELECT id, waktu, pv_arus,
5   pv_tegangan, mppt_arus, mppt_tegangan,
6   mppt_daya, bat_arus, bat_tegangan, bat_daya,
7   round(pv_arus * pv_tegangan,2) AS pv_daya
8   FROM tbcharging_2 WHERE id < 26;"
9
10  set result to mysqli_query(conn, query);
11  set number_of_rows to
12  mysqli_num_rows(result)
13
14  set temp_array to array()
15
16  if number_of_rows > 0
17     then
18     while row = mysqli_fetch_assoc(result))
19     do
20     set temp_array[] to row
21
22  echo
23  json_encode(array("tbcharging"=>temp_array))
24
25  mysqli_close(conn)
26
27
28
29
```

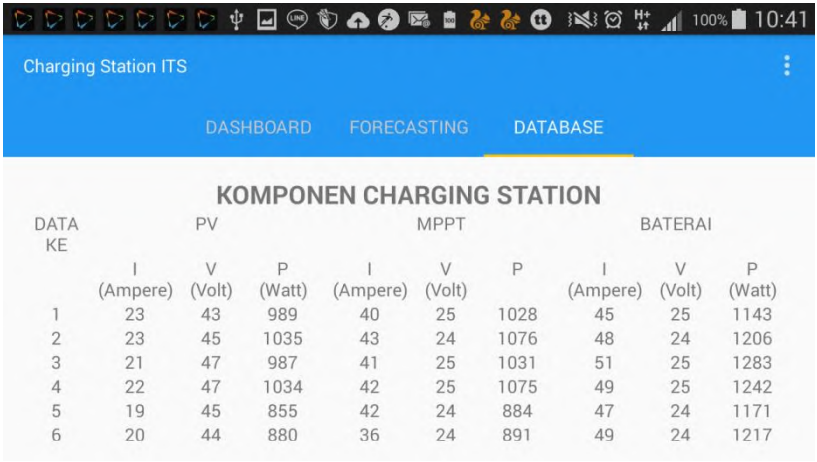
Kode Sumber 4.8 *Pseudocode* mengambil data untuk isi database

Setelah berformat *JSON*, data tersebut kemudian diproses di perangkat bergerak dengan mengubah kembali format *JSON* menjadi bentuk *String*. Kemudian akan dimasukkan ke dalam *textview* untuk ditampilkan di perangkat bergerak seperti pada Kode Sumber 4.9.

```
void addDataTable(){
initialize jsonObjectRequest
set jsonObjectRequest = new
JsonObjectRequest(Request.Method.POST,
"http://chargingstation.its.ac.id/android_database.php"
, null, new Response.Listener<JSONObject>() {
try
   set jsonArray = create new JSONArray
   set jsonArray to tbcharging
   for i = 0 to tbcharging
      set jsonObject = create new JSONObject
      set jsonObject to isi
      set isi to jsonObject(i)

          set isi.id to id
          set isi.pv_arus to pv_arus
          set isi.pv_tegangan to pv_tegangan
          set isi.pv_daya to pv_daya

          set isi.mppt_arus to mppt_arus
          set isi.mppt_arus to mppt_arus
          set isi.mppt_arus to mppt_arus

          set isi.bat_arus to bat_arus
          set isi.bat_arus to bat_tegangan
          set isi.bat_daya to bat_daya

          add id to txtView1
          add pv_arus to txtView2
          add pv_tegangan to txtView3
          add pv_daya to txtView4

          add mppt_arus to txtView5
          add mppt_tegangan to txtView6
          add mppt_daya to txtView7

```

```
38
39              add bat_arus to txtView8
40              add bat_tegangan to txtView9
41              add bat_daya to txtView10
42
43              add txtView1 to tableRow3
44              add txtView2 to tableRow3
45              add txtView3 to tableRow3
46              add txtView4 to tableRow3
47              add txtView5 to tableRow3
48              add txtView6 to tableRow3
49              add txtView7 to tableRow3
50              add txtView8 to tableRow3
51              add txtView9 to tableRow3
52              add txtView10 to tableRow3
53
54              add tableRow3 to tableMain
55
56  catch JSONException e
57     e.printStackTrace()
58
59  add jsonObjectRequest to requestQueue
60
```

Kode Sumber 4.9 *Pseudocode* menampilkan isi database pada aplikasi



Gambar 4.3 *Pseudocode* tampilan fitur melihat isi database

[Halaman ini sengaja dikosongkan]

BAB V
UJI COBA DAN EVALUASI

Pada bab ini akan dijelaskan uji coba yang dilakukan pada aplikasi yang telah dikerjakan serta analisa dari uji coba yang telah dilakukan. Pembahasan pengujian meliputi lingkungan uji coba, skenario uji coba yang meliputi uji tingkat akurasi dan uji kinerja serta analisa setiap pengujian.

5.1 Lingkungan Uji Coba

Pada subbab ini menjelaskan tentang lingkungan yang digunakan untuk menguji fungsionalitas dan tingkat akurasi dari implementasi sistem yang telah dirancang. Lingkungan uji coba menggunakan bantuan perangkat lunak dan perangkat keras dengan spesifikasi sebagai berikut:

1. Perangkat Keras:
   a. Laptop HP Pavilion g4 i7-3612QM @ 2.10Ghz
   b. RAM : 4,00 GB
   c. Tipe sistem : 64-bit sistem operasi
   d. Smartphone Samsung Galaxy Note II GT - N7100

2. Perangkat Lunak:
   a. Sistem Operasi : *Windows 10*
   b. *IDE : arduino 1.6.3*
   c. Sebuah *web browser*

5.2 Skenario Uji Coba

Uji coba ini dilakukan untuk menguji apakah fungsionalitas program telah diimplementasikan dengan benar dan berjalan sesuai rencana. Pada sistem ini, akan dilakukan uji coba fungsionalitas sistem dan uji coba performa atau tingkat akurasi metode peramalan.

5.2.1 Uji Coba Fungsionalitas

Pada uji coba fungsionalitas akan diuji fungsi - fungsi utama pada komponen sensor dan aplikasi monitoring. Keberhasilan pada uji coba ini akan menunjang kualitas dari uji coba yang selanjutnya yaitu uji coba performa atau tingkat akurasi sistem.

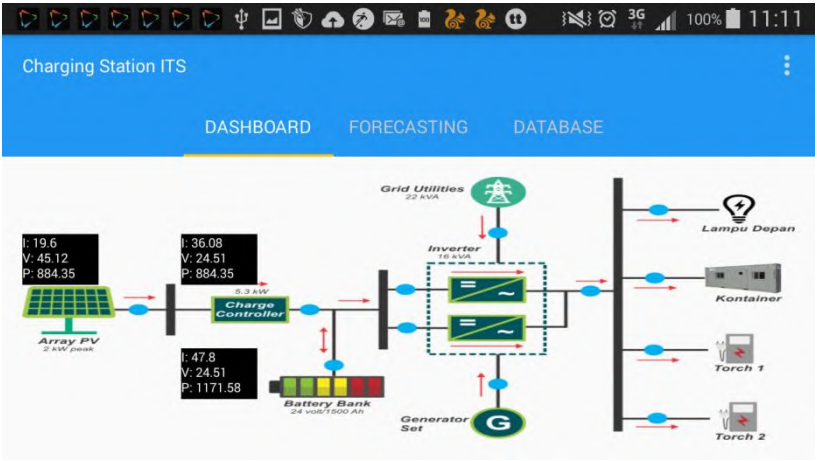
5.2.1.1 Uji Coba Melihat Tampilan Utama

Uji coba untuk fitur ini yaitu dengan membandingkan isi dari komponen *charging station* yang ditampilkan di perangkat bergerak sudah sesuai dengan yang ada di *database server*. Fitur ini akan menampilkan data dari setiap komponen penyuplai daya *charging station*. Pada Gambar 5.1 dapat dilihat bahwa *id* terakhir pada database server yaitu 309. Data yang ditampilkan merupakan data yang terakhir masuk ke *database*. Perubahan nilai yang ditampilkan pada menu *dashboard* tergantung pada isi *database* masing - masing komponen penyuplai daya. Untuk dapat melihat apakah pergantian data dapat berlangsung yaitu dengan cara melakukan *reload* ulang aplikasi. Melakukan *reload* ulang bisa dengan mengganti ke menu lainnya kemudian kembali ke menu *dashboard*.

Berdasarkan Gambar 5.2, terlihat perubahan nilai pada setiap kotak. Setiap nilai komponen di *database server* diletakkan sesuai dengan kotak yang mewakili komponen penyuplai daya masing - masing. Untuk nilai *pv_arus*, *pv_tegangan*, dan *pv_daya* diletakaan pada kotak yang terdapat di atas gambar *PV*, begitu juga dengan data *baterai* dan *mppt*. Hal ini menunjukkan menu *dashboard* sudah dapat menampilkan data dari setiap komponen penyuplai data dari *charging station*.

id	pv_arus	pv_tegangan	mppt_arus	mppt_tegangan	mppt_daya	bat_arus	bat_tegangan	bat_daya
306	4.7	65.3	12.35	24.5	302.57	29.8	24.5	730.1
307	4.7	65.3	12.35	24.5	302.57	31.3	24.5	766.85
309	19.6	45.12	36.08	24.51	884.35	47.8	24.51	1171.58

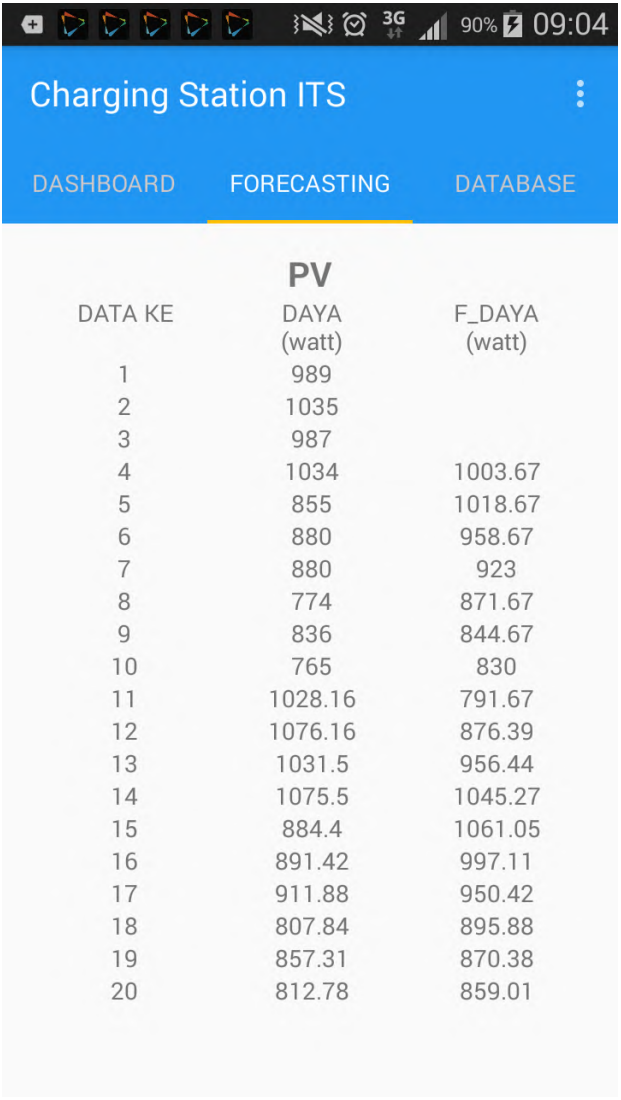
Gambar 5.1 Isi database terakhir



Gambar 5.2 Tampilan fitur melihat tampilan utama

5.2.1.2 Uji Coba Melihat Hasil Peramalan

Uji coba dilakukan dengan cara membandingkan data daya *PV* asli yang ada di *database server* dan hasil peramalan yang telah dihitung dengan menggunakan fungsi *moving average* dengan data yang ditampilkan pada menu *forecasting*. Data yang ditampilkan merupakan daya dari *PV* yang telah masuk database dan telah melalui proses peramalan dengan metode *moving average*. Berdasarkan Gambar 5.3, pada setiap kolom yang disediakan sudah terdapat nilai. Terdapat 3 kolom yaitu *data ke*, *daya*, dan *f*_daya. Untuk kolom *data ke* menampilkan banyaknya data yang digunakan yaitu sebanyak 20 data. Kemudian untuk kolom *daya* digunakan untuk menampilkan daya *PV* asli yang digunakan sebagai perbandingan dengan daya *PV* yang telah diramal. Kolom *f_daya* yaitu berisi daya PV yang telah diramal. Tampilan berbentuk seperti tabel untuk mempermudah pengguna dalam membaca hasil peramalan. Hal ini menunjukkan menu *forecasting* sudah dapat menampilkan data dari setiap komponen penyuplai data dari *charging station*.



Gambar 5.3 Tampilan melihat hasil peramalan

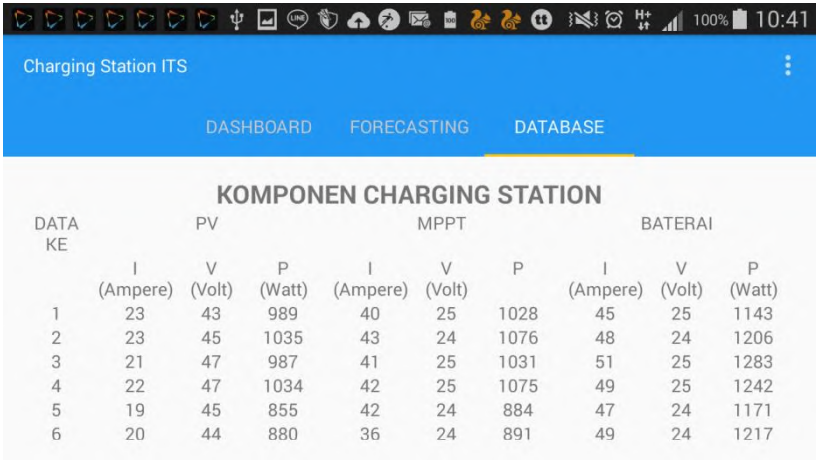
5.2.1.3 Uji Coba Melihat Database

Uji coba fitur ini yaitu dengan cara membandingkan isi *database* pada *server* dengan yang ditampilkan di menu database pada perangkat bergerak. Data yang ditampilkan dari komponen *charging station* yaitu meliputi arus, tegangan, dan daya dari pv, mppt, dan juga baterai.

pv_arus	pv_tegangan	mppt_arus	mppt_tegangan	mppt_daya	bat_arus	bat_tegangan	bat_daya
23	43	40	25	1028	45	25	1143
23	45	43	24	1076	48	24	1206
21	47	41	25	1031	51	25	1283
22	47	42	25	1075	49	25	1242
19	45	42	24	884	47	24	1171
20	44	36	24	891	49	24	1217
20	44	34	26	911	49	26	1298
18	43	30	26	807	52	26	1370
19	44	32	26	857	50	26	1339
17	45	30	26	812	51	26	1373
23.8	43.2	40.64	25.3	1028.16	45.2	25.3	1143.56
23.6	45.6	43.17	24.93	1076.16	48.4	24.93	1206.61
21.9	47.1	41.06	25.12	1031.49	51.1	25.12	1283.63
22.5	47.8	42.85	25.1	1075.5	49.5	25.1	1242.45
19.6	45.12	36.08	24.51	884.35	47.8	24.51	1171.58
20.2	44.13	36.02	24.75	891.43	49.2	24.75	1217.7
20.3	44.92	34.91	26.12	911.88	49.7	26.12	1298.16
18.7	43.2	30.83	26.2	807.84	52.3	26.2	1370.26
19.3	44.42	32.57	26.32	857.31	50.9	26.32	1339.69

Gambar 5.4 Tampilan isi seluruh database

Berdasarkan Gambar 5.5, pada setiap kolom yang disediakan sudah terdapat nilai. Terdapat 4 kolom yaitu *data ke*, *PV*, *MPPT*, dan *BATERAI*. Untuk kolom *data ke* menampilkan banyaknya data yang ditampilkan. Sedangkan ketiga kolom lainnya berisi setiap komponen penyuplai daya yang ditampilkan nilainya.



DATA KE	PV			MPPT			BATERAI		
	I (Ampere)	V (Volt)	P (Watt)	I (Ampere)	V (Volt)	P	I (Ampere)	V (Volt)	P (Watt)
1	23	43	989	40	25	1028	45	25	1143
2	23	45	1035	43	24	1076	48	24	1206
3	21	47	987	41	25	1031	51	25	1283
4	22	47	1034	42	25	1075	49	25	1242
5	19	45	855	42	24	884	47	24	1171
6	20	44	880	36	24	891	49	24	1217

Gambar 5.5 Tampilan melihat isi database

5.2.2 Uji Coba Performa Metode Peramalan

Pada uji coba performa akan dilakukan pengujian untuk mengukur tingkat akurasi dan seberapa besar *error* yang didapat dari prediksi daya dengan menggunakan metode *moving average*.

Peramalan ini menggunakan angka *moving average* sebanyak 3. Jadi setiap 3 data akan dilakukan rata – rata untuk mendapat prediksi hari selanjutnya. Setelah itu bergeser 1 data ke data selanjutnya yaitu 2, 3, 4 untuk mendapat prediksi data ke-5. Proses ini dilakukan hingga data mencapai akhir.

Penghitungan error dilakukan dengan menggunaka rumus RMSE yaitu *root means square error*. Untuk seluruh mengecek error dari setiap data asli dikurangi dengan data yang telah diprediksi. Hasilnya dikuadratkan kemudian dijumlah. Setelah itu dibagi sebanyak jumlah *forecasting* yang telah dilakukan dalam skenario yaitu 17. Nilai RMSE yang telah didapat yaitu 101,05. Untuk mendapat nilai akurasi yang baik, nilai RMSE harus mendekati 0. [9]

$$RMSE = \sqrt{\frac{\sum(Y_t - Y_{t+1})^2}{n}}$$

n = banyaknya operasi

Untuk mengetahui apakah nilai moving average (m) dapat mempengaruhi hasil peramalan, uji coba dilakukan dengan menggunakan 4 macam nilai m yaitu 3, 6, 9, 12. Tabel uji coba untuk setiap nilai *m* bisa dilihat di bawah ini.

Tabel 5.1 Untuk m = 3

DATA KE	PV		ERROR	
	ASLI	FORECAST	A - F	$(A - F)^2$
1	989	-	-	-
2	1035	-	-	-
3	987	-	-	-
4	1034	1003.67	30.33	919.9089
5	855	1018.67	-163.67	26787.87
6	880	958.67	-78.67	6188.969
7	880	923	-43	1849
8	774	871.67	-97.67	9539.429
9	836	844.67	-8.67	75.1689
10	765	830	-65	4225
11	1028.16	791.67	236.49	55927.52
12	1076.16	876.39	199.77	39908.05
13	1031.5	956.44	75.06	5634.004
14	1075.5	1045.27	30.23	913.8529
15	884.4	1061.05	-176.65	31205.22
16	891.42	997.11	-105.69	11170.38
17	911.88	950.42	-38.54	1485.332
18	807.84	895.88	-88.04	7751.042
19	857.31	870.38	-13.07	170.8249
20	812.78	859.01	-46.23	2137.213
Jumlah	-	-	-	205888.8

DATA KE	PV		ERROR	
	ASLI	FORECAST	A - F	$(A - F)^2$
RMSE	-	-	-	**110.0505**

Tabel 5.2 Untuk m = 6

DATA KE	PV		ERROR	
	ASLI	FORECAST	A - F	$(A - F)^2$
1	989	-	-	-
2	1035	-	-	-
3	987	-	-	-
4	1034	-	-	-
5	855	-	-	-
6	880	-	-	-
7	880	963.3333	-83.3333	6944.444
8	774	945.1667	-171.167	29298.03
9	836	901.6667	-65.6667	4312.111
10	765	876.5	-111.5	12432.25
11	1028.16	831.6667	196.4933	38609.63
12	1076.16	860.5267	215.6333	46497.73
13	1031.5	893.22	138.28	19121.36
14	1075.5	918.47	157.03	24658.42
15	884.4	968.72	-84.32	7109.862
16	891.42	976.7867	-85.3667	7287.468
17	911.88	997.8567	-85.9767	7391.987
18	807.84	978.4767	-170.637	29116.87
19	857.31	933.7567	-76.4467	5844.093
20	812.78	904.725	-91.945	8453.883
Jumlah	-	-	-	247078.1
RMSE	-	-	-	**128.3428**

Tabel 5.3 Untuk m = 9

DATA KE	PV		ERROR	
	ASLI	FORECAST	A - F	$(A - F)^2$
1	989	-	-	-
2	1035	-	-	-
3	987	-	-	-
4	1034	-	-	-
5	855	-	-	-
6	880	-	-	-
7	880	-	-	-
8	774	-	-	-
9	836	-	-	-
10	765	918.8889	-153.889	23681.79
11	1028.16	894	134.16	17998.91
12	1076.16	893.24	182.92	33459.73
13	1031.5	903.1467	128.3533	16474.58
14	1075.5	902.8689	172.6311	29801.5
15	884.4	927.3689	-42.9689	1846.325
16	891.42	927.8578	-36.4378	1327.712
17	911.88	929.1267	-17.2467	297.4475
18	807.84	944.4467	-136.607	18661.38
19	857.31	941.3178	-84.0078	7057.307
20	812.78	951.5744	-138.794	19263.9
Jumlah	-	-	-	169870.6
RMSE	-	-	-	**124.269**

Tabel 5.4 Untuk m = 12

DATA KE	PV		ERROR	
	ASLI	FORECAST	A - F	$(A - F)^2$
1	989	-	-	-
2	1035	-	-	-
3	987	-	-	-
4	1034	-	-	-
5	855	-	-	-
6	880	-	-	-
7	880	-	-	-
8	774	-	-	-
9	836	-	-	-
10	765	-	-	-
11	1028.16	-	-	-
12	1076.16	-	-	-
13	1031.5	928.2767	103.2233	10655.06
14	1075.5	931.8183	143.6817	20644.42
15	884.4	935.1933	-50.7933	2579.963
16	891.42	926.6433-	-35.2233	1240.683
17	911.88	914.7617	2.88167	8.304003
18	807.84	919.5017	-111.662	12468.33
19	857.31	913.4883	-56.1783	3156.005
20	812.78	911.5975	-98.8175	9764.898
Jumlah	-	-	-	60517.66
RMSE	-	-	-	**59.66458**

Tabel 5.5 Perbandingan Hasil Uji Coba Nilai m

Periode Moving Average (m)	RMSE
3	110.0505
6	128.3428
9	124.269
12	59.66458

Berdasarkan perbandingan pada tabel 5.5, memang terdapat perbedaan nilai error dari RMSE. Ketika angka *m* dinaikkan menjadi 6 hingga 9 terdapat peningkatan nilai error. Akan tetapi setelah diuji coba dengan nilai *m* yang lebih tinggi, nilai error menjadi lebih rendah dari pengujian pertama kali dengan nilai $m = 12$. Hal yang menyebabkan ini terjadi adalah perubahan data yang satu dengan yang lainnya terlalu signifikan, sehingga dibutuhkan data yang banyak dan data yang tidak terlalu signifikan untuk mendapatkan hasil prediksi yang optimal.

[Halaman ini sengaja dikosongkan]

LAMPIRAN

1. Berikut adalah potongan kode untuk fitur melihat tampilan utama

```
<?php

	define('HOST','localhost');
	define('USER','cc_dbcharging');
	define('PASS','yanuar');
	define('DB','dbcharging');
	$conn = mysqli_connect(HOST,USER,PASS,DB) or die('Unable to
Connect');

	if ($_SERVER["REQUEST_METHOD"]=="POST") {

		showData();
	}

	function showData(){

		global $conn;

		$query = "SELECT id, waktu, pv_arus, pv_tegangan,
mppt_arus, mppt_tegangan, mppt_daya, bat_arus, bat_tegangan,
bat_daya, round(pv_arus * pv_tegangan,2) AS pv_daya FROM
tbcharging_2 ORDER BY id DESC LIMIT 1;";

		$result = mysqli_query($conn, $query);
		$number_of_rows = mysqli_num_rows($result);

		$temp_array = array();

		if ($number_of_rows > 0) {
			while ($row = mysqli_fetch_assoc($result)) {
				$temp_array[] = $row;
			}
		}
		echo json_encode(array("tbcharging"=>$temp_array));
		mysqli_close($conn);
	}
?>
```

Kode Sumber A.1 Mengambil data untuk tampilan utama

```
public class Tab_1 extends Fragment {

    Button btnMakeJsonRequest;
    TextView txtResponse;

    private static String TAG =
MainActivity.class.getSimpleName();

    // temporary string to show the parsed response
    String jsonResponse;

    String urlJson =
"http://chargingstation.its.ac.id/android_dashboard.php";
    RequestQueue requestQueue;

    // Progress dialog
    ProgressDialog pDialog;

    TextView isiPV;
    TextView isiMPPT;
    TextView isiBaterai;

    public Tab_1() {
        // Required empty public constructor
    }


    @Override
    public View onCreateView(LayoutInflater inflater,
ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment

        pDialog = new ProgressDialog(getActivity());
        pDialog.setMessage("Please Wait... ");
        pDialog.setCancelable(false);

        requestQueue =
Volley.newRequestQueue(getActivity());

        View v =
inflater.inflate(R.layout.fragment_tab_1, container,
false);
```

```
        isiPV = (TextView) v.findViewById(R.id.isiPV);
        isiMPPT = (TextView)
v.findViewById(R.id.isiMPPT);
        isiBaterai = (TextView)
v.findViewById(R.id.isiBaterai);

        dashboard();
        return v;
    }

    void dashboard(){

        JsonObjectRequest jsonObjectRequest = new
JsonObjectRequest(Request.Method.POST, urlJson, null, new
Response.Listener<JSONObject>() {
            @Override
            public void onResponse(JSONObject jsonObject)
{
                try {
                    JSONArray tbcharging =
jsonObject.getJSONArray("tbcharging");
                    for (int i = 0; i
<tbcharging.length(); i++){
                        JSONObject isi =
tbcharging.getJSONObject(i);

                        String pv_arus =
isi.getString("pv_arus");
                        String pv_tegangan =
isi.getString("pv_tegangan");
                        String pv_daya =
isi.getString("pv_daya");

                        String mppt_arus =
isi.getString("mppt_arus");
                        String mppt_tegangan =
isi.getString("mppt_tegangan");
                        String mppt_daya =
isi.getString("mppt_daya");

                        String bat_arus =
isi.getString("bat_arus");
```

```
                        String bat_tegangan =
isi.getString("bat_tegangan");
                        String bat_daya =
isi.getString("bat_daya");

                        isiPV.append("I: "+ pv_arus+"\n"
+
                                     "V: "+
pv_tegangan+"\n" +
                                         "P: "+
pv_daya+"\n");

                        isiMPPT.append("I: "+
mppt_arus+"\n" +
                                "V: "+ mppt_tegangan+"\n"
+
                                "P: "+ mppt_daya+"\n");

                        isiBaterai.append("I: "+
bat_arus+"\n" +
                                "V: "+ bat_tegangan+"\n"
+
                                "P: "+ bat_daya+"\n");

                    }
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        }, new Response.ErrorListener(){

            @Override
            public void onErrorResponse(VolleyError
volleyError) {
                VolleyLog.d("ERROR", "ERROR: ",
volleyError.getMessage());
            }
        });
        requestQueue.add(jsonObjectRequest);
    }

    private void showpDialog() {
        if (!pDialog.isShowing())
```

```
            pDialog.show();
    }

    private void hidepDialog() {
        if (pDialog.isShowing())
            pDialog.dismiss();
    }
}
```

Kode Sumber A.2 Menampilkan tampilan utama pada aplikasi

2. Berikut adalah potongan kode untuk fitur melihat hasil peramalan

```
<?php

       define('HOST','localhost');
       define('USER','cc_dbcharging');
       define('PASS','yanuar');
       define('DB','dbcharging');
       $conn = mysqli_connect(HOST,USER,PASS,DB) or
die('Unable to Connect');

       if ($_SERVER["REQUEST_METHOD"]=="POST") {
              movingAverage();
       }

       function movingAverage(){
              global $conn;

              $sql = "SELECT id, waktu, pv_arus,
pv_tegangan, mppt_arus, mppt_tegangan, mppt_daya,
bat_arus, bat_tegangan, bat_daya, round(pv_arus *
pv_tegangan,2) AS pv_asli FROM tbcharging_2 WHERE id <
26;";

              $hasil = mysqli_query($conn, $sql);
              $number_of_rows =
mysqli_num_rows($hasil);

              $data_asli = array();
              $data1 = array();

              if ($number_of_rows > 0) {
```

```
                    while ($row =
mysqli_fetch_assoc($hasil)) {
                            $data_asli[] = $row;
                            $data1[] = $row['pv_asli'];
                    }
            }

            $periode = 5;

            $jumlah1 = 0;
            //$jumlah2 = 0;
            //$jumlah3 = 0;
            for ($i=0; $i < $periode ; $i++) {
                    $jumlah1 = $jumlah1 + $data1[$i];
            }
            $data_f=array();

            $data_terakhir = count($data1);

            for ($i=$periode; $i < $data_terakhir ;
$i++) {
                    $data_f[] = array("id_f" => $i,
                            "pv_f" =>  round(($jumlah1 /
($periode)),2),
                            );

                    $jumlah1 = $jumlah1 - $data1[$i -
$periode] + $data1[$i];
            }
            echo
json_encode(array("tbcharging"=>$data_asli,
"tbcharging_f"=>$data_f));
      }
      mysqli_close($conn);
?>
```

Kode Sumber A.3 Mengambil data untuk hasil peramalan

```
public class Tab_2 extends Fragment {

    TableLayout tableMain;
    TableRow tableRow3;
    View v;

    String urlJson =
"http://chargingstation.its.ac.id/android_forecasting.php";
    RequestQueue requestQueue;

    public Tab_2() {
    }

    @Override
    public View onCreateView(LayoutInflater inflater,
ViewGroup container,
                             Bundle savedInstanceState) {
        // Inflate the layout for this fragment
        v = inflater.inflate(R.layout.fragment_tab_2,
container, false);

        requestQueue =
Volley.newRequestQueue(getActivity());
        tableMain = (TableLayout)
v.findViewById(R.id.tableMain);
        addDataForecasting();
        return v;
    }

    private void addDataForecasting(){

        int trWidthContent =
TableRow.LayoutParams.MATCH_PARENT;
        int trHeightContent =
TableRow.LayoutParams.WRAP_CONTENT;

        final TableRow.LayoutParams tr = new
TableRow.LayoutParams(trWidthContent, trHeightContent);
        tr.gravity = Gravity.CENTER | Gravity.BOTTOM;

        JsonObjectRequest jsonObjectRequest = new
JsonObjectRequest(Request.Method.POST, urlJson, null, new
Response.Listener<JSONObject>() {
```

```
            @Override
            public void onResponse(JSONObject jsonObject) {
                try {
                    JSONArray tbcharging =
jsonObject.getJSONArray("tbcharging");
                    JSONArray tbcharging_f =
jsonObject.getJSONArray("tbcharging_f");
                    for (int i = 0;
i<tbcharging.length();i++){
                        JSONObject isi =
tbcharging.getJSONObject(i);

                        String id = isi.getString("id");
                        String pv_asli =
isi.getString("pv_asli");

                        tableRow3 = new
TableRow(getActivity());
                        tableRow3.setId(i);
                        tableRow3.setLayoutParams(tr);

                        TextView txtView3a = new
TextView(getActivity());
                        txtView3a.setId(i);
                        txtView3a.setText(id);
                        txtView3a.setLayoutParams(tr);
                        tableRow3.addView(txtView3a);

                        TextView txtView3b = new
TextView(getActivity());
                        txtView3b.setId(i);
                        txtView3b.setText(pv_asli);
                        txtView3b.setLayoutParams(tr);
                        tableRow3.addView(txtView3b);

                        tableMain.addView(tableRow3);
                    }
                    for (int i = 0;
i<tbcharging_f.length();i++){
                        JSONObject isi =
tbcharging_f.getJSONObject(i);

                        String id_f =
```

```
isi.getString("id_f");
                            String pv_f =
isi.getString("pv_f");

                            tableRow3 = new
TableRow(getActivity());
                            tableRow3.setId(i);
                            tableRow3.setLayoutParams(tr);

                            TextView txtView3c = new
TextView(getActivity());
                            txtView3c.setId(i);
                            txtView3c.setText(pv_f);
                            txtView3c.setLayoutParams(tr);
                            tableRow3.addView(txtView3c);

                            tableMain.addView(tableRow3);
                        }

                    } catch (JSONException e) {
                        e.printStackTrace();
                    }
                }
            }, new Response.ErrorListener() {
                @Override
                public void onErrorResponse(VolleyError
volleyError) {
                    VolleyLog.d("ERROR", "ERROR: ",
volleyError.getMessage());
                }
            });
            requestQueue.add(jsonObjectRequest);
        }
}
```

Kode Sumber A.4 Menampilkan hasil peramalan di aplikasi

3. Berikut adalah potongan kode untuk fitur melihat isi database

```
<?php

        define('HOST','localhost');
        define('USER','cc_dbcharging');
        define('PASS','yanuar');
        define('DB','dbcharging');
        $conn = mysqli_connect(HOST,USER,PASS,DB) or
die('Unable to Connect');

        if ($_SERVER["REQUEST_METHOD"]=="POST") {
                showData();
        }

        function showData(){

                global $conn;

                $query = "SELECT id, waktu, pv_arus,
pv_tegangan, mppt_arus, mppt_tegangan, mppt_daya,
bat_arus, bat_tegangan, bat_daya, round(pv_arus *
pv_tegangan,2) AS pv_daya  FROM tbcharging_2 WHERE id <
26;";

                $result = mysqli_query($conn,
$query);
                $number_of_rows =
mysqli_num_rows($result);

                $temp_array = array();

                if ($number_of_rows > 0) {
                        while ($row =
mysqli_fetch_assoc($result)) {
                                $temp_array[] =
$row;
                        }
                }
                echo
json_encode(array("tbcharging"=>$temp_array));
```

```
                mysqli_close($conn);
        }
?>
```

Kode Sumber A.5 Mengambil data untuk isi database

```
public class Tab_3 extends Fragment {

    TableLayout tableMain;
    View v;
    TableRow tableRow3;

    String urlJson =
"http://chargingstation.its.ac.id/android_database.php";
    RequestQueue requestQueue;

    public Tab_3() {

    }

    @Override
    public View onCreateView(LayoutInflater inflater,
ViewGroup container,
                             Bundle savedInstanceState)
{
        // Inflate the layout for this fragment
        v = inflater.inflate(R.layout.fragment_tab_3,
container, false);


        requestQueue =
Volley.newRequestQueue(getActivity());
        tableMain = (TableLayout)
v.findViewById(R.id.tableMain);
        addDataTable();
        return v;
    }

    private void addDataTable(){

        int trWidthContent =
TableRow.LayoutParams.MATCH_PARENT;
        int trHeightContent =
```

```
TableRow.LayoutParams.MATCH_PARENT;

        final TableRow.LayoutParams tr = new
TableRow.LayoutParams(trWidthContent, trHeightContent);
        tr.gravity = Gravity.CENTER;

        JsonObjectRequest jsonObjectRequest = new
JsonObjectRequest(Request.Method.POST, urlJson, null,
new Response.Listener<JSONObject>() {
            @Override
            public void onResponse(JSONObject
jsonObject) {
                try {
                    JSONArray tbcharging =
jsonObject.getJSONArray("tbcharging");
                    for (int i = 0; i
<tbcharging.length(); i++){
                        JSONObject isi =
tbcharging.getJSONObject(i);

                        String id = isi.getString("id");
                        String pv_arus =
isi.getString("pv_arus");
                        String pv_tegangan =
isi.getString("pv_tegangan");
                        String pv_daya =
isi.getString("pv_daya");

                        String mppt_arus =
isi.getString("mppt_arus");
                        String mppt_tegangan =
isi.getString("mppt_tegangan");
                        String mppt_daya =
isi.getString("mppt_daya");

                        String bat_arus =
isi.getString("bat_arus");
                        String bat_tegangan =
isi.getString("bat_tegangan");
                        String bat_daya =
isi.getString("bat_daya");

                        tableRow3 = new
```

```
TableRow(getActivity());
                        tableRow3.setId(i);
                        tableRow3.setLayoutParams(tr);

                        TextView txtView3a = new
TextView(getActivity());
                        txtView3a.setId(i);
                        txtView3a.setText(id);
                        txtView3a.setLayoutParams(tr);
                        tableRow3.addView(txtView3a);

                        TextView txtView3b = new
TextView(getActivity());
                        txtView3b.setId(i);
                        txtView3b.setText(pv_arus);
                        txtView3b.setLayoutParams(tr);
                        tableRow3.addView(txtView3b);

                        TextView txtView3c = new
TextView(getActivity());
                        txtView3c.setId(i);
                        txtView3c.setText(pv_tegangan);
                        txtView3c.setLayoutParams(tr);
                        tableRow3.addView(txtView3c);

                        TextView txtView3d = new
TextView(getActivity());
                        txtView3d.setId(i);
                        txtView3d.setText(pv_daya);
                        txtView3d.setLayoutParams(tr);
                        tableRow3.addView(txtView3d);

                        TextView txtView3e = new
TextView(getActivity());
                        txtView3e.setId(i);
                        txtView3e.setText(mppt_arus);
                        txtView3e.setLayoutParams(tr);
                        tableRow3.addView(txtView3e);

                        TextView txtView3f = new
TextView(getActivity());
                        txtView3f.setId(i);
```

```
txtView3f.setText(mppt_tegangan);
                        txtView3f.setLayoutParams(tr);
                        tableRow3.addView(txtView3f);

                        TextView txtView3g = new
TextView(getActivity());
                        txtView3g.setId(i);
                        txtView3g.setText(mppt_daya);
                        txtView3g.setLayoutParams(tr);
                        tableRow3.addView(txtView3g);

                        TextView txtView3h = new
TextView(getActivity());
                        txtView3h.setId(i);
                        txtView3h.setText(bat_arus);
                        txtView3h.setLayoutParams(tr);
                        tableRow3.addView(txtView3h);

                        TextView txtView3i = new
TextView(getActivity());
                        txtView3i.setId(i);
                        txtView3i.setText(bat_tegangan);
                        txtView3i.setLayoutParams(tr);
                        tableRow3.addView(txtView3i);

                        TextView txtView3j = new
TextView(getActivity());
                        txtView3j.setId(i);
                        txtView3j.setText(bat_daya);
                        txtView3j.setLayoutParams(tr);
                        tableRow3.addView(txtView3j);

                        tableMain.addView(tableRow3);
                    }
                } catch (JSONException e) {
                    e.printStackTrace();
                }
            }
        }, new Response.ErrorListener(){
            @Override
            public void onErrorResponse(VolleyError
volleyError) {
                VolleyLog.d("ERROR", "ERROR: ",
```

```
volleyError.getMessage());
            }
        });
        requestQueue.add(jsonObjectRequest);
    }
}
```

Kode Sumber A.6 Menampilkan isi database pada aplikasi

[Halaman ini sengaja dikosongkan]

BAB VI
KESIMPULAN DAN SARAN

6.1 Kesimpulan

Dari hasil pengamatan dan percobaan selama perancangan, implementasi, dan uji coba aplikasi, maka dapat diambil kesimpulan sebagai berikut:

1. Berdasarkan uji coba fungsionalitas, aplikasi sistem *monitoring charging station* sudah dapat menampilkan masing - masing kebutuhan fungsional.
2. Peramalan dengan metode *moving average* sangat ditentukan oleh bilangan periode *moving average* (*m*) dan tingkat konsistensi data.
3. Dengan adanya aplikasi sistem *monitoring charging station* berbasis perangkat bergerak, pemantauan kondisi daya dapat dilakukan dimana saja dan kapan saja.

6.2 Saran

Saran yang diberikan untuk pengembangan aplikasi ini adalah:

1. Untuk lebih mempermudah pengguna untuk membaca hasil peramalan, sebaiknya digunakan grafik.
2. Memperbanyak data pengujian akan membuat tingkat keakuratan metode prediksi ini meningkat.

[Halaman ini sengaja dikosongkan]

DAFTAR PUSTAKA

[1] Hydro-Quebec, Electrical Vehicle Charging Stations Technical Installation Guide, Montreal, Canada: Hydro-Quebec, 2015.

[2] M. Margelis, Arduino Cookbook, USA: O'Reilly Media, Inc, 2011.

[3] LEM, Current Tranducers LA 55 - P, LEM, 2014.

[4] Interplus Industry Co. Ltd., ZMPT101B, Shen Zhen, China: Interplus Industry Co. Ltd..

[5] WIZnet Co., Inc., W5100 Datasheet, Gyeonggi: WIZnet Co., Inc., 2008.

[6] Editorial Team, "Reading and Writing Files from an SD Card with an Arduino," EETech Media, LLC, [Online]. Available: http://www.allaboutcircuits.com/projects/reading-and-writing-files-from-an-sd-card-with-an-arduino/.

[7] R. Meier, Professional Android Application Development, Wiley Publishing, Inc., 2009.

[8] R. J. H. George Athanasopoulous, Forecasting: Principles and Practice, Australia: University of Western, 2014.

[9] W. Enders, Applied Economic Time Series Second Edition, New Jersey: Willey, 2004.

[Halaman ini sengaja dikosongkan]

BIODATA PENULIS

Sanindya Lesario, lahir di Jakarta pada 6 Agustus 1994. Penulis menempuh pendidikan mulai dari TK Islam Baitussalam (1998 - 2000). SD Negeri 010 Percontohan Pondok Kelapa (2000 - 2006), SMPN 109 Jakarta Timur (2006 - 2009), SMAN 61 Jakarta Timur (2009 - 2012) dan S1 Teknik Informatika ITS (2012 - 2016).Selama masa kuliah, penulis aktif dalam organisasi Himpunan Mahasiswa Teknik Computer (HMTC). Diantaranya adalah menjadi Staf Departemen Dalam Negeri HMTC 2013– 2014, menjadi panitia acara tahunan jurusan teknik informatika (SCHEMATICS) 2014-2015.Selama kuliah di teknik informatika ITS, penulis mengambil rumpun mata kuliah Komputasi Berbasis Jaringan (KBJ). Komunikasi dengan penulis dapat melalui email :**lesario12@mhs.if.its.ac.id**