PERANCANGAN *MIDDLEWARE* UNTUK APLIKASI *SMART GRID*

Reza Azman Akbari[1], Susmini Indriani Lestariningati[2]
[1,2] Program Studi Sistem Komputer Unikom, Bandung
[1]reza.azmanakbari@gmail.com, [2]lestariningati@gmail.com

ABSTRAK

Teknologi Smart Grid harus memiliki keandalan dan informasi yang real-time antara gardu listrik dan operator. Terlebih lagi seorang operator membutuhkan fleksibilitas dan mobilitas dari setiap aktivitas pada gardu listrik. Teknologi Smart Grid dapat berjalan maksimal dengan dukungan middleware. Perancangan middleware untuk aplikasi Smart Grid, yakni yang mampu menjadi sebuah jembatan antara beberapa perangkat heterogen dapat menjadi solusinya. Middleware yang dirancang haruslah memiliki data real-time, dapat diakses dengan perangkat heterogen, menyediakan data yang memiliki format umum, hingga mampu menyediakan antarmuka pemantauan gardu listrik. Middleware yang dibangun untuk aplikasi Smart Grid memanfaatkan web server, web services, serta basis data. Berdasarkan hasil pengujian yang telah dilakukan dari perancangan middleware untuk aplikasi Smart Grid bahwa middleware yang dibangun mampu berjalan dengan sesuai prosedur dan berhasil menjalankan fungsi-fungsi yang diinginkan berdasarkan perancangan, akan tetapi perancangan middleware masih memiliki kekurangan sumber gardu yang tersedia untuk dilakukan pengolahan data.

Kata kunci : Smart Grid, Middleware, Web Services, Web Server, Basis Data.

1. PENDAHULUAN

Kebutuhan pokok saat ini bukan sekadar sandang, pangan, dan papan saja, namun energi listrik juga sudah termasuk sebagai kebutuhan pokok. Oleh karena itu, energi listrik harus dapat tercapai setiap waktu kepada konsumen ketika dibutuhkan. Guna dapat mendukung penggunaan energi listrik kepada konsumen, maka teknologi *Smart Grid* bisa menjadi salah satu solusinya. *Smart Grid* adalah jaringan listrik pintar yang memanfaatkan teknologi digital dan teknologi canggih lainnya untuk memantau dan mengelola pendistribusian listrik. Keandalan dan informasi yang *real-time* antara gardu listrik dan operator merupakan kunci utama agar proses pemantauan berjalan sesuai prosedur serta mampu menganalisis gangguan pada gardu listrik. Maka dari itu, seorang operator haruslah memiliki fleksibilitas dan mobilitas dari setiap aktivitas pada gardu listrik tersebut.

Teknologi *Smart Grid* tidak akan berjalan dengan baik bila tidak didukung oleh sebuah *middleware*. *Middleware* menggabungkan teknologi komunikasi nirkabel dan pengolahan sinyal multimedia. Bisa dikatakan bahwa *middleware* memiliki peran kepada operator agar dapat terhubung dengan perangkat heterogen dengan berbagai protokol komunikasi[1]. Tidak hanya itu saja, operator pun dapat menyimpan data informasi gardu listrik, memantau secara *real-time*, dan menerima informasi keadaan gardu listrik kapan saja dan di mana saja.

Perancangan *middleware* ditujukan untuk operator pada sebuah perusahaan pendistribusian listrik agar lebih memiliki fleksibilitas dan mobilitas dalam melakukan pemantauan gardu listrik. Sistem kerjanya adalah informasi data gardu yang ditransmisikan akan diakuisisi dan dilakukan pemisahan data, lalu *middleware* akan menyediakan data untuk perangkat heterogen baik untuk dikembangkan ke perangkat heterogen ataupun diakses oleh perangkat heterogen. Dengan adanya pembuatan *middleware* untuk aplikasi sistem *Smart Grid,* diharapkan dapat mengintegrasikan komunikasi untuk perangkat heterogen pada *Smart Grid* dan menciptakan kemudahan bagi operator untuk memantau gardu listrik sehingga masalah-masalah yang sering terjadi pada gardu listrik dapat diketahui lebih cepat.

2. TEORI PENUNJANG

Pembahasan berisi tentang *Smart Grid, middleware,* komunikasi data, *web services*, *HyperText Transfer Protocol, Uniform Resource Identifier*, *Dynamic Domain Name System, JavaScript Object Notation* dan *Wireshark*.

2.1 *Smart Grid*

Smart Grid adalah suatu jaringan listrik yang menggunakan teknologi digital dan

teknologi maju lainnya untuk memantau dan mengelola transportasi listrik dari sumber pembangkitan listrik untuk memenuhi perubahan kebutuhan listrik dari pelanggan[2]. Suatu *Smart Grid* secara cerdas mengintegrasikan kegiatan semua *user* dalam rangka memberikan atau menghasilkan suplai listrik secara efisien, berkesinambungan, ekonomis dan aman.

Smart Grid memakai produk inovatif dan bersama-sama melayani dengan pemantauan, pengendalian, komunikasi dan *self-healing technologies* untuk:

a. Memfasilitasi lebih baik hubungan dan operasi dari semua generator dan teknologi.
b. Memberikan keleluasaan kepada pelanggan untuk menentukan bagian dari optimasi operasi dari sistem.
c. Menyediakan pelanggan dengan informasi dan pilihan untuk suplai.
d. Secara signifikan mengurangi dampak lingkungan dari sistem suplai listrik seluruhnya.
e. Meningkatkan kehandalan dan keamanan suplai.

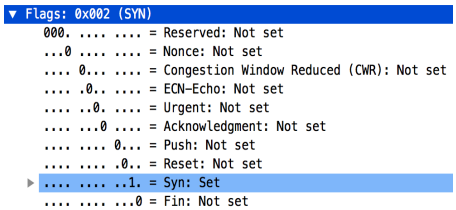
2.2 *Middleware*

Guna mengintegrasikan semua perangkat yang berbeda dengan menyediakan layanan komunikasi data, oleh karena itu, diperlukan lapisan terpadu. Lapisan ini tidak hanya memadukan semua kompleksitas dari komponen-komponen *Smart Grid* yang berfungsi untuk pengukuran cerdas, tetapi juga struktur jaringan listrik, lalu kemudian menjaga *user* dari struktur jaringan tersebut. Masalah pada lapisan harus dikembangkan, karena *middleware* dapat mengintegrasikan komponen dan perangkat yang berbeda pada *Smart Grid* sehingga bisa menjadi satu lapisan heterogen. Seperti kasus konversi nilai mata uang pada transaksi pasar uang *online* yang memiliki kemungkinan terjadinya kesalahan konversi bila tanpa adanya lapisan *middleware* atau timbulnya masalah interpretasi (sebagai contoh, bila informasi diterima oleh perangkat atau peralatan dengan prosesor yang berbeda atau perbedaan metode penyimpanan *byte*). Masalah interpretasi mampu ditangani bila adanya lapisan *middleware*. Bukan hanya itu saja, permasalahan mengenai format yang berbeda pun dapat teratasi oleh *middleware*.

2.3 Protokol TCP/IP

Protokol TCP/IP dikembangkan sebelum adanya model OSI. Oleh karena itu, lapisan dalam TCP/IP berbeda dengan model OSI. Protokol TCP/IP memiliki empat lapisan, yaitu *host-to-network*, *internet, transport,* dan *application*. Namun, ketika TCP/IP dibandingkan dengan OSI, bisa dikatakan bahwa lapisan *host-to-network* sama dengan kombinasi dari *physical layer* dan *data link layer*. Lapisan *internet* sama dengan *network layer*, dan lapisan *application* bisa disamakan dengan *session layer*, *presentation layer*, dan *application layer*, tetapi lapisan *transport* pada TCP/IP mengurus bagian dari tugas *session layer*. Jadi bisa diasumsikan bahwa protokol TCP/IP terbuat dari lima lapisan, yaitu *physical*, *data link*, *network*, *transport*, dan *application*. Empat lapisan pertama mengurus standar perangkat fisik, antar muka jaringan, *internetworking*, dan melakukan pengiriman yang sesuai dengan empat lapisan pertama dari model OSI. Tiga lapisan paling luar dari model OSI bila dalam protokol TCP/IP dapat terwakilkan oleh lapisan *application*.

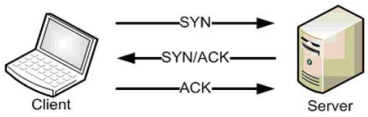
Protokol TCP merupakan protokol yang beroirientasikan koneksi. Berorientasi koneksi pada TCP maksudnya adalah setiap *client-server* yang ingin melakukan komunikasi data wajib melakukan suatu persetujuan komunikasi atau biasanya sering disebut dengan *three-way handshake*. *Three-way handshake* sendiri merupakan 3 tahapan *client* dan *server* yang ingin melakukan komunikasi data dengan mengirimkan *flag* TCP dari *client* dan ke *server*. *Flag* TCP dapat dilihat pada Gambar 2.1.



Gambar 2.1 Contoh Flag Pada Wiresharsk.

Untuk melakukan *handshake*, *flag* yang dibutuhkan adalah *flag* SYN dan ACK. Berikut adalah *three-way handshake*:

1. *Client* mengirimkan *flag* SYN ke *server*.
2. *Server* mengimkan balasan *flg* SYN dan ACK ke *client*.
3. *Client* mengirimkan balasan ACK ke *server*.



Gambar 2.2 Three-Way Handshake Protokol TCP.

2.4 *Web Services*

Penggunaan *web services* pada *World Wide Web* (www) berkembang dengan pesat seiring dengan kebutuhan komunikasi antar aplikasi dan meningkatnya interoperabilitas. Layanan ini menyediakan sarana komunikasi standar di antara berbagai aplikasi perangkat lunak yang memberikan informasi secara dinamis kepada *user*[3].

Web services adalah sebuah sistem perangkat lunak yang didesain untuk mendukung interaksi yang interoperabilitas antar mesin melalui sebuah jaringan. *Web services* dibangun di atas protokol-protokol yang sudah terkenal dan memiliki *platform* yang *independen*, seperti HTTP, XML (*Extensible Markup Language*), UDDI (*Universal Discovery Description Integration*), dan WSDL (*Web Services Description Language*)[4].

2.5 *Web Server*

Web server merupakan perangkat lunak yang memberikan layanan data, berfungsi menerima permintaan HTTP atau HTTPS (*HyperText Transfer Protocol Secure*) dari *client* yang dikenal dengan *web browser* dan mengirimkan kembali hasilnya dalam bentuk halaman-halaman *web* yang umumnya berbentuk dokumen HTML.

2.6 Basis Data

Basis Data Oracle 11g adalah salah satu produk Oracle Corporation yang sampai saat ini telah dipercaya sebagai perangkat lunak *Relational Database Management System* (RDBMS) dengan bahasa pemrograman *Structured Query Language* (SQL) dan *Prosedural Language*/*Structured Query Language* (PL/SQL). RDBMS sendiri merupakan sistem yang dapat memindahkan data ke dalam basis data, menyimpan data, menerima data, sehingga basis data dapat dimanipulasi dengan berbagai macam aplikasi.

2.7 *HyperText Transfer Protocol*

Protokol HTTP adalah protokol yang menganut *request/response*. *User* mengirimkan sebuah *request* (permintaan) ke dalam *server* dalam bentuk *request method* (metode permintaan), URI (*Uniform Resurce Identifier*), dan versi protokol.

Dalam HTTP, selain berguna sebagai jembatan komunikasi antara beberapa perangkat, HTTP dapat digunakan untuk melakukan penerimaan dokumen dan perbaruan dokumen melalui URI dengan menggunakan *request methods*, yaitu *GET method* dan *POST method*. Kedua *methods* tersebut merupakan sebuah layanan yang sudah diberikan oleh HTTP.

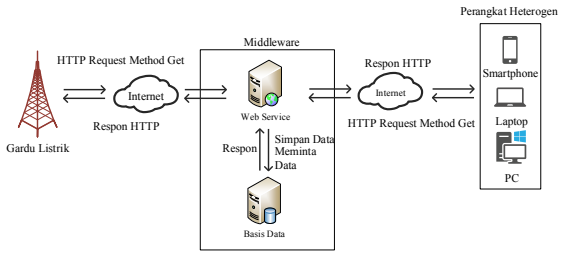
2.8 *Uniform Resource Identifier*

Uniform Resource Identifier (URI) merupakan sebuah identifikasi yang terdiri dari urutan karakter yang tepat dengan aturan tertentu yang digunakan untuk mengakses suatu *resource* (sumber) melalui sebuah jaringan dengan menggunakan protokol tertentu. URI merepresentasikan entitas dari suatu *resource* yang memiliki format protokol://host/path atau contohnya adalah seperti http://facebook.com/login. HTTP menunjukkan protokol yang digunakan, facebook.com menunjukkan host dari *server*, dan login menunjukkan lokasi berkas yang diakses.

Selain itu, dalam formatnya, URI memiliki yang namanya *query*. *Query* digunakan untuk mengirimkan informasi dari *server* ke *client* atau sebaliknya. *Query* yang umum digunakan ketika melakukan autentikasi pada halaman *login*. Ketika *user* melakukan *input* informasi dari *client*, maka yang dikirimkan ke *server* adalah *query* dari URI. Format dari *query* dimulai dengan menggunakan tanda "?", dan bisa dilihat seperti http://facebook.com/login?username=xxx&password=xxx. Bila dilihat dari contoh, *query* mengirimkan informasi *username* dan *password* ke *server*. Oleh karena itu, *query* bisa digunakan untuk mengirimkan suatu informasi dari *server* ke *client* atau sebaliknya.

3. PERANCANGAN SISTEM

Langkah awal dari perancangan sistem adalah analisis dan penentuan kebutuhan sistem. Pada langkah ini menentukan kebutuhan apa saja yang harus terpenuhi.



Gambar 3.1 Kinerja Middleware Pada Gambaran Umum Sistem.

3.1 Komponen Sistem *Middleware*

Untuk perancangan *middleware* untuk aplikasi *Smart Grid* dibutuhkan beberapa perangkat lunak dan perangkat keras pada *web service* dan basis data yang digunakan, yaitu sebagai berikut:

a. Satu perangkat komputer dengan RAM 8 GB dan Prosesor AMD Phenom II X4 965 3,5 GHz.
b. Sistem Operasi Windows 10 Pro 64-*bit.*
c. XAMPP 1.8.2 digunakan sebagai *web services* dari layanan PHP dan sebagai *web server* dari layanan Apache di dalamnya.
d. Basis Data Oracle 11g 64-*bit*.
e. SQL Developer.
f. Sublime Text digunakan sebagai *text editor*.
g. Satu buah layanan *Dynamic Domain Name. Server* dari https://ip-dynamic.com.

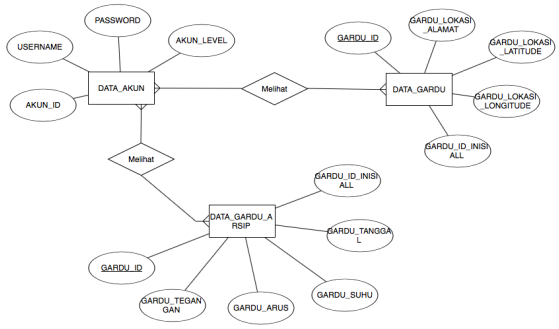
3.2 Proses Sistem *Middleware*

Sistem ini akan menerima data yang dikirimkan melalui protokol HTTP oleh perangkat pada gardu dan memberikan data pada *end device* yang ingin mengakses dan mengolah data gardu yang dikirimkan, berikut adalah langkah-langkahnya:

1. Perangkat pada gardu diberikan tanda pengenal agar bisa teridentifkasi pada basis data dan sistem. Pada sistem ini perangkat gardu diberikan tanda pengenal yaitu berupa "ID".
2. Perangkat pada gardu mengakses *web services* melalui protokol HTTP dengan URI yang telah disediakan.
3. Perangkat pada gardu mengirimkan beberapa *query* pada URI dengan format yang telah ditentukan.
4. *Web services* akan memisahkan data gardu yang dikirimkan berdasarkan tanda pengenal gardu ke dalam basis data.
5. Bila perangkat pada gardu berhasil mengirimkan data gardu, maka perangkat pada gardu akan menerima respon dari *web services*
6. Data gardu yang telah berhasil disimpan ke dalam basis data diolah *web services* ke dalam format JSON.
7. Data gardu yang memiliki format JSON dapat dibaca oleh *end device* yang ingin mengolah data gardu melalui *web services* dengan protokol HTTP dan dengan halaman yang telah disediakan.

3.3 Konfigurasi Basis Data

Pada Gambar 3.2 merupakan keterangan diagram hubungan antar entitas dan atribut pada perancangan *middleware* untuk aplikasi *Smart Grid*:

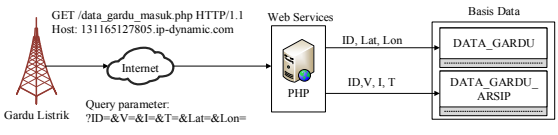


Gambar 3.2 Entity Relationship Diagram Sistem.

3.4 Konfigurasi *Web Services*

Setelah melakukan instalasi pada perangkat komputer yang digunakan, tahap selanjutnya adalah melakukan konfigurasi untuk *web services*. Konfigurasi yang harus dilakukan adalah sebagai berikut:

1. Konfigurasi Akuisisi Data Gardu ke Basis Data.

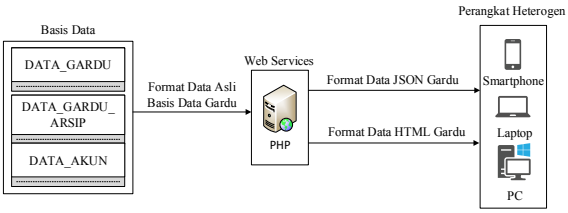


Gambar 3.3 Konsep Akuisisi Data dan Pemisahan Data Ke Basis Data.

Perangkat gardu yang terhubung yang akan mengirimkan data gardu harus memiliki tanda "ID" pada setiap pengiriman. Dari setiap tanda yang ditransmisikan, akan dipisahkan sesuai dengan basis datanya masing-masing. Perangkat gardu tersebut mengirimkan data gardu melalui *query* "?ID=", "&V=", "&I=", "&T=", "&Lat=", "&Lon=", "&user=", "&pass=". *Query* tersebut mewakili data gardu sebagai berikut:

a. "ID" untuk tanda pengenal gardu.
b. "V" untuk data tegangan gardu.
c. "I" untuk data arus gardu.
d. "T" untuk data suhu gardu.
e. "Lat" untuk data *latitude* koordinat lokasi gardu.
f. "Lon" untuk data *longitude* koordinat lokasi gardu.
g. "user" untuk autentikasi pengiriman gardu.
h. "pass" untuk autentikasi pengiriman gardu.

2. Konfigurasi Mengubah Format Basis Data Gardu ke Format JSON.

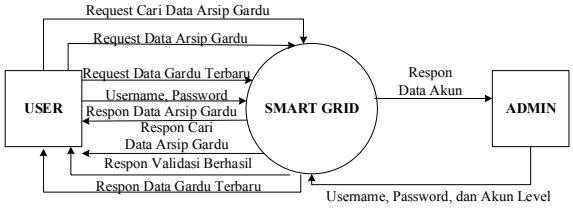


Gambar 3.4 Konsep Ketersediaan Data Untuk Perangkat Heterogen.

Konfigurasi ini dilakukan agar perangkat heterogen seperti perangkat Android yang ingin menggunakan basis data gardu dapat melakukan pengembangan basis data gardu ke dalam aplikasi Android. Kode PHP yang akan dibuat diberi nama berkas "json_arsip.php", "json_last.php", dan "json_akun". Berkas-berkas tersebut disimpan di /xampp/htdocs/ sehingga data JSON dapat diakses melalui protokol HTTP.

3.5 Perancangan Sistem Antarmuka Data Gardu

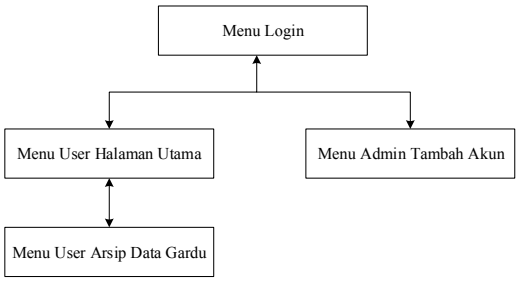
Perancangan sistem antarmuka bertujuan agar data yang terakuisisi oleh sistem *middleware* dapat diakses atau dipantau dengan mudah oleh *user*. Pada perancangan antarmuka akan tersedia fitur *login*, akses data gardu, akses data gardu terbaru, akses pencarian data gardu, dan penambahan akun baru.



Gambar 3.5 Diagram Konteks Sistem.

3.6 Arsitektur Menu Antarmuka Data Gardu

Struktur menu yang memaparkan mengenai menu yang akan dibangun merupakan suatu kebutuhan dalam melakukan perancangan sebuah perangkat lunak. Berikut adalah struktur menu dari perancangan antarmuka data gardu:



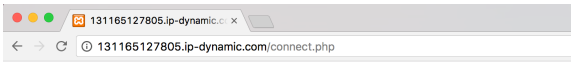
Gambar 3.6 Struktur Menu Antarmuka Data Gardu.

4. PENGUJIAN DAN ANALISIS

Pengujian dilakukan agar mengetahui bagaimana kinerja sistem yang telah dibangun. Berikut adalah beberapa pengujian yang dilakukan untuk perancangan *middleware* untuk aplikasi *Smart Grid*.

4.1 Pengujian Akses *Middleware*

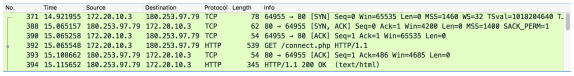
Pengujian ini dilakukan dengan cara melakukan *request* ke halaman *web middleware* dan melihat koneksi yang terjadi dengan bantuan *wireshark*. Berikut adalah hasil pengujian akses ke *middleware*.



Gambar 4.1 Salah Satu Hasil Respon Akses ke Middleware.

Pada Gambar 4.1 menunjukkan bahwa *middleware* yang dibangun dengan DDNS (*Dynamic Domain Name System*) dan XAMPP dapat diakses melalui jaringan *Internet*. Bila diteliti lebih lanjut dari Gambar 4.2, *user* yang ingin mengakses *middleware* telah berhasil

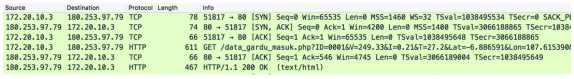
melakukan *three-way handshake* koneksi protokol TCP dan telah berhasil melakukan pengiriman *request* dan sistem memberikan respon melalui protokol HTTP.



Gambar 4.2 Filter Koneksi Protokol TCP dan HTTP Akses Middleware.

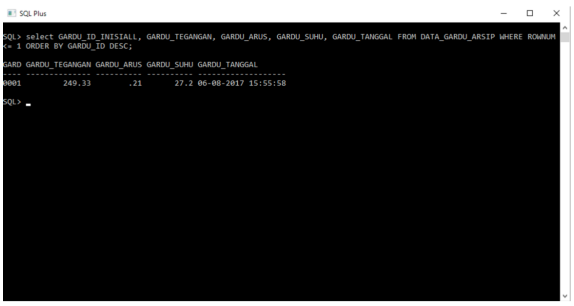
4.2 Pengujian Pemisahan Data Gardu Berdasarkan *Query*

Pengujian ini dilakukan dengan cara mengakuisi data yang ditransmisikan oleh perangkat gardu ke dalam *middleware* berdasarkan format *query* yang telah ditentukan. *Query* yang dikenali oleh *middleware* yaitu "ID", "V", "I", "T", "Lat" dan "Lon". Data yang terakuisisi selanjutnya dipisahkan berdasarkan *query*-nya ke dalam basis data yang dituju.

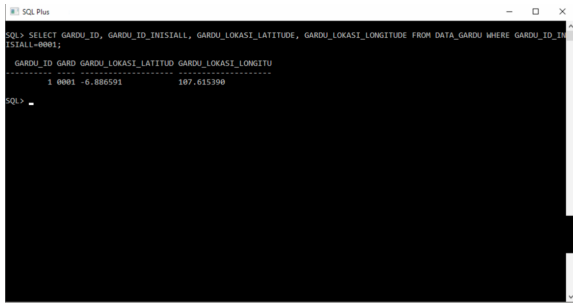


Gambar 4.3 Filter Koneksi Protokol HTTP dan TCP Akuisisi Data Gardu.

Pada Gambar 4.3 dapat dilihat bahwa perangkat gardu melakukan *handshake* dengan *middleware* dan melakukan pengiriman data yang berupa *query*, sedangkan *middleware* menerima *handshake* perangkat gardu dan mengakuisisi *query* tersebut. Perangkat gardu yang melakukan *handshake* dapat diketahui berdasarkan 3 protokol TCP pertama pada Gambar 4.3. Setelah itu *user* mengirimkan *query* "?ID=0001&V=249.33&I=0.21&T=27.2&Lat=-6.886&Lon=107.615" melalui protokol HTTP dengan *request method* GET. Kemudian *middleware* mengirimkan respon "HTTP/1.1 200 OK" yang berarti data gardu telah berhasil diakuisisi atau pentransmisian data gardu dapat diterima.



Gambar 4.4 Hasil Akuisisi di Basis Data Gardu Arsip.

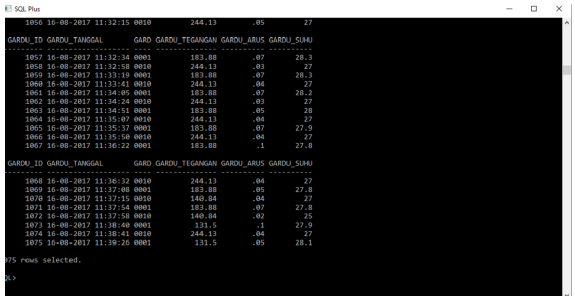


Gambar 4.5 Hasil Akuisisi di Basis Data Gardu.

Pada Gambar 4.4 dan Gambar 4.5, *middleware* mampu mengakuisisi data gardu yang ditransmisikan dan mampu untuk memisahkan data yang terakuisisi berdasarkan basis data yang dituju. *Query* yang ditransmisikan oleh perangkat gardu yaitu memiliki format seperti "?ID=&V=&I=&T=&Lat=&Lon=". *Query* tersebut mewakili data gardu yang ditransmisikan. Oleh sebab itu, *query* "ID", "V", "I", dan "T" mewakili data inisial gardu, tegangan, arus, dan suhu yang berada pada basis data gardu arsip pada Gambar 4.4, sedangkan *query* "Lat" dan "Lon", mewakili data lokasi *latitude* dan lokasi *longitude* yang berada pada tabel basis data gardu pada Gambar 4.5.

4.3 Pengujian Pengarsipan Data Gardu

Pengujian ini dilakukan dengan cara mengkuisisi data gardu yang dikirimkan sebagai *query,* kemudian data tersebut disimpan dan diarsipkan ke dalam basis data yang tersedia.

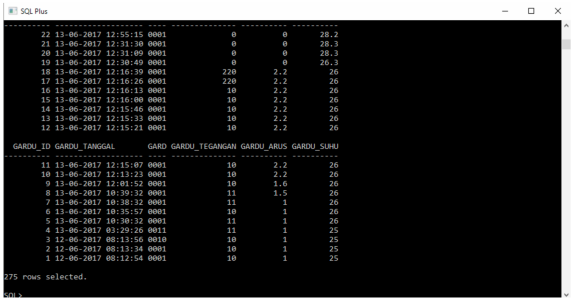


Gambar 4.6. Hasil Arsip Data Gardu.

Pada Gambar 4.6 merupakan hasil dari penyimpanan data gardu yang ditransmisikan oleh perangkat gardu berdasarkan hasil *query* yang terakuisisi. Setiap satu paket *query* yang berisi data gardu yang ditransmisikan ke *middleware*, selanjutnya *query* tersebut disimpan berurutan secara terus-menerus hingga menjadi sebuah arsip data gardu yang nantinya dapat dimanfaatkan untuk melakukan analisis data gardu tersebut.

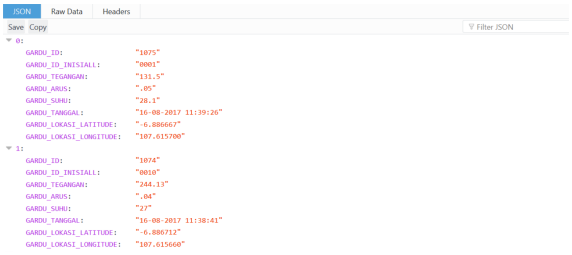
4.4 Pengujian Ketersediaan JSON Untuk Perangkat Heterogen

Pengujian ini dilakukan dengan cara mengakuisisi data gardu yang dikirimkan sebagai *query*, lalu data gardu yang tersimpan di basis data selanjutnya di konversi ke format JSON. Basis data yang dikonversi berdasarkan data gardu terbaru, data arsip gardu, dan data akun.

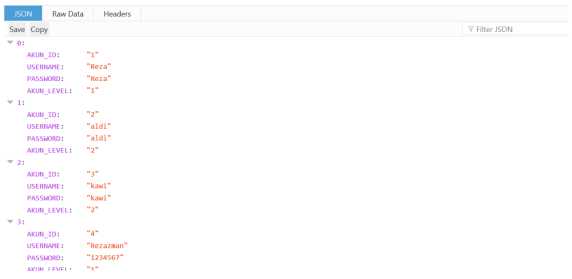


Gambar 4.7. Contoh Format Basis Data.

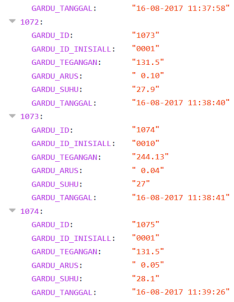
Pada Gambar 4.7 merupakan data yang terakuisisi oleh ke dalam basis data, kemudian data tersebut akan diolah ke dalam format JSON.



Gambar 4.8 Hasil Konversi Format JSON Data Gardu Terbaru.



Gambar 4.9 Hasil Konversi Format JSON Data Akun.

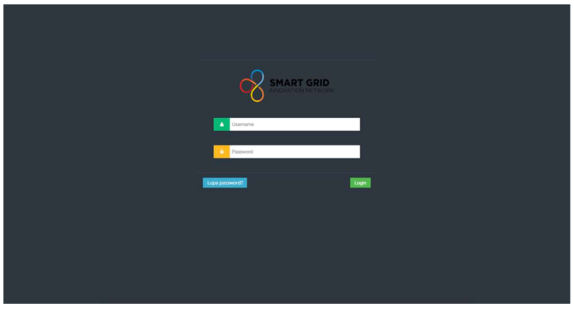


Gambar 4.10 Hasil Konversi Format JSON Arsip Data Gardu.

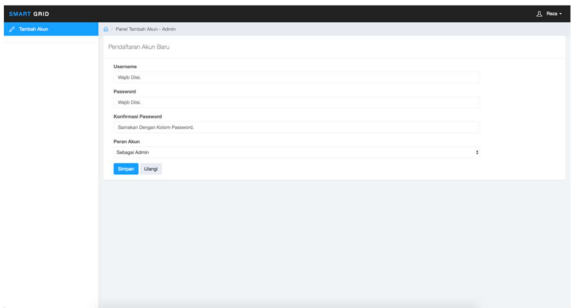
Pada Gambar 4.8, Gambar 4.9, dan Gambar 4.10 merupakan hasil konversi format JSON dari basis data *middleware* seperti pada Gambar 4.7. Format JSON dari data gardu ini berfungsi sebagai data yang memiliki fleksibilitas, mampu diolah oleh perangkat heterogen yang memiliki bahasa pemrograman yang berbeda-beda.

4.5 Pengujian Akses Data Gardu Untuk Aplikasi Berbasis *Web*

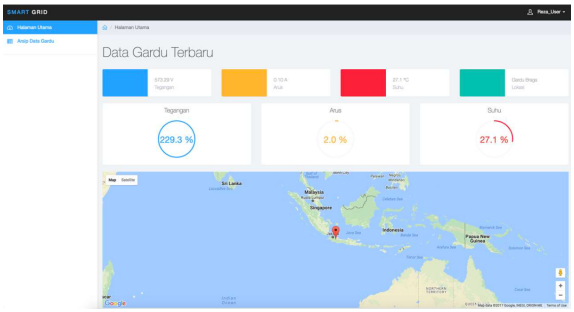
Pengujian ini dilakukan dengan cara mengakses halaman antarmuka dengan menggunakan *web browser*.



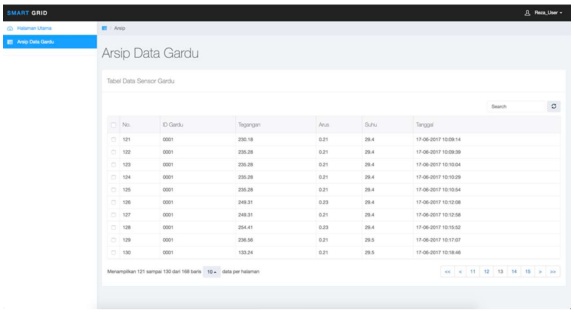
Gambar 4.11 Antarmuka Halaman Login Pada Laptop.



Gambar 4.12 Antarmuka Halaman Admin Tambah Akun Pada Laptop.

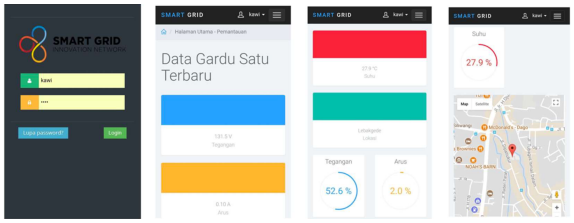


Gambar 4.13 Antarmuka Halaman Utama User Pada Laptop.

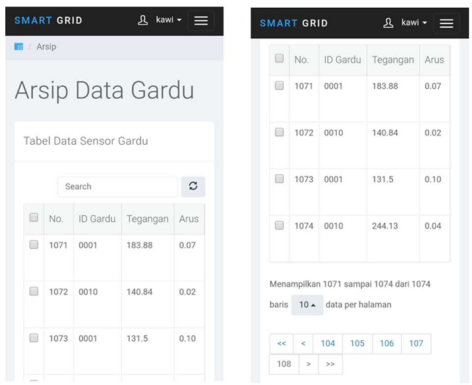


Gambar 4.14 Antarmuka Halaman Arsip Data Gardu User Pada Laptop.

Pada Gambar 4.11 antarmuka mewajibkan validasi *user* terlebih dulu. Jika *user* memiliki hak akses *user* biasa, maka akan diarahkan ke halaman antarmuka *user*. Jika *user* memiliki hak akses sebagai admin, maka *user* akan diarahkan ke halaman antarmuka *admin*.



Gambar 4.15 Antarmuka Halaman Login dan Utama Pada Smartphone.



Gambar 4.16 Antarmuka Halaman Arsip Pada Smartphone.

5. KESIMPULAN DAN SARAN

5.1 Kesimpulan

Dari hasil pengujian yang telah dilakukan, kesimpulan yang dapat diambil adalah sebagai berikut:

1. *Middleware* yang dibangun mampu mengakuisisi data tegangan, arus, suhu, koordinat lokasi, dan tanggal untuk aplikasi *Smart Grid*.
2. *Middleware* yang dibangun mampu menyediakan format data JSON untuk aplikasi berbasis *web*.
3. *Middleware* yang dibangun mampu diakses oleh perangkat yang mendukung aplikasi *web*.

5.2 Saran

Berikut adalah beberapa saran yang dapat dijadikan pertimbangan untuk pengembangan perancangan *middleware* untuk aplikasi *Smart Grid* ini selanjutnya:

1. Menambahkan enkripsi untuk semua data gardu untuk aplikasi *Smart Grid*.
2. Menambahkan fitur beberapa gardu untuk aplikasi *Smart Grid*.
3. Menambahkan fitur *real-time* notifikasi peringatan langsung ke *smartphone* bila terjadinya beban berlebih pada gardu.
4. Menggunakan *IP Public* statis untuk *web server middleware*.

DAFTAR PUSTAKA

[1] J. F. Martinez, J. R. Molina, P. Castillejo, and R. de Diego, "Middleware Architectures for the Smart Grid: Survey and Challenges in the Foreseeable Future," *Energies,* no. 6, pp. 3593-3621, 2013.

[2] BPPT, *Teknologi Smart Grid Untuk Smart City*. Jakarta: Badan Pengkajian dan Penerapan Teknologi, 2012.

[3] W3C, "Web Services Architecture Working Group," 11 Februari 2004. [Online]. Available: http://w3.org

[4] S. R. Pungus, "Penerapan Service Oriented Architecture Untuk Pengintegrasian Sistem Informasi Perguruan Tinggi," Tesis (Sekolah Teknik Elektro dan informatika Institut Teknologi Bandung), 2008.