BAB III

ANALISIS DAN PERANCANGAN SISTEM

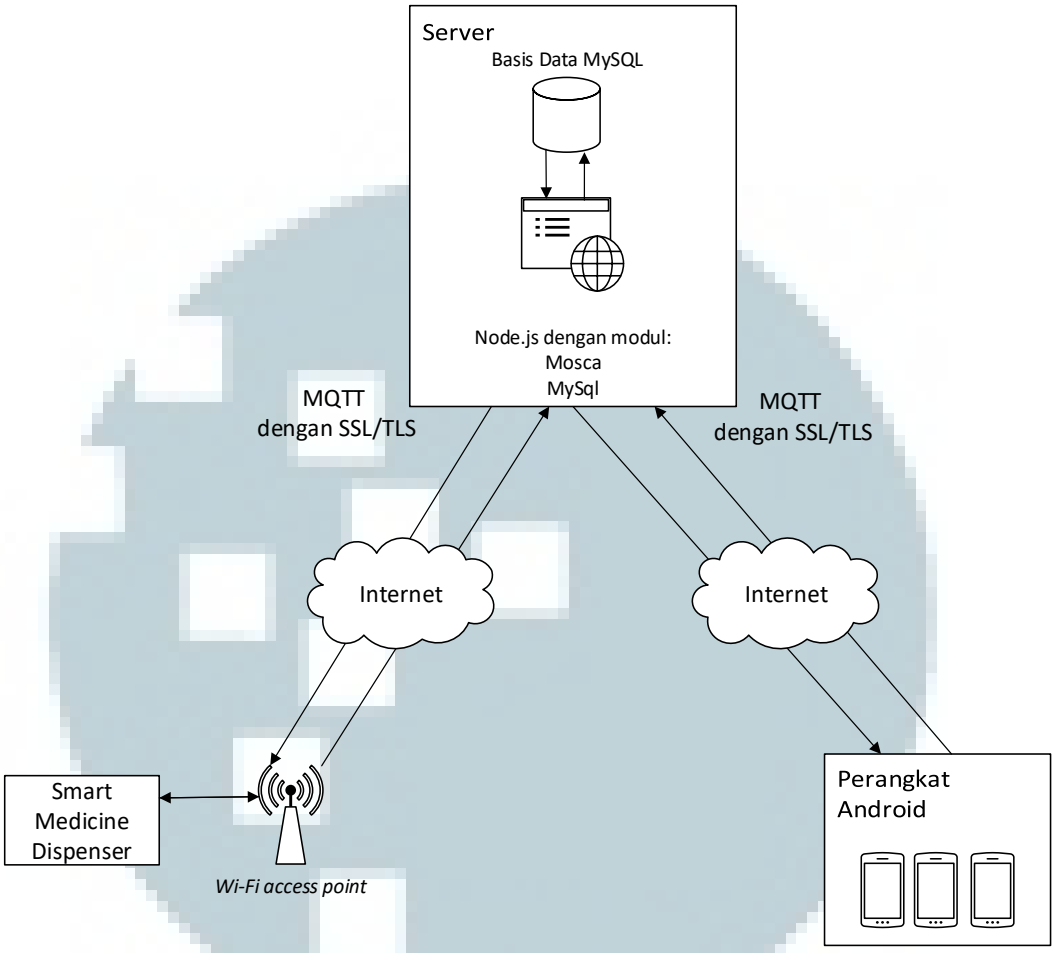
3.1 Spesifikasi Smart Medicine Dispenser

Smart Medicine Dispenser yang dirancang memiliki spesifikasi sebagai berikut:

1) *Smart Medicine Dispenser* memiliki dua buah fungsi yaitu untuk obat pil dan obat cair.
2) Kapasitas maksimum dispenser obat pil adalah 20 butir;
3) Kapasitas maksimum dispenser obat cair adalah 60 ml ;
4) Obat pil dikeluarkan satu per satu hingga jumlah pil sesuai dengan yang dibutuhkan oleh pasien;
5) Obat cair digunakan dengan takaran per 5ml atau satu sendok teh;
6) Mekanisme *dispense* untuk pil dispenser adalah dengan memutarkan motor servo hingga tracker sensor menerima sinyal bahwa obat telah berhasil keluar sejumlah yang dibutuhkan pasien;
7) Mekanisme dispense untuk obat cair dispenser adalah dengan menyalakan relay untuk mengaktifkan motor dc peristatic dengan delay 5000 untuk 5ml obat cair;
8) Pengeluaran obat lebih dari 5ml maka akan dikeluarkan per 5ml;
9) Tempat obat pil dan cair dikondisikan berada di tempatnya;
10) Pembaruan stok obat berdasarkan dari perhitungan oleh sistem tanpa menggunakan sensor pengukuran tambahan;

11) Jumlah pengisian stok obat dilakukan melalui masukan data pada aplikasi dengan konfirmasi dari perangkat pintar;
12) Aplikasi Android digunakan untuk mendaftarkan perangkat pintar, mengatur jadwal pengeluaran obat , melihat waktu log konsumsi obat dan konfirmasi pengisian stok obat;
13) Alarm obat akan menyala selama 15 menit hingga obat diambil;
14) Jika dalam 15 menit obat tidak diambil maka akan dikirim obat tidak diambil meskipun obat diambil setelah 15 menit;
15) Obat yang telah dikonsumsi ataupun tidak dikonsumsi dapat dilihat pada pilihan menu log di aplikasi;
16) Protokol komunikasi yang digunakan perangkat pintar adalah *Wi-Fi*;
17) Protokol pertukaran pesan untuk seluruh pertukaran data yang digunakan adalah MQTT;
18) Pengendali mikro yang digunakan adalah Arduino Uno yang berbasis ATmega328;
19) Menggunakan modul RTC (*real-time clock)* DS3231 sebagai penentu waktu pada sistem *Smart Medicine Dispenser*;
20) Modul *Wi-Fi* yang digunakan adalah ESP8266 berupa *development board* yang telah terpasang *firmware* NodeMCU;
21) Pengguna dapat mengakses perangkat pintar melalu smartphone Android;
22) Perangkat pintar akan mengeluarkan jumlah obat sesuai dengan jadwal yang telah diatur oleh pengguna.

3.2 Rancangan Sistem Keseluruhan



Gambar 3. 1 Diagram Arsitektur Keseluruhan

Diagram pada gambar 3.1 merupakan gambaran arsitektur sistem perangkat pintar secara keseluruhan. Perangkat pintar berkomunikasi dengan server melalui modul Wi-Fi ESP8266 yang berupa *development board* NodeMCU dengan protokol MQTT yang terenkripsi menggunakan OpenSSL. Data yang diterima oleh ESP8266 dari server akan dikirim ke Arduino UNO melalui komunikasi serial UART. Untuk mengirimkan perintah ke server , Arduino akan menggunakan cara yang digunakan saat menerima data dari server. Data dalam semua komunikasi dalam sistem ini berupa JSON *object.*

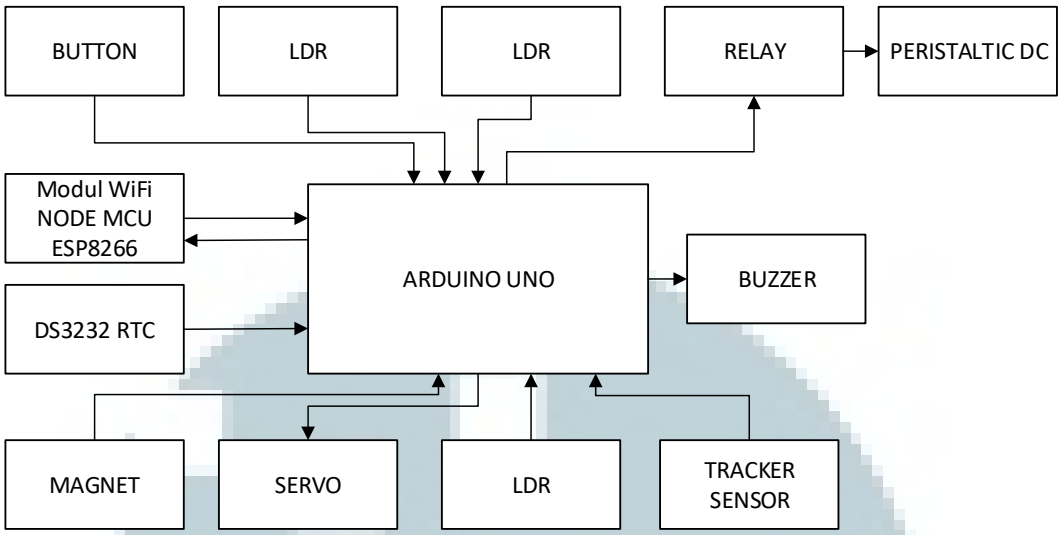
Aplikasi pada perangkat Android berfungsi sebagai pusat kendali perangkat-perangkat pintar yang dimiliki oleh pengguna. Melalui aplikasi ini, pengguna yang belum terdaftar dapat mendaftarkan diri terlebih dahulu dengan mengisi data-data yang dibutuhkan dan kemudian disimpan dalam basis data. Jika proses pendaftaran telah selesai, maka pengguna dapat melakukan *log-in* menggunakan data *username* dan *password* yang telah didaftarkan sebelumnya. Setelah berhasil melakukan *log-in*, pengguna dapat melakukan pendaftaran perangkat yang dimilikinya, mengakses informasi terkait perangkat pintar yang dimiliki, memberikan perintah ke perangkat pintar, dan dapat mengakses serta memperbarui informasi pengguna. Seluruh perintah dari aplikasi yang dikirimkan ke server berupa JSON *string* kemudian diolah oleh server.

Peran *server* dalam sistem ini adalah melakukan otentikasi dan otorisasi setiap perangkat pintar maupun perangkat Android yang melakukan koneksi dan mengatur lalu lintas data berdasarkan topik. Otentikasi dilakukan dengan mencocokkan data *username* dan *password* yang dikirim perangkat untuk melakukan koneksi dengan data *username* dan *password* yang tercatat pada basis data. Otorisasi pada sistem ini berupa otorisasi topik yang boleh digunakan oleh sebuah perangkat untuk melakukan *publish* dan *subscribe* data. Server menerima data JSON dari perangkat pintar dan aplikasi Android.

3.3 Rancangan Perangkat Keras

Komponen-komponen yang digunakan untuk membangun *Smart Medicine Dispenser* adalah:

1) Arduino Uno R3;
2) Satu buah modul *Wi-Fi* ESP8266 yang berupa *development board* yang telah terpasang *firmware* NodeMCU;
3) Satu buah modul RTC (*real-time clock*) DS3231 sebagai penentu waktu pada sistem *Smart Medicine Dispenser*;
4) Satu buah motor servo MG995 360^0 *continuous* untuk mengeluarkan obat pill;
5) Satu buah tracker sensor untuk menghitung jumlah obat yang telah dikeluarkan;
6) Satu buah magnet sensor untuk mengecek apakah tempat stok sedang dibuka atau ditutup;
7) Satu buah motor dc peristaltic untuk mengambil obat cair;
8) Satu buah modul relay untuk menghidupkan motor dc peristaltic;
9) Satu buah buzzer sebagai alarm;
10) Satu buah button untuk pengaturan motor dc peristaltic;
11) 3 buah sensor cahaya
    - Digunakan untuk mengecek status stok obat cair;
    - Digunakan untuk mengecek apakah obat cair sudah diambil;
    - Digunakan untuk mengecek apakah obat pill sudah diambil
12) Modul step down;
13) Satu buah catu daya 12v.



Gambar 3. 2 Blok Diagram Smart Medicine

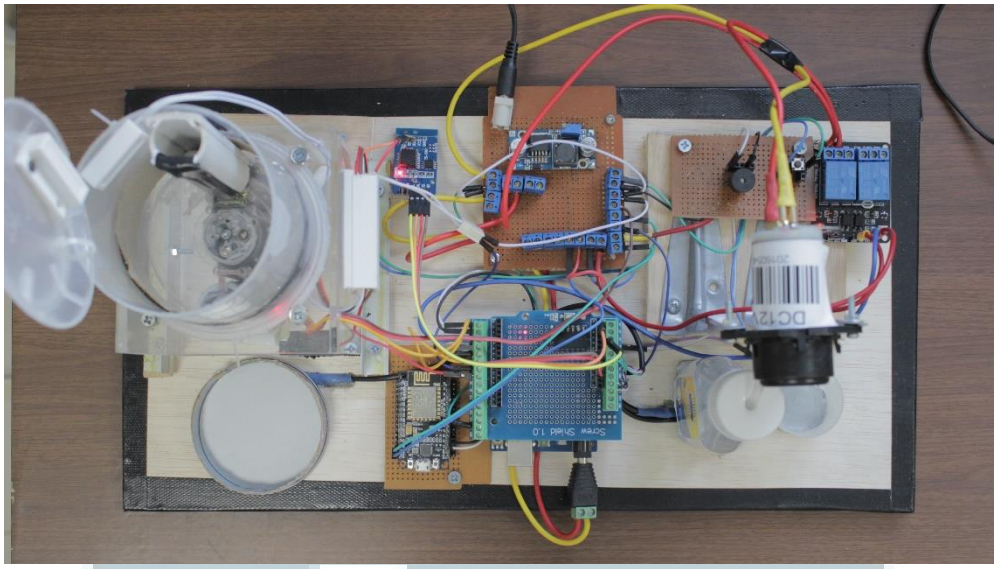
Gambar 3.2 menggambarkan diagram blok dari *Smart Medicine Dispenser*. Pada bagian tengah merupakan mikrokontroler yang dipakai pada *Smart Medicine Dispenser*. Arduino UNO mengatur seluruh sistem yang ada pada *Smart Medicine Dispenser*. Arduino UNO akan memberikan perintah untuk menggerakkan motor servo , menyalakan dan mematikan relay dan juga akan membunyikan buzzer. Selain memberikan perintah, Arduino Uno juga bertugas untuk menerima perintah yang berasal dari masukan berupa bacaan dari sensor *tracker* , tiga buah Light dependent Resistor , magnet sensor, sebuah tombol dan RTC yang digunakan untuk mendapatkan waktu.

Komponen yang digunakan pada dispenser obat pil adalah magnet sensor, tracker sensor, satu buah light dependent resistor dan motor servo. Magnet sensor berfungsi untuk mengecek apakah tempat obat sedang terbuka atau tertutup. Motor servo berfungsi untuk memutarkan tempat obat agar obat keluar sesuai dengan kebutuhan pengguna. Tracker sensor berfungsi untuk melakukan pengecekan apakah obat yang dikeluarkan sudah sesuai dengan jumlah yang dibutuhkan oleh

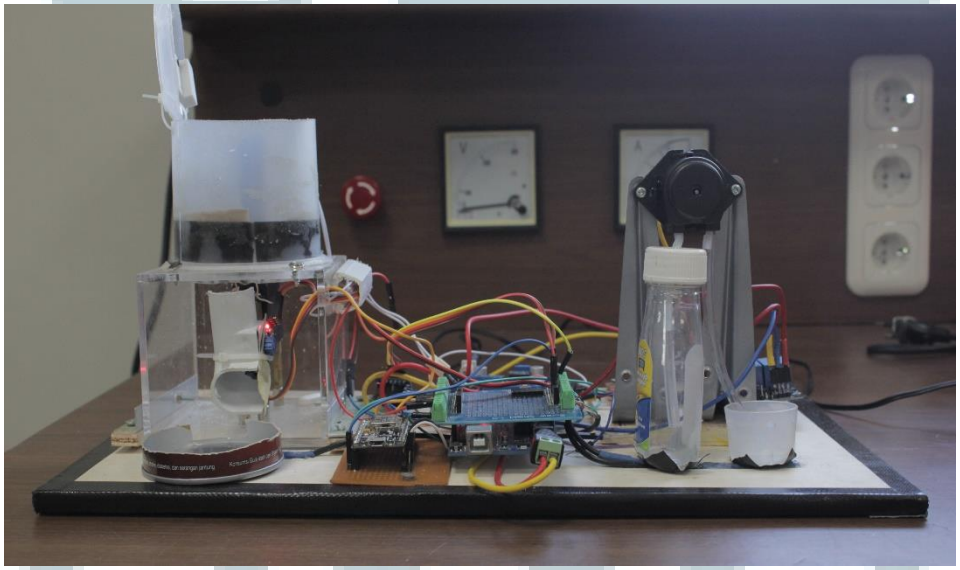
pengguna. *Light dependent Resistor* berfungsi sebagai bacaan sensor untuk memastikan obat telah diambil oleh pengguna.

Pada dispenser obat cair , komponen yang digunakan adalah tombol , dua buah light dependent resistor , relay dan satu buah motor dc peristaltic. Tombol digunakan untuk mengatur obat cair agar masuk kedalam selang pada motor dc peristaltic. Dua buah light dependent resistor digunakan untuk mengecek apakah ada botol obat dan apakah obat telah diambil. Relay berfungsi untuk mengontrol nyala dan mati motor dc peristaltic.

Komponen lain yang yang terpasang seperti modul WiFi ESP8266 berfungsi sebagai media komunikasi Arduino Uno dengan server. Buzzer berfungsi untuk memberikan informasi kepada pengguna apabila obat telah dikeluarkan namun belom diambil. DS3231 RTC berguna sebagai jam pada Arduino Uno yang berfungsi untuk mengatur *alarm* untuk kemudian diproses oleh Arduino Uno.



Gambar 3. 3 Tampak atas Smart Medicine Dispenser



Gambar 3. 4 Tampak depan Smart Medicine Dispenser

3.4 Rancangan Perangkat Lunak

Rancangan perangakat lunak pada *Smart Medicine Dispenser* terdiri dari rancangan protokol komunikasi, rancangan basis data, rancangan perangkat lunak *Smart Medicine Dispenser*,rancangan NodeMCU dan rancangan aplikasi Android.

3.4.1 Protokol Komunikasi

Dalam protokol komunikasi terdapat dua bagian komunikasi yang terdiri dari komunikasi perangkat keras ke server dan komunikasi perangkat Android ke server.

A. Komunikasi Perangkat Keras ke Server

Komunikasi perangkat keras dengan server merupakan komunikasi antara *Smart Medicine Dispenser* dengan server. Untuk dapat berkomunikasi dengan server , perangkat keras menggunakan modul Wi-Fi ESP8266 yang berupa *development board* NodeMCU. NodeMCU bertugas untuk mengkomunikasikan setiap perintah ke server ataupun ke Arduino.

Dalam komunikasi ini, terdapat beberapa format komunikasi yang dipakai dalam format JSON. Baik server ataupun *Smart Medicine Dispenser* akan mengambil isi dari atribut *msg* untuk diolah sesuai dengan fungsinya. Berikut adalah format komunikasi yang digunakan.

1) Mengambil Informasi Pemilik Perangkat Pintar

Perintah *getOwner* dipakai untuk mengambil informasi pemilik perangkat pintar. Informasi mengenai *username* ini diperlukan untuk melakukan komunikasi lain. Pesan ini akan dikirimkan melalui topik */smd/<serial_number>*. Berikut adalah format pengiriman yang dikirimkan yang dapat dilihat pada Tabel 3.1.

Tabel 3. 1 Tabel Format Pengiriman getOwner

Name	Value	Keterangan
msg	getOwner	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “getOwner”,
“applianceserial” : “123456789012”
}		

Server akan mengirimkan format balasan sebagai berikut pada Tabel 3.2.

Tabel 3. 2 Tabel Format Balasan getOwner dari server

Name	Value	Keterangan
msg	owner	Identifikasi pesan.
owner	<username_pemilik> / NOTFOUND	Berisi username pemilik perangkat pintar tersebut. Mengembalikan NOTFOUND jika tidak ditemukan.

Contoh:
{
“msg” : “owner”,
“owner” : “user1”
}		

Balasan dari server akan bernilai *“NOTFOUND”* bila perangkat pintar belum memiliki pemilik. Sedangkan, bila sudah memiliki pemilik, akan menghasilkan nilai berupa *username* pemilik yang akan dipakai untuk melakukan komunikasi pada topik */smd/<username>*.

2) Mengambil Status Perangkat Tidak Aktif

Status perangkat dibutuhkan untuk memastikan perangkat siap untuk menerima dan mengirimkan perintah. Bila status perangkat sedang tidak aktif,

server tidak dapat memberikan perintah apapun kepada perangkat. Server akan mengubah status perangkat menjadi aktif secara otomatis bila perangkat pintar mengirimkan pesan *getOwner*.

Pesan yang diberikan untuk menyatakan status tidak aktif adalah *offline*. Topik yang dipakai untuk pesan ini adalah */smd/<username>*. Berikut format pengiriman yang dikirimkan pada Tabel 3.3.

Tabel 3. 3 Tabel Format Pengiriman Status Offline

Name	Value	Keterangan
msg	offline	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “offline”,
“applianceserial” : “123456789012”
}		

3) Mengambil Jadwal Pemberian Obat dari basis data

Untuk mengambil jadwal mengeluarkan obat, *Smart Medicine Dispenser* akan mengirimkan sebuah JSON *object* ke server dengan topik */smd/<username>*. Format perintah yang dikirimkan adalah sebagai berikut pada Tabel 3.4.

Tabel 3. 4 Tabel Format Pengiriman getMedicineSchedule

Name	Value	Keterangan
msg	getMedicineSchedule	Identifikasi pesan.
owner	<username>	Username pemilik perangkat pintar.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
Contoh: { “msg” : “getMedicineSchedule”, “owner” : “user1”,		

“applianceserial” : “210987654321”
}

Kemudian server akan mengolah perintah yang dikirimkan oleh *Smart Medicine Dispenser* dan mengirimkan dua balasan. Format balasan dari server adalah sebagai berikut pada Tabel 3.5 dan Tabel 3.6.

Tabel 3. 5 Tabel Format Balasan Server untuk pesan Medicine

Name	Value	Keterangan
msg	medicine	Identifikasi pesan.
owner	<username>	Username pemilik yang telah terdaftar.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
pilschedule	<array_jadwal_pil>	Array berisi jadwal pemberian pill pada pill dispenser ke n.
pilamount	<jumlah_pil>	Jumlah pill yang harus dikeluarkan oleh pill dispenser ke n (dalam butir).
lqschedule	<array_jadwal_obat_cair>	Array berisi jadwal pemberian obat cair pada solution dispenser ke n.
lqamount	<volume_obat_cair>	Jumlah obat cair yang harus dikeluarkan oleh solution dispenser ke n (dalam ml).

Contoh:
{
“msg” : “medicine”,
“owner” : “user1”,
“applianceserial” : “210987654321”,
"pilschedule" :["08:00","12:00","20”00"],
"pilamount" :8,
"lqschedule" :["08:00","12:00","20:00"],
"lqamount" :8
}		

Server akan mengirimkan format balasan sebagai berikut pada Tabel 3.6.

Tabel 3. 6 Tabel Format Balasan cmcid dari Server

Name	Value	Keterangan
msg	cmdid	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

cmdid	<id_perintah> / -1	Berisi id perintah yang dikirimkan, didapat dari *auto-increment* pada basis data.

Contoh:
{
“msg” : “cmdid”,
“applianceserial” : “210987654321”,
“cmdid” : 2
}		

4) Memberikan notifikasi apakah obat telah dikonsumsi ketika waktunya

Untuk memberikan jika obat telah diambil, *Smart Medicine Dispenser* akan mengirimkan sebuah JSON *object* ke server dengan topik */smd/<username>*. Format perintah yang dikirimkan adalah sebagai berikut pada Tabel 3.4.

Tabel 3. 7 Tabel Format pengiriman medicineTook

Name	Value	Keterangan
msg	medicineTook	Identifikasi pesan.
rtimestamp	<timestamp_laporan_kondisi_obat>	Waktu pelaporan kondisi obat saat jadwal minum obat tiba.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
rprt	<laporan_obat_dimakan>	Berisi laporan apakah makanan telah disajikan, dimakan, atau tidak dimakan.
cmdid	<id_perintah>	Berisi id perintah penjadwalan yang telah diselesaikan.

Contoh:
{
“msg” : “medicineTook”,
“rtimestamp” : “2016-04-15 19:10:00”,
“applianceserial” : “210987654321”,
“rprt” : “Pill taken”,
“cmdid” : 2
}		

5) Memberikan notifikasi ketika pada waktunya , obat belum diminum

Untuk memberikan notifikasi ketika obat belum diminum pada waktunya, *Smart Medicine Dispenser* akan mengirimkan sebuah JSON *object* ke server dengan topik */smd/<username>*. Format perintah yang dikirimkan adalah sebagai berikut pada Tabel 3.4.

Tabel 3. 8 Tabel Format Pengiriman medicineNotTook

Name	Value	Keterangan
msg	medicineNotTook	Identifikasi pesan.
owner	<username>	Username pemilik yang telah terdaftar.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “medicineNotTook”,
“owner” : “user1”,
“applianceserial” : “210987654321”,
}		

6) Mengambil informasi stok obat

Untuk mengambil informasi stok obat, *Smart Medicine Dispenser* akan mengirimkan sebuah JSON *object* ke server dengan topik */smd/<username>*. Format perintah yang dikirimkan adalah sebagai berikut pada Tabel 3.4.

Tabel 3. 9 Tabel format pengiriman smdStock

Name	Value	Keterangan
msg	smdStock	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
pil	<jumlah_pil_tabung_1> / -1	Berisi jumlah pil (dalam gram).
solution	<volume_obat_cair> / -1	Berisi volume obat cair (dalam ml).

Contoh:
{
“msg” : “smdStock”,
“applianceserial” : “210987654321”,
“pil” : 60,
“liquid” : 500
}		

7) Memberikan konfirmasi perubahan stok

Untuk memberikan konfrimasi perubahan stok, *Smart Medicine Dispenser* akan mengirimkan sebuah JSON *object* ke server dengan topik */smd/<username>*. Format perintah yang dikirimkan adalah sebagai berikut pada Tabel 3.4.

Tabel 3. 10 Tabel Format pengiriman getMedStock

Name	Value	Keterangan
msg	getMedStock	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “getMedStock”,
“applianceserial” : “210987654321”
}		

Kemudian server akan mengolah perintah yang dikirimkan oleh *Smart Medicine Dispenser* dan mengirimkan balasan dengan format balasan dari server adalah sebagai berikut pada Tabel 3.5.

Tabel 3. 11 Tabel format balasan smdStock

Name	Value	Keterangan
msg	smdStock	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
pil	<jumlah_pil_tabung_1> / -1	Berisi jumlah pil (dalam gram).
solution	<volume_obat_cair> / -1	Berisi volume obat cair (dalam ml).

Contoh:
{
“msg” : “smdStock”,
“applianceserial” : “210987654321”,
“pil” : 60,
“liquid” : 500
}		

B. Komunikasi Perangkat Android ke Server

Komunikasi perangkat Android dengan server merupakan komunikasi antara aplikasi android dengan server. Dalam komunikasi ini, terdapat beberapa format komunikasi yang dipakai dalam format JSON. Baik server ataupun aplikasi android akan mengambil isi dari atribut *msg* untuk diolah sesuai dengan fungsinya. Berikut adalah format komunikasi yang digunakan.

1) Mendaftarkan Pengguna Baru

Perintah *signup* digunakan untuk mendaftarkan pengguna baru pada aplikasi Appliance Hub. Pengguna akan memasukkan data-data yang diperlukan dan dikemas dalam format JSON. Pesan ini akan dikirimkan pada topik */signup/<username>*. *Username* akan didapat dari hasil masukan data dari pengguna pada halaman *sign up*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.12.

Tabel 3. 12 Tabel Format Pengiriman Perintah Sign Up

Name	Value	Keterangan
msg	signup	Identifikasi pesan.
username	<username>	Username baru yang akan didaftarkan.
password	<password>	Password dari username baru yang akan didaftarkan dan akan di-*hashing* SHA2-256.
fullname	<fullname>	Nama lengkap pengguna yang akan didaftarkan.
email	<email>	Email pengguna yang akan didaftarkan.
phone	<phone_number>	Nomor telepon pengguna yang akan didaftarkan.
Contoh: { “msg” : “signup”, “username” : “user1”, “password” : “0a041b9462caa4a31bac3567e0b6e6fd9100787db2ab433d96f6d178cabfce90”, “fullname” : “User One”,		

“email” : “user1@domain.com”,
“phone” : “081818181818”
}

Server akan memberikan balasan seperti yang dapat dilihat pada Tabel 3.13.

Tabel 3. 13 Tabel Format Balasan Pesan dari Sign Up dari Server

Name	Value	Keterangan
msg	signupStatus	Identifikasi pesan.
status	SUCCESS / FAIL	Menyatakan apakah pengguna berhasil didaftarkan atau tidak.

Contoh:
{
“msg” : “signupStatus”,
“status” : “SUCCESS”
}		

2) Mendapatkan Perangkat pintar

Untuk mendapatkan perangkat pintar yang dimiliki, aplikasi akan mengirimkan pesan *getMyAppliance*. Pesan ini dikirimkan melalui topik */smd/<username>*. Berikut format pengiriman pesan yang dapat dilihat pada Tabel 3.14.

Tabel 3. 14 Tabel Format Pengiriman getMyAppliance

Name	Value	Keterangan
msg	getMyAppliance	Identifikasi pesan.
owner	<username_pemilik>	Berisi username pemilik yang menginginkan daftar perangkat yang dimilikinya.

Contoh:
{
“msg” : “getMyAppliance”,
“owner” : “user1”
}		

Server akan mengirimkan balasan seperti pada Tabel 3.15.

Tabel 3. 15 Tabel Format Balasan getMyAppliance dari Server

Name	Value	Keterangan
msg	myAppliance	Identifikasi pesan.
dat	<array_appliance_detail> / NOTFOUND	Berisi array JSON object yang terdiri atas

		applianceserial, applianceid, nickname, status. Mengembalikan NOTFOUND jika tidak ditemukan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
applianceid	<applianceid>	Jenis perangkat pintar.
nickname	<nickname>	Nama perangkat yang diberikan oleh pemilik.
status	ACTIVE / INACTIVE	Status perangkat, apakah aktif/tidak.

Contoh:

```
{
“msg” : “myAppliance”
“dat”  : [
            {
            “applianceserial” : “123456789012”,
            “applianceid"      : “rck”,
            “nickname”        : “Kitchen Rice Cooker”,
            “status”          : “INACTIVE”
            },
            {
            “applianceserial” : “210987654321”,
            “applianceid"      : “smd”,
            “nickname”        : “Grandma’s Pill Dispenser”,
            “status”          : “ACTIVE”
            },
            {
            “applianceserial” : “789012345612”,
            “applianceid"      : “sdf”,
            “nickname”        : “Brownie’s Feeder”,
            “status”          : “ACTIVE”
            }
        ]
}
```

3) Mendaftarkan perangkat pintar

Perintah yang digunakan untuk mendaftarkan perangkat pintar adalah *registerMyAppliance*. Dengan perintah ini, sebuah perangkat akan terdaftar telah memiliki pemilik dan dapat diakses melalui aplikasi Android. Perintah ini

akan dikirimkan melalui topik */general/<username>*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.16.

Tabel 3. 16 Tabel Format Pengiriman Pesan registerMyAppliance

Name	Value	Keterangan
msg	registerMyAppliance	Identifikasi pesan.
owner	<username_pemilik>	Berisi username pemilik perangkat pintar tersebut.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “registerMyAppliance”,
“owner” : “user1”,
“applianceserial” : “123456789012”
}		

Server akan mengirimkan balasan dengan format yang dapat dilihat pada Tabel 3.17.

Tabel 3. 17 Tabel Format Balasan Pesan register MyAppliance dari Server

Name	Value	Keterangan
msg	registrationStatus	Identifikasi pesan.
status	SUCCESS / FAIL	Menyatakan apakah pemilik perangkat berhasil didaftarkan.

Contoh:
{
“msg” : “registrationStatus”,
“status” : “SUCCESS”
}		

4) Menghapus perangkat pintar

Perintah *unregisterMyAppliance* dapat dipakai bila pengguna ingin menghapus perangkat dari *database*. Dengan memberikan perintah ini, seluruh data yang berkaitan dengan perangkat tersebut akan hilang. Pesan ini dikirimkan melalui topik */general/<username>*. Berikut format lengkap pengiriman yang dapat dilihat pada Tabel 3.18.

Tabel 3. 18Tabel Format Pengiriman Pesan unregisterAppliance

Name	Value	Keterangan
msg	unregisterMyAppliance	Identifikasi pesan.
owner	<username_pemilik>	Berisi username pemilik perangkat pintar tersebut.
password	<password>	Password dari username baru yang akan didaftarkan dan akan di-*hashing* SHA2-256.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “unregisterMyAppliance”,
“owner” : “user1”,
“applianceserial” : “123456789012”
}		

Server akan memberikan balasan dengan format yang dapat dilihat pada Tabel 3.19.

Tabel 3. 19 Tabel Format Balasan Pesan unregisterMyAppliance dari Server

Name	Value	Keterangan
msg	unregisterStatus	Identifikasi pesan.
status	SUCCESS / FAIL	Menyatakan apakah pemilik perangkat berhasil didaftarkan.

Contoh:
{
“msg” : “unregisterStatus”,
“status” : “SUCCESS”
}		

5) Memperbarui Profil Pengguna

Perintah *updateProfile* digunakan untuk membarui profil pengguna selain *username* dan *password*. Pesan ini dikirimkan melalui topik */general/<username>*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.22.

Tabel 3. 20 Tabel Format Pengiriman Pesan updateProfile

Name	Value	Keterangan
msg	updateProfile	Identifikasi pesan.
username	<username>	Username yang telah terdaftar.
password	<password>	Password yang di-*hashing* SHA2-256.
fullname	<fullname>	Nama lengkap pengguna yang terdaftar.
email	<email>	Email baru pengguna.
phone	<phone_number>	Nomor telepon pengguna yang baru.

Contoh:
{
“msg” : “updateProfile”,
“username” : “user1”,
“password” : “0a041b9462caa4a31bac3567e0b6e6fd9100787db2ab433d96f6d178cabfce90”,
“fullname” : “User One”,
“email” : “**user1@domain.com**”,
“phone” : “081616161616”
}		

Server akan membalas dengan format pada Tabel 3.23.

Tabel 3. 21 Tabel Format Balasan Pesan updateProfileStatus dari Server

Name	Value	Keterangan
msg	updateProfileStatus	Identifikasi pesan.
status	SUCCESS / FAIL	Menyatakan apakah profil pengguna berhasil diperbarui atau tidak.

Contoh:
{
“msg” : “updateProfileStatus”,
“status” : “SUCCESS”
}		

6) Mengambil profil pengguna

Perintah *getProfile* digunakan untuk mengambil informasi terkait profil pengguna. Pesan ini dikirimkan melalui topik */general/<username>*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.20.

Tabel 3. 22 Tabel Format Pesan getProfile

Name	Value	Keterangan
msg	getProfile	Identifikasi pesan.
username	<username>	Username yang telah terdaftar.

password	<password>	Password yang di-*hashing* SHA2-256.

Contoh:
{
“msg” : “getProfile”,
“username” : “user1”,
“password” :
“0a041b9462caa4a31bac3567e0b6e6fd9100787db2ab433d96f6d178cabfce90”
}		

Server kemudian akan mengirimkan balasan dengan format seperti pada Tabel 3.21.

Tabel 3. 23 Tabel Format Balasan Pesan myProfile dari Server

Name	Value	Keterangan
msg	myProfile	Identifikasi pesan.
username	<username> / NOTFOUND	Username yang telah terdaftar. Mengembalikan nilai NOTFOUND jika tidak ditemukan.
fullname	<fullname>	Nama lengkap pengguna yang terdaftar.
email	<email>	Email baru pengguna.
phone	<phone_number>	Nomor telepon pengguna yang baru.

Contoh:
{
“msg” : “myProfile”,
“username” : “user1”,
“fullname” : “User One”,
“email” : “**user1@domain.com**”,
“phone” : “081616161616”
}		

7) Memperbarui Password Pengguna

Untuk mengganti *password*, aplikasi akan mengirimkan pesan *updatePassword* melalui topik */general/<username>*. Format pengiriman pesan secara lengkap dapat dilihat pada Tabel 2.24.

Tabel 3. 24 Tabel Format pengiriman pesan updatePassword dari Server

Name	Value	Keterangan
msg	updatePassword	Identifikasi pesan.
username	<username>	Username yang telah terdaftar.

oldpassword	<old_password>	Password lama yang di-*hashing* SHA2-256.
newpassword	<new_password>	Password baru yang di-*hashing* SHA2-256.

Contoh:
{
“msg” : “updatePassword”,
“username” : “user1”,
“oldpassword”:
“0a041b9462caa4a31bac3567e0b6e6fd9100787db2ab433d96f6d178cabfce90”,
“newpassword” :
“0aa0s97fg8y0k09ausd0ahsa0jdj0safyad807fd9as0d8as9d8a0s09d7fhgdffh99d”
}		

Server akan mengirimkan balasan dengan format seperti pada Tabel 2.25.

Tabel 3. 25 Tabel Format Balasan Pesan updatePasswordStatus dari Server

Name	Value	Keterangan
msg	updatePasswordStatus	Identifikasi pesan.
status	SUCCESS / FAIL	Menyatakan apakah password pengguna berhasil diperbarui atau tidak.

Contoh:
{
“msg” : “updatePasswordStatus”,
“status” : “SUCCESS”
}		

8) Merubah nama Panggilan perangkat

Perangkat-perangkat pintar dalam sistem Appiance Hub dapat diberi nama panggilan (*nickname)* sesuai keinginan pemilik. Perintah yang dipakai untuk merubah nama tersebut adalah *renameAppliance*. Pesan ini akan dikirim melalui topik */general/*<*username*>. Berikut format pengiriman yang dapat dilihat pada Tabel 3.26.

Tabel 3. 26 Tabel Format Pengiriman Pesan renameAppliance

Name	Value	Keterangan
msg	renameAppliance	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

owner	<owner>	Berisi username pemilik perangkat pintar tersebut.
nickname	<nickname>	Berisi nama perangkat pintar

Contoh:
{
“msg” : “renameAppliance”,
“applianceserial” : “123456789012”,
“owner” : “user1”,
“nickname” : “Smart Rice Cooker”
}		

Server akan membalas dengan format seperti pada Tabel 3.27.

Tabel 3. 27 Tabel Format Balasan Pesan renameStatus dari Server

Name	Value	Keterangan
msg	renameStatus	Identifikasi pesan.
status	SUCCESS / FAIL	Menyatakan apakah nama perangkat pengguna berhasil diperbarui atau tidak.

Contoh:
{
“msg” : “renameAppliance”,
“status” : “SUCCESS”
}		

9) Mendapatkan Status Perangkat

Perintah *getStatus* digunakan untuk mendapatkan informasi perangkat yang dikirimkan melalui topik */general/username*. Pesan ini menghasilkan nilai status aktif dan tidak aktif. Berikut format pengiriman yang dapat dilihat pada Tabel 3.28.

Tabel 3. 28 Tabel Format Pengiriman Pesan getStatus

Name	Value	Keterangan
msg	getStatus	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
owner	<owner>	Berisi username pemilik perangkat pintar tersebut.
Contoh: { “msg” : “getStatus”, “applianceserial” : “123456789012”,		

“owner” : “user1” }

Server kemudian akan mengirimkan balasan dengan format seperti pada Tabel 3.29.

Tabel 3. 29 Tabel Format Balasan Pesan applianceStatus dari Server

Name	Value	Keterangan
msg	applianceStatus	Identifikasi pesan.
status	ACTIVE / INACTIVE / NOTFOUND	Berisi status perangkat pintar. Mengembalikan NOTFOUND jika tidak ditemukan.

Contoh:
{
“msg” : “applianceStatus”,
“status” : “ACTIVE”
}		

10) Mendapatkan Log

Perintah *getLog* digunakan untuk mendapatkan catatan perintah dari aplikasi Android dan perangkat pintar. Pesan ini dikirimkan melalui topik */general/<username>*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.30.

Tabel 3. 30 Tabel Format Pengiriman Pesan getLog

Name	Value	Keterangan
msg	getLog	Identifikasi pesan.
owner	<username>	Username yang telah terdaftar.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “getLog”,
“owner” : “user1”,
“applianceserial” : “123456789012”
}		

Server kemudian akan mengirimkan balasan dengan format seperti pada Tabel 3.31.

Tabel 3. 31 Tabel Format Balasan Pesan reportLog dari Server

Name	Value	Keterangan
msg	reportLog	Identifikasi pesan.
dat	<array_appliance_detail> / NOTFOUND	Berisi array JSON object yang terdiri atas ctimestamp, cmd, rtimesamp, rprt. Mengembalikan NOTFOUND jika tidak ditemukan.
rtimestamp	<timestamp_report>	Berisi timestamp laporan diterima.
rprt	<laporan>	Berisi string JSON yang berisi laporan.

Contoh:
{
“msg” : “reportLog”,
“dat” : [
{
“rtimestamp” : “2016-04-15 15:15:00”,
“rprt” : “Cook rice finished”
},
{
“rtimestamp” : “2016-04-16 15:15:00”,
“rprt” : “Cook porridge finished”
}
]
}		

11) Memberikan jadwal pemberian Obat

Perintah yang digunakan untuk memberikan jadwal pengeluaran obat ke server adalah *medicine*. Pesan ini dikirimkan melalui topik */smd/<username>*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.32.

Tabel 3. 32 Format Pengiriman Pesan Medicine

Name	Value	Keterangan
msg	Medicine	Identifikasi pesan.
owner	<username>	Username yang telah terdaftar.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
pilschedule	<array_jadwal_pil>	Array berisi jadwal pemberian pill pada pill dispenser ke n.
pilamount	<jumlah_pil>	Jumlah pill yang harus dikeluarkan oleh pill dispenser ke n (dalam butir).

lqschedule	<array_jadwal_obat_cair>	Array berisi jadwal pemberian obat cair pada solution dispenser ke n.
lqamount	<volume_obat_cair>	Jumlah obat cair yang harus dikeluarkan oleh solution dispenser ke n (dalam ml).

Contoh:

```
{
“msg”                  : “medicine”,
“owner”                : “user1”,
“applianceserial”      : “210987654321”,
"pilschedule"          :["08:00","12:00","20:00"],
"pilamount"            :8,
"lqschedule"           :["08:00","12:00","20:00"],
"lqmount"              :8
}
```

Server kemudian akan mengirimkan balasan dengan format seperti pada tabel 3.33.

Tabel 3. 33 Format Balasan Pesan cmcid dari Server

Name	Value	Keterangan
msg	cmdid	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
cmdid	<id_perintah> / -1	Berisi id perintah yang dikirimkan, didapat dari *auto-increment* pada basis data.

Contoh:

```
{
“msg”                  : “cmdid”,
“applianceserial”      : “210987654321”,
“cmdid”                : 2
}
```

12) Memperbarui Stok Obat

Perintah yang digunakan untuk memperbaharui stok obat adalah *updateMedStock*. Pesan ini dikirimkan melalui topik */smd/<username>*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.32.

Tabel 3. 34 Format Pengiriman Pesan updateMedStock

Name	Value	Keterangan
msg	updateMedStock	Identifikasi pesan.
Applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.
pil	<jumlah_pil>	Berisi jumlah pil (dalam gram).
liquid	<volume_obat_cair>	Berisi volume obat cair (dalam ml).

Contoh:
{
“msg” : “updateMedStock”,
“applianceserial” : “210987654321”,
“pil” : 60,
“liquid” : 500
}		

13) Mengambil Stok Obat

Perintah yang digunakan untuk mengambil stok obat yang tersimpan di server untuk ditampilkan di aplikasi android adalah *getMedStock*. Pesan ini dikirimkan melalui topik */smd/<username>*. Berikut format pengiriman yang dapat dilihat pada Tabel 3.32.

Tabel 3. 35 Tabel pengiriman Pesan getMedStock

Name	Value	Keterangan
msg	getMedStock	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

Contoh:
{
“msg” : “getMedStock”,
“applianceserial” : “210987654321”
}		

Server kemudian akan mengirimkan balasan dengan format seperti pada tabel 3.36.

Tabel 3. 36 Tabel Balasan getMedStock dari Server

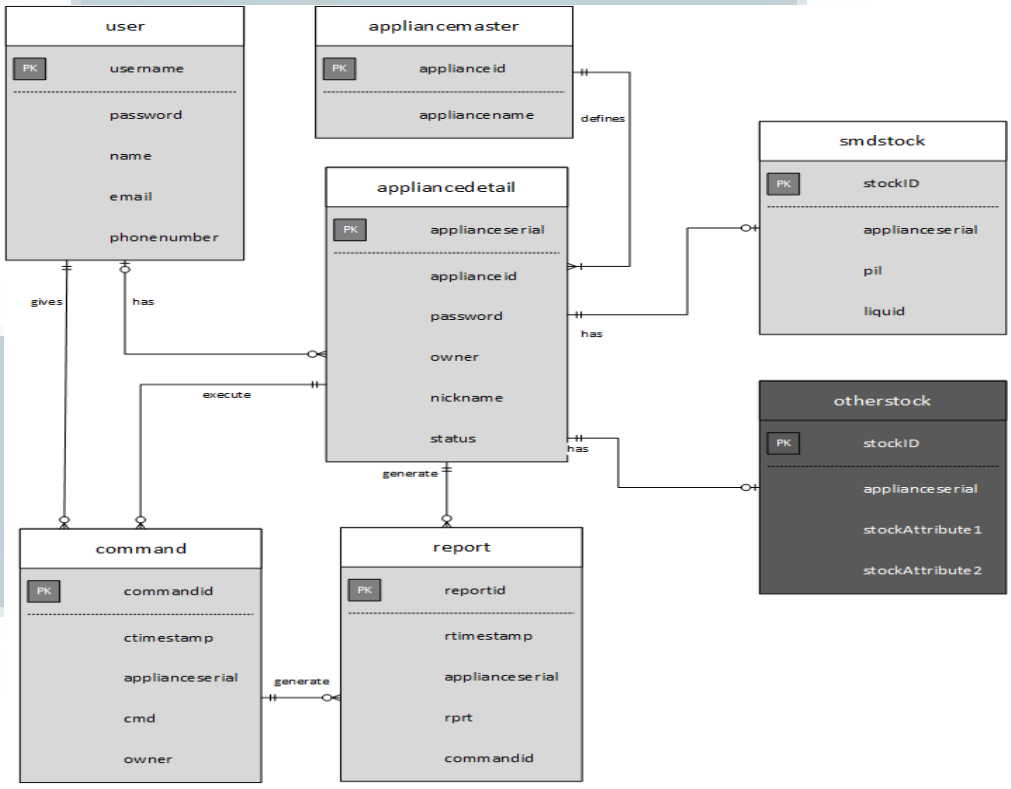
Name	Value	Keterangan
msg	smdStock	Identifikasi pesan.
applianceserial	<applianceserial>	Nomor serial perangkat pintar yang didapat dari alamat MAC perangkat pintar.

pil	<jumlah_pil_tabung_1> / -1	Berisi jumlah pil (dalam gram).
solution	<volume_obat_cair> / -1	Berisi volume obat cair (dalam ml).

Contoh:
{
“msg” : “smdStock”,
“applianceserial” : “210987654321”,
“pil” : 60,
“liquid” : 500
}		

3.4.2 Rancangan Basis Data

Smart Medicine Dispenser menggunakan sebuah basis data yang menggunakan MySQL dan diletakkan pada *cloud server*. Gambar 3.33 merupakan *Entity Relationship Diagram* (ERD) dari *database* sistem Appliance Hub khusus perangkat *Smart Medicice Dispenser*, dengan struktur dari setiap tabel adalah sebagai berikut.



Gambar 3. 5 Gambar ERD database

1) Tabel appliancemaster

Tabel 3. 37 Tabel appliancedetail pada Basis Data

#	Name	Type
1	**applianceserial**	varchar(15)
2	**applianceid**	varchar(3)
3	**password**	varchar(100)
4	**owner**	varchar(50)
5	**nickname**	varchar(50)
6	**status**	varchar(10)

Tabel *appliancedetail* dapat dilihat pada tabel 3.55. Tabel ini berisi 6 atribut, yaitu *applianceserial* yang merupakan nomor serial dari perangkat pintar yang terdaftar dan diambil dari alamat MAC perangkat pintar tersebut, *applianceid* yang merupakan *foreign key* yang mengacu pada *applianceid* pada tabel *appliancemaster, password* untuk validasi perangkat pada saat terhubung dengan server, *owner* yang juga merupakan *foreign key* yang mengacu pada *username* pada tabel *user*, *nickname* yang merupakan nama panggilan perangkat pintar, dan *status* untuk mengetahui apakah perangkat pintar tersebut sedang aktif atau tidak. Atribut *applianceserial* merupakan *primary key* pada tabel ini. Tabel ini digunakan untuk menyimpan daftar perangkat pintar yang dikenali oleh sistem dan dimiliki oleh pengguna.

2) Tabel Appliancedetail

Tabel 3. 38 Tabel appliancemaster pada Basis Data

#	Name	Type
1	**applianceid**	varchar(3)
2	**appliancename**	varchar(50)

Tabel *appliancemaster* dapat dilihat pada tabel 3.54. Tabel ini berisi 2 atribut, yaitu *applianceid* yang merupakan kode perangkat pintar yang terdaftar (untuk *Smart Medicine Dispenser*, *applianceid*-nya adalah smd) dan *appliancename* yang merupakan nama jenis perangkat pintar yang terdaftar. Atribut *applianceid* merupakan *primary key* pada tabel ini. Tabel ini digunakan untuk menyimpan daftar jenis perangkat pintar yang dikenali oleh sistem.

3) Tabel User

Tabel 3. 39 Tabel user pada Basis Data

#	Name	Type
1	**username**	varchar(50)
2	**password**	varchar(100)
3	**name**	varchar(25)
4	**email**	varchar(50)
5	**phonenumber**	varchar(15)

Tabel *user* dapat dilihat pada tabel 3.53. Tabel ini berisi 5 atribut, yaitu *username* yang merupakan *primary key* pada table ini, *password* yang telah di-*hash* menggunakan SHA2-256, *name*, *email*, dan *phonenumber*. Tabel ini digunakan sebagai pembanding pada saat pengguna melakukan *login* pada aplikasi, menyimpan data pengguna ketika *signup*, dan akan diperbarui

ketika pengguna memperbarui informasi *password*, email, dan *phonenumber*.

4) Tabel sdfStock

Tabel 3. 40 Tabel smdstock pada Basis Data

#	Name	Type
1	**stockID** 🔑	int(11)
2	**applianceserial**	varchar(15)
3	**pil**	int(5)
4	**liquid**	int(5)

Tabel *smdkstock* dapat dilihat pada tabel 3.60. Tabel ini berisi 4 atribut, yaitu *stockid* sebagai *identifier* stok, *applianceserial* yang merupakan *foreign key* yang mengacu pada *applianceserial* pada tabel *appliancedetail*, *pil* yang berisi data jumlah pil saat ini, dan *liquid* yang berisi data jumlah obat cair saat ini. Atribut *stockid* merupakan *primary key* pada tabel ini. Tabel ini digunakan untuk menyimpan data stok dari perangkat pintar *smart medicine dispenser*.

5) Tabel Command

Tabel 3. 41 Tabel command pada Basis Data

#	Name	Type
1	**commandid** 🔑	int(11)
2	**ctimestamp**	datetime
3	**applianceserial**	varchar(15)
4	**cmd**	varchar(256)
5	**owner**	varchar(50)

Tabel *command* dapat dilihat pada tabel 3.56. Tabel ini berisi 5 atribut, yaitu *commandid* sebagai *identifier* perintah, *ctimestamp* yang merupakan

waktu perintah diterima/dijalankan, *applianceserial* yang merupakan *foreign key* yang mengacu pada *applianceserial* pada tabel *appliancedetail*, *cmd* yang berisi perintah yang diterima, dan *owner* yang juga merupakan *foreign key* yang mengacu pada *username* pada tabel *user*. Atribut *commandid* merupakan *primary key* pada tabel ini. Tabel ini digunakan untuk menyimpan perintah-perintah yang dikirimkan pengguna kepada perangkat pintar.

6) Tabel Report

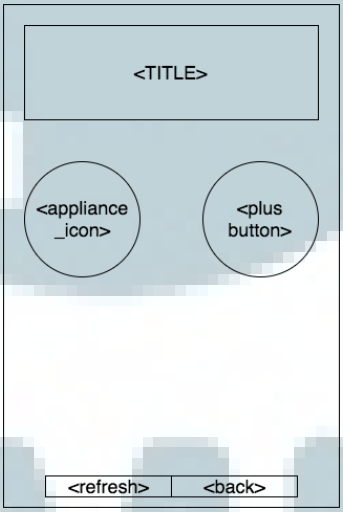
Tabel 3. 42 Tabel report pada Basis Data

#	Name	Type
1	reportid	int(11)
2	rtimestamp	datetime
3	applianceserial	varchar(15)
4	rprt	varchar(256)
5	commandid	int(11)

Tabel *report* dapat dilihat pada tabel 3.57. Tabel ini berisi 5 atribut, yaitu *reportid* sebagai *identifier* laporan, *rtimestamp* yang merupakan waktu perintah selesai dijalankan, *applianceserial* yang merupakan *foreign key* yang mengacu pada *applianceserial* pada tabel *appliancedetail*, *rprt* yang berisi laporan yang diterima, dan *commandid* yang juga merupakan *foreign key* yang mengacu pada *commandid* pada tabel *command*. Atribut *reportid* merupakan *primary key* pada tabel ini. Tabel ini digunakan untuk menyimpan laporan-laporan dari perangkat pintar.

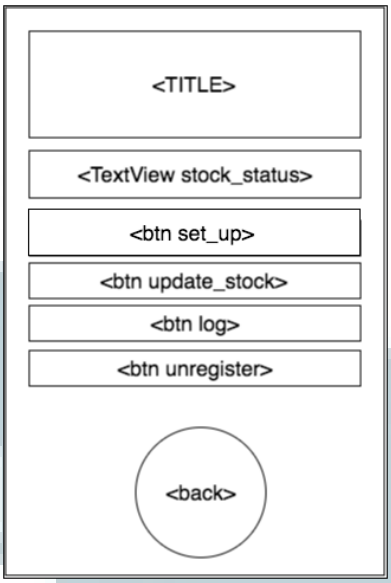
3.4.3 Rancangan Aplikasi Android

Secara keseluruhan, aplikasi Appliance Hub terdiri atas 8 *activity* umum, yaitu *activity sign in, sign up, main setting, appliance list, setting, account setting, change password,* dan *edit profile* [23]. Ketika pengguna masuk dalam aplikasi Appliance Hub, pengguna diharuskan melakukan *login*. Apabila pengguna belum memiliki akun , maka pengguna dapat melakukan *sign up* dengan mengisi form yang telah disediakan pada aplikasi. Setelah berhasil *login* ,pengguna akan masuk pada halaman *appliance list.* Pada halaman ini pengguna dapat melihat daftar *appliance* yang dimiliki dan dapat diakses oleh pengguna dan sebuah tombol untuk menambah *appliance* melalui *add appliance activity.* Desain halaman ini dapat dilihat pada gambar 3.3.



gambar 3. 6 Rancangan Tampilan Halaman Daftar Appliance

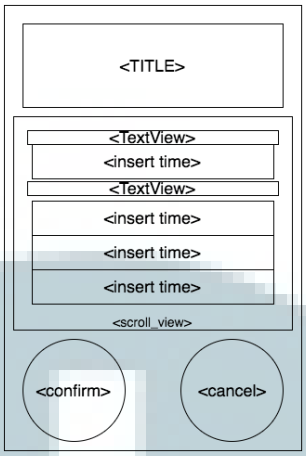
Ketika pengguna mengakses *smart medicine dispenser,* pengguna dapat mengakses 4 buah *activity* lainnya, yaitu *Set up, update stock, log,* dan *unregister.* Desain halaman ini dapat dilihat pada gambar 3.4.



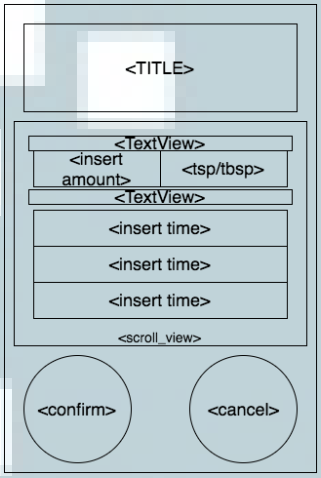
gambar 3. 7 Rancangan Tampilan Halaman Menu Smart Medicine Dispenser

Ketika pengguna menekan tombol “Set_Up”, pengguna akan dihadapkan pada halaman untuk mengatur obat pill yang nantinya akan dikeluarkan. Halaman ini bertugas menerima data jumlah pil yang dikeluarkan pada satu waktu beserta jadwal pemberian obat yang memiliki maksimal enam waktu. Apabila pengguna tidak menginginkan untuk melakukan mengatur obat pil , maka pengguna dapat langsung menekan tombol *next.*

Selanjutnya, pengguna akan dihadapkan pada halaman untuk mengatur obat cair yang nantinya akan dikeluarkan. Halaman ini bertugas menerima data jumlah obat cair yang nantinya akan dikeluarkan pada satu waktu beserta jadwal pemberian obat yang maksimal enam waktu. Apabila sebelumnya pengguna tidak mengisi pada halaman obat pil maka pengguna wajib mengisi jadwal pada halaman ini. Masukkan jumlah obat cair adalah per 5 ml. Setelah mengisi jadwal , pengguna dapat langsung menekan tombol *confirm.*

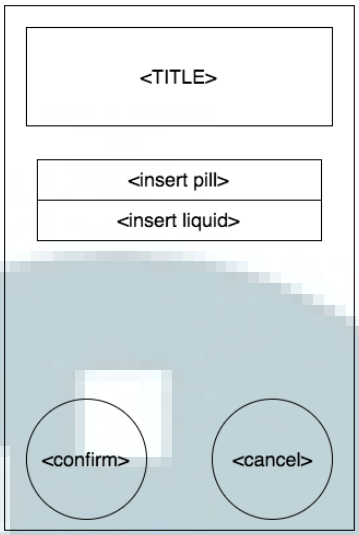


gambar 3. 8 Rancangan Tampilan Halaman Menu Pill



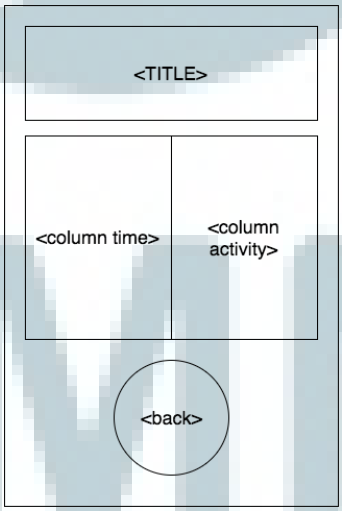
gambar 3. 9 Rancangan Tampilan Halaman Menu

Ketika pengguna mengakses menu “Update_Stock”, pengguna akan dihadapkan pada halaman dengan desain seperti gambar 3.7. Pada halaman ini, nilai jumlah obat pil dan obat cair yang baru saja dimasukkan diketikkan pada 2 buah *edit text* dan jika data sudah benar, pengguna tinggal menyentuh tombol “Confirm”. Pengguna perlu membiarkan tutup penyimpan pil sampai proses pembaruan selesai. Pada penyimpanan obat cair , pengguna hanya perlu meletakkan obat diatas sensor cahaya.



gambar 3. 10 Rancangan Tampilan Halaman Update Smart Medicine Dispense

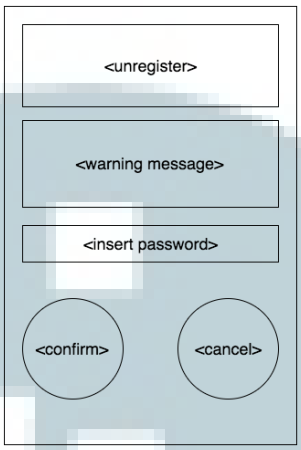
Ketika pengguna menekan tombol “Log”, pengguna akan dihadapkan pada halaman dengan desain seperti gambar 3.17. Pada halaman ini, informasi mengenai daftar laporan yang berasal dari perangkat pintar tersebut dapat diakses oleh pengguna. Daftar ini terdiri atas 2 kolom, yaitu waktu laporan diterima dan isi laporannya.



gambar 3. 11 Rancangan Tampilan Halaman Log

Ketika pengguna menekan tombol “Unregister”, pengguna akan dihadapkan pada halaman dengan desain seperti gambar 3.9. Pada halaman ini, pengguna

perlu mengisi kembali *password* akun pengguna untuk konfirmasi *unregister* pemilik perangkat tersebut.



gambar 3. 12 Rancangan Tampilan Halaman Unregister

3.4.4 Rancangan Arduino UNO

Program pada Arduino Uno bertugas untuk memproses dan mengeksekusi pesan yang diterima dari modul *Wi-Fi* ESP8266 melalui komunikasi *serial* dengan *baudrate* 9600. Program dimulai dengan inisialisasi variable dan mode pin(*input/output).* Selanjutnya program akan menunggu data *serial* dari modul Wi-Fi ESP8266. Data *serial* yang masuk akan ditampung pada variable "crecv" dan akan di-*parse* ke variable "injobj". Jika *parse* berhasil, maka program akan mencocokkan *value* pada name "msg" dengan 3 *value* yang dikenali oleh program, yaitu "medicine", "updateMedStock", dan "setApplianceSerial".

Jika *value* "msg" adalah "setApplianceSerial", maka program akan mengatur variabel "applianceserial" menjadi *value* dari serial number yang ada pada pesan tersebut. Setelah itu, program akan mengecek *alarm* dari RTC.

Jika *value* "msg" adalah "updateMedStock", maka program akan mengecek posisi tutup tempat obat pil dan juga posisi obat cair. Untuk pengecekan posisi

tempat obat pil digunakan sebuah *magnetic* sensor dan untuk obat cair digunakan sebuah sensor cahaya. Jika posisi obat terbuka maka program akan mengkonfirmasi pembaruan stok obat pil. Jika sensor cahaya mendeteksi ada obat diatasnya maka akan mengkonfirmasi pembaruan stok obat cair. Setelah itu , program akan mengecek alarm dari RTC.

Jika *value* “msg” adalah “medicine” , maka program akan melihat apakah ada jadwal untuk pil. Apabila terdapat jadwal, maka program akan mengatur alarm 1 untuk obat pil. Selanjutnya akan melihat jadwal untuk obat cair , apabila terdapat jadwal maka akan mengatur alarm 2. Setelah mengatur alarm 1 dan alarm 2 maka program akan melanjutkan untuk pengecekan alarm.

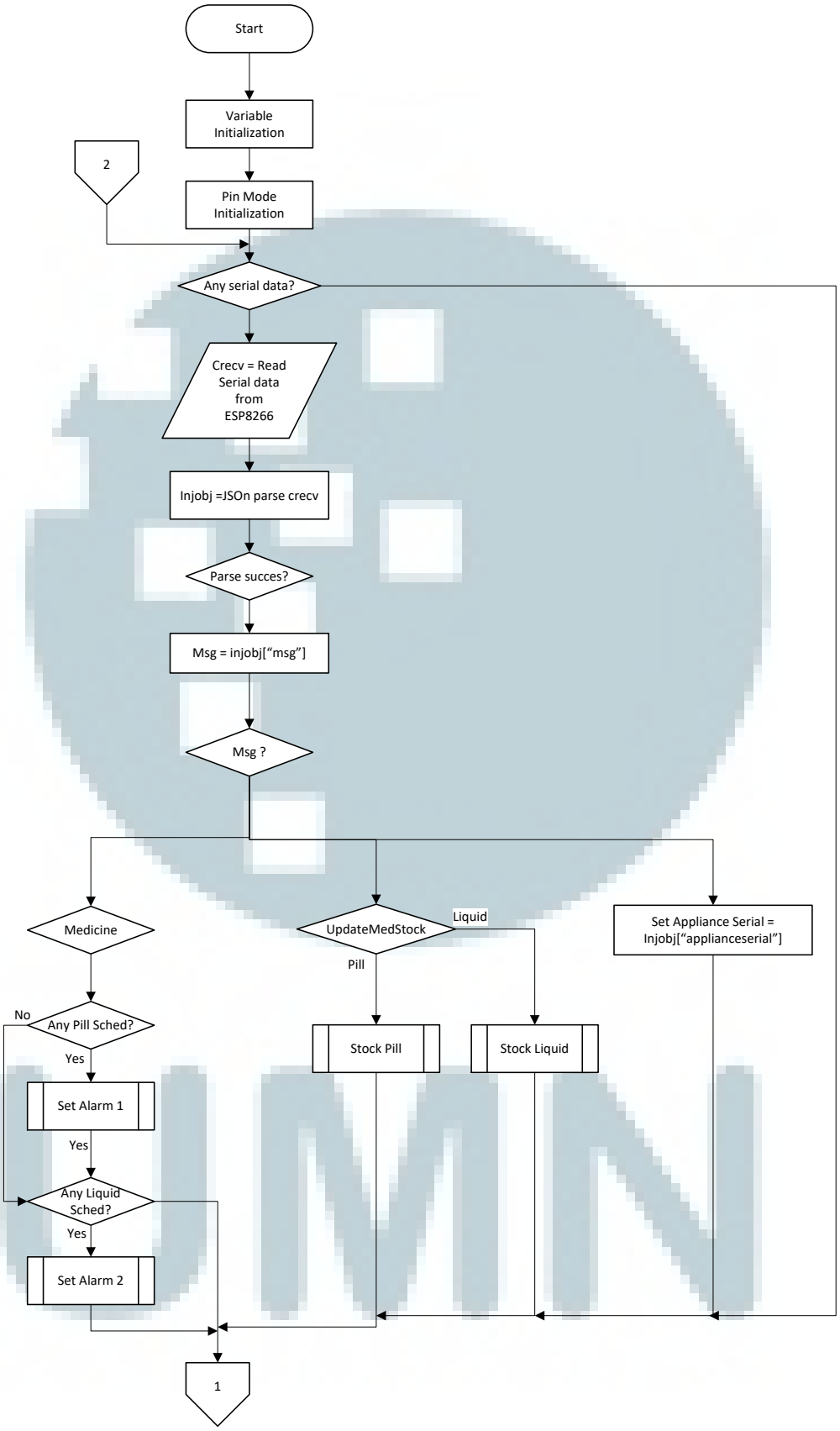
Pengecekan alarm dilakukan dengan melihat pada alarm 1 terlebih dahulu. Jika terdapat alarm satu maka akan dilakukan pengecekan kembali pada status pill. Apabila status pill adalah 0 maka akan merubah status pil menjadi satu dan akan mengeluarkan obat pil. Setelah selesai mengeluarkan obat maka mengatur *timeout* selama 15 menit untuk mengecek kembali status obat.

Setelah selesai mengatur *timeout* pada alarm 1 , selanjutnya program akan melakukan hal yang sama pada obat cair. Program akan melakukan pengecekan pada alarm 2. Jika terdapat alarm 2 , maka akan melakukan pengecekan pada status obat cair. Jika status obat cair adalah 0 maka akan melakukan pengaturan pada status obat cair. Setelah itu akan melakukan pengeluaran obat dan mengatur *timeout* selama 15 menit untuk mengecek kembali status obat.

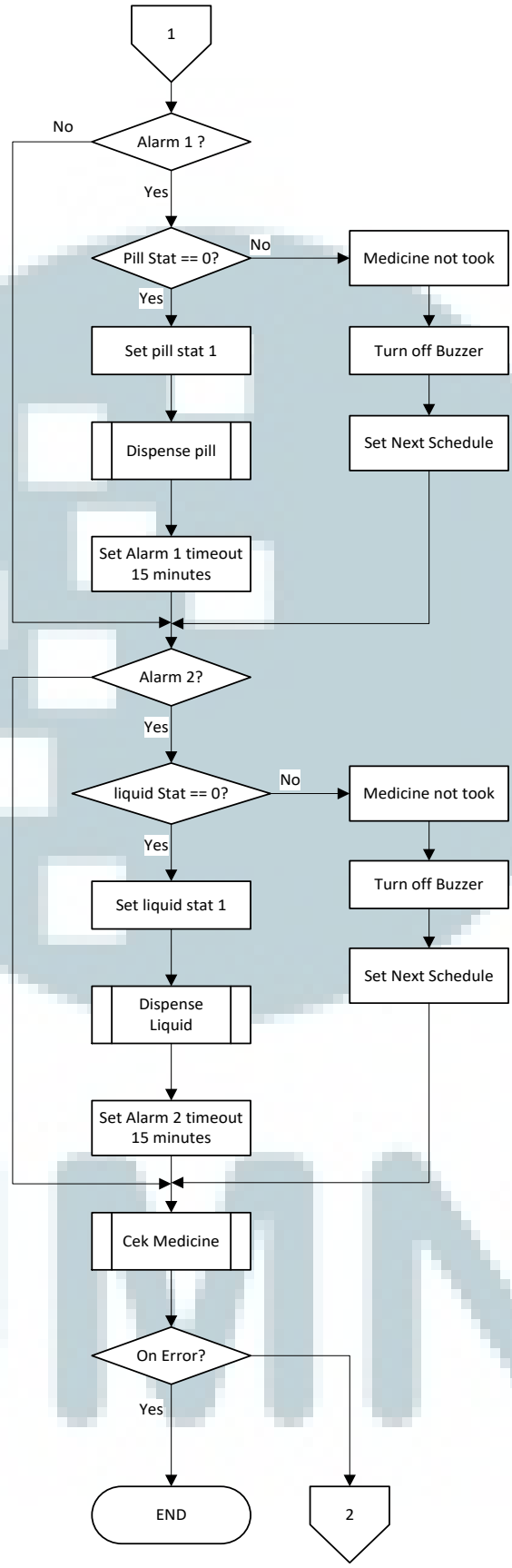
Setelah semua selesai mengeluarkan obat maka program akan melakukan pengecekan pada tempat masing masing obat. Jika obat belum diambil maka *buzzer* akan tetep menyala selama 15 menit. Jika obat telah diambil maka

buzzer akan dimatikan dan akan mengirimkan laporan pesan “medicineTook”. Jika dalam 15 menit obat tidak diambil maka akan mengirimkan pesan “medicineNotTook” ke server dan *buzzer* akan dimatikan. Setelah mematikan *buzzer* selanjutnya program akan mengatur untuk jadwal selanjutnya.

Jika tidak terjadi error selama program berjalan maka program akan terus melihat pesan selanjutnya dari server. Namun jika mengalami error maka program akan berhenti. Gambar 3.11 dan 3.12 adalah gambaran flowchart utama program Arduino. *Subroutine* pada diagram alur terlapir pada lampiran.



Gambar 3. 13 Flow Chart Arduino(1)



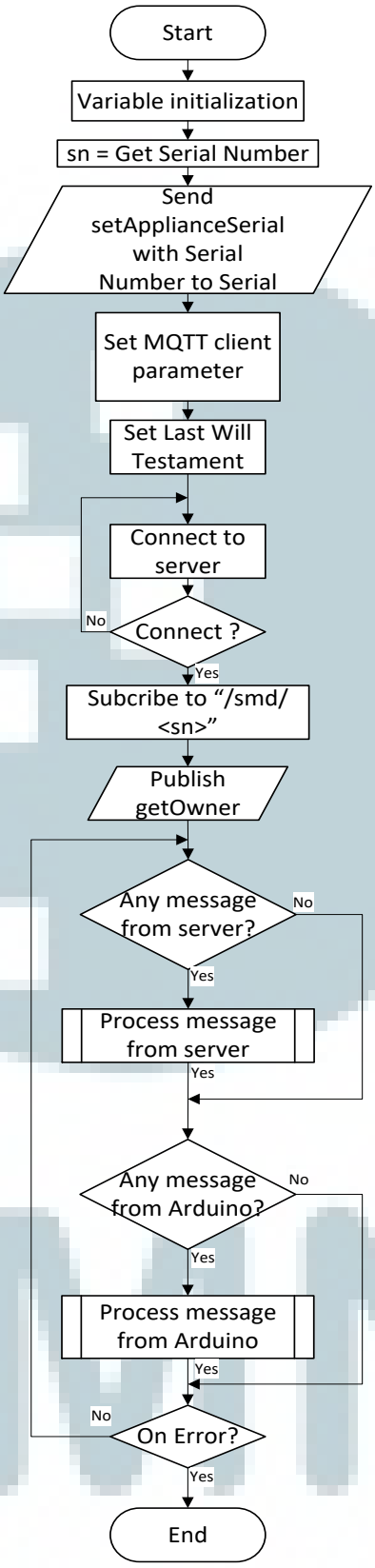
Gambar 3. 14 Flow Chart Arduino (2)

3.4.5 Rancangan Aplikasi NodeMCU

Program dimulai dengan inisialisasi variabel, kemudian program mengambil nomor serial dari alamat MAC perangkat dan akan mengirimkannya ke Arduino Uno melalui komunikasi serial. Setelah itu, program akan mengatur parameter *client* MQTT. Parameter yang diatur adalah ID, waktu *keepalive*, *username, password, dan* mengatur pesan *last-will-Testament.* Setelah selesai mengatur parameter awal, program akan mencoba melakukan koneksi ke *server*.

Parameter untuk melakukan koneksi ke server terdiri atas *host* dan *port server, security* (nilai 0 berarti koneksi tanpa *security*, sementara nilai 1 berarti dengan *security*), dan *auto-reconnect* (nilai 0 berarti program tidak akan melakukan *reconnect* secara otomatis ketika koneksi terputus atau gagal, sementara nilai 1 berarti program akan melakukan *reconnect* secara otomatis). Setelah berhasil terkoneksi, program akan melakukan *subscribe* ke topik "/smd/<serial_number>" dan meminta informasi pemilik perangkat ke *server* dengan mengirimkan pesan "getOwner".

Setelah *server* berhasil mem*publish* "getOwner", program akan menunggu pesan dari *server* dan Arduino untuk diolah berdasarkan pesannya masing masing. Gambaran lebih detail mengenai pengolahan masing-masing pesan dapat dilihat pada Lampiran 1. NodeMCU.



Gambar 3. 15 Flow Chart NodeMCU