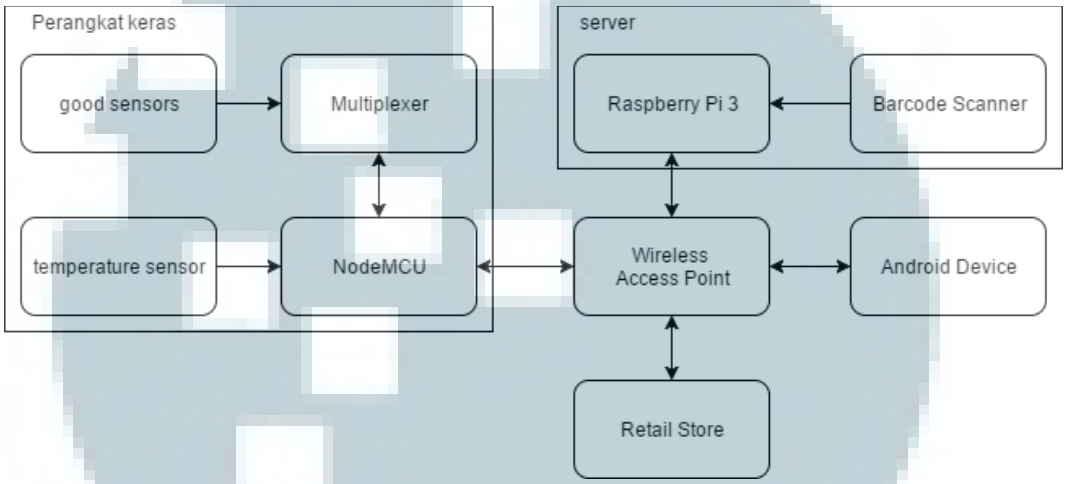
BAB III

RANCANGAN SYSTEM

3.1. Diagram Blok

Perancangan alat secara keseluruhan tergambar melalui blok diagam berikut.



Gambar 3. 1 Blok diagram sistem secara keseluruhan

Dari gambar 3.1 terlihat komunikasi antar perangkat, baik perangkat keras, *server*, android, maupun toko retail melalui jaringan internet.

3.2. Fitur kulkas pintar

Fitur kulkas pintar adalah sebagai berikut :

a. Kulkas dapat mengetahui posisi barang yang dimasukkan ke dalam kulkas secara urut satu per satu.
b. Kulkas ini dapat mendeteksi barang yang dipindahkan jika perpindahan dilakukan satu per satu.
c. Kulkas dapat mendeteksi barang apa saja yang diambil oleh pengguna.
d. Kulkas dapat memesan sendiri barang yang jumlahnya kurang dari batas yang ditentukan oleh pengguna.

e. Kulkas dapat memberitahu pengguna ketika terjadi kenaikan suhu yang signifikan pada kulkas.
f. Pengguna dapat melihat daftar barang yang tersisa dalam kulkas
g. Pengguna dapat mengganti tanggal kadaluarsa barang dalam kulkas.
h. Pengguna dapat mengatur jumlah minimal dan jumlah normal dari setiap barang yang telah terdaftar di dalam database.
i. Pengguna dapat memesan barang ke penyedia jasa retail melalui aplikasi smartphone kulkas pintar
j. Pengguna dapat melihat rekaman transaksi yang pernah dilakukan
k. Pengguna dapat menghidupmatikan fitur *auto buy* serta menentukan interval pembelian.
l. Barang yang akan dibeli oleh kulkas pintar secara otomatis adalah barang yang jumlah barangnya kurang dari batas minimal yang telah ditentukan oleh pengguna. Jumlah barang yang dibeli adalah selisih dari jumlah barang normal dengan jumlah barang saat proses pemesanan berlangsung.

3.3. Perancangan sistem

Perancangan sistem kulkas pintar terdiri dari tiga tahap, yaitu :

3.3.1. Perancangan Perangkat Kulkas Pintar

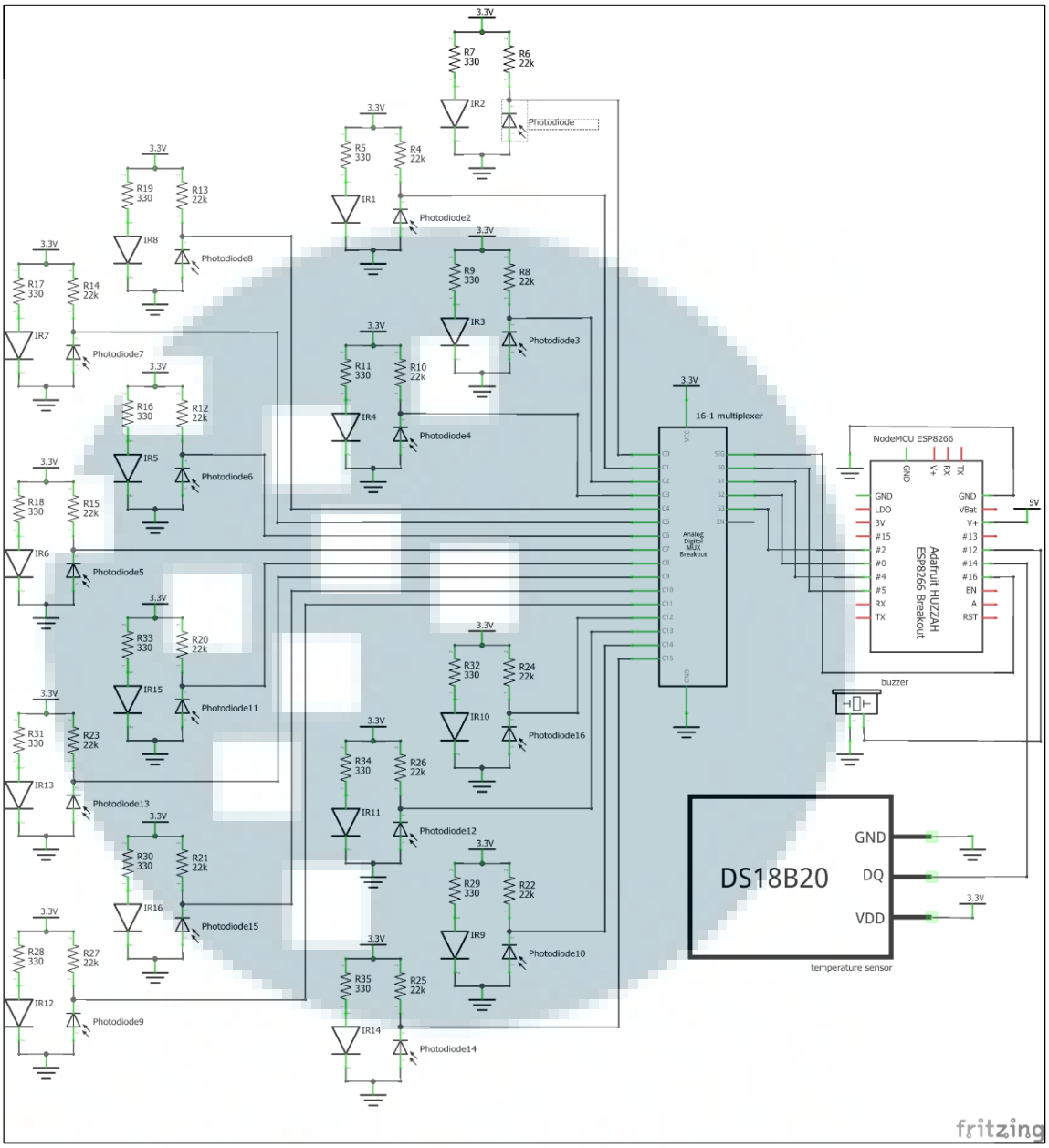
Perancangan perangkat Kulkas Pintar terbagi menjadi dua bagian, yaitu perancangan perangkat keras dan perancangan perangkat lunak.

A. Perancangan Perangkat Keras Kulkas Pintar

Perangkat keras dari Kulkas Pintar terdiri dari komponen sebagai berikut :

a. ESP8266 NodeMCU DevKit sebagai *microcontroller* yang membaca data dari sensor dan mengirimnya ke *home server*.
b. *Photodioda* khusus infra merah dan inframerah 16 pasang sebagai sensor untuk mendeteksi keberadaan barang.
c. Resistor
d. *Buzzer*
e. Sensor suhu DS18B20 untuk mengetahui temperatur pada kulkas.
f. *Multiplexer* CD74HC4067 16 *Channel* untuk menghemat penggunaan *port* pada nodeMCU.
g. Papan PCB sebagai tempat menyambung berbagai komponen
h. Nampan dua buah sebagai tempat melekatnya sensor
i. Modul daya untuk menyediakan listrik dengan tegangan 5V dan 3.3V

Gambar 3.2 adalah diagram skematik dari perangkat keras kulkas pintar. Sumber tegangan yang digunakan berasal dari sebuah modul daya dengan masukan USB 5V sehingga mempermudah user dalam mencari sumber tenaga. Modul ini menghasilkan dua keluaran tegangan yang berbeda, yakni 3.3V dan 5V. Tegangan 5V digunakan untuk sumber daya NodeMCU. Sedangkan sumber tegangan 3.3V digunakan untuk sumber daya komponen lainnya seperti IR, *photodioda*, sensor suhu DS18B20, *buzzer*, dan *multiplexer*. Penggunaan tegangan 3.3V pada semua komponen dikarenakan tegangan masukan dan keluaran yang dapat diterima oleh NodeMCU hanya 3.3V.



Gambar 3. 2 Diagram skematik perangkat keras

Sensor barang yang digunakan terdiri dari empat buah komponen, yaitu inframerah, *photodioda* khusus inframerah, resistor 330 ohm, dan resistor 22k ohm. Penggunaan inframerah dan photodioda ini bertujuan agar nilai pembacaan sensor tidak mudah terpengaruh oleh cahaya luar yang diterima. Resistor 330 ohm yang digunakan di antara inframerah dan VCC bertujuan agar cahaya yang dihasilkan inframerah masih

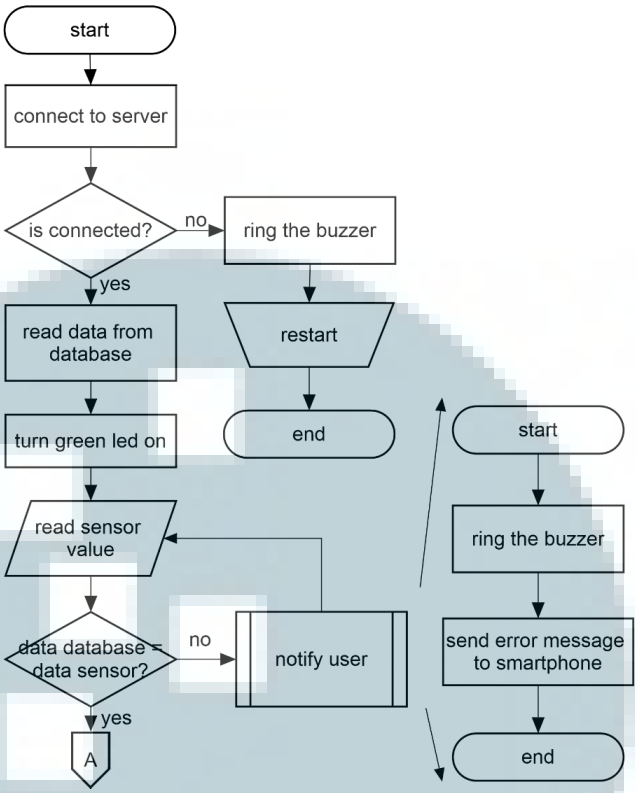
cukup terang dan dapat dipantulkan oleh objek di atasnya. Sedangkan hambatan 22k ohm di antara VCC dan photodioda digunakan untuk meningkatkan kepekaan photodioda agar photodioda tetap dapat mendeteksi benda dengan warna gelap.

Pada rangkaian sensor suhu DS18B20 terdapat sebuah resistor dengan nilai resistansi 4k7 ohm di antara VCC dan pin data. Resistor ini bertujuan sebagai resistor *pull up* agar hasil pembacaan sensor dapat dibaca oleh NodeMCU.

B. Perancangan Perangkat Lunak Kulkas Pintar

Perangkat lunak pada kulkas pintar berada pada NodeMCU dan dibuat dengan menggunakan Arduino IDE. Program ini terdiri dari dua bagian utama, yaitu inisialisasi dan *looping*. Alur kerja dari proses inisialisasi tergambar pada gambar 3.3 dan alur kerja dari proses *looping* tergambar dari gambar 3.5.

Pada saat pertama kali NodeMCU dijalankan, maka NodeMCU akan otomatis mencoba terhubung dengan *Access Point* dan *home server*. Apabila koneksi gagal tersambung dengan *home server*, maka NodeMCU akan membunyikan *buzzer* sebagai tanda bahwa terjadi kesalahan pada koneksi. Selanjutnya, NodeMCU harus di-*restart* secara manual agar NodeMCU dapat mencoba terhubung dengan *home sever* kembali. Apabila koneksi telah terjalin dengan baik, maka selanjutnya NodeMCU akan menghidupkan LED hijau lalu meminta data dari home *server* mengenai lokasi mana saja yang telah terisi.



Gambar 3. 3 Diagram alir NodeMCU bagian 1

Respon yang dikirim dari *home server* berubah sebuah JSON dengan format seperti gambar 3.4.

```
{
 "kulkasID":"1",
 "location":
  [
   X0,X1,x2,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12,X13,X14,X15
  ]
}
```

Gambar 3. 4 Format JSON untuk mendapat lokasi barang di *server*

kulkasID digunakan jika terdapat lebih dari satu kulkas dalam sebuah rumah sehingga dapat dipastikan bahwa data yang mereka terima adalah sesuai dengan kulkas. Sedangkan *location* berisi *array* yang terdiri dari 16 nilai, yakni X_0 hingga X_{15} yang melambangkan nilai

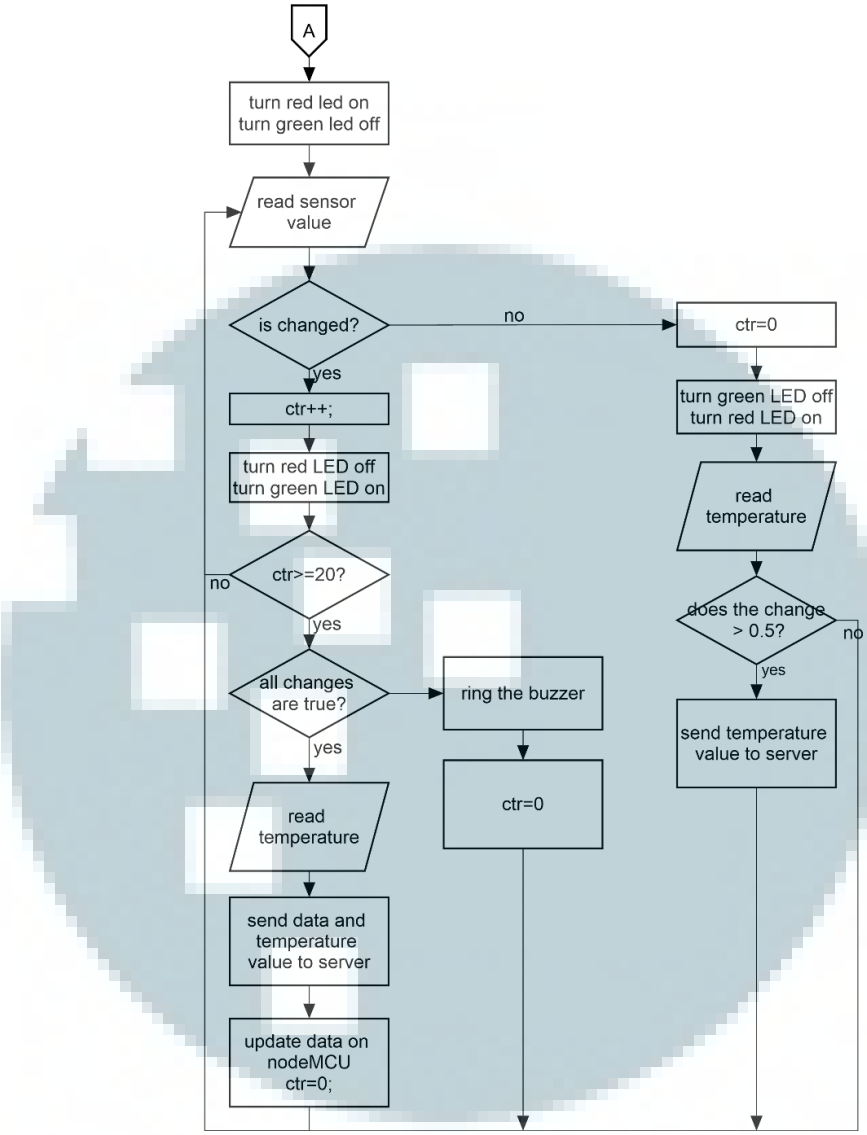
dari setiap sensor. "0" jika tidak terdapat benda dan "1" jika terdapat benda pada lokasi sensor tersebut.

Setelah mendapat data dari *server*, NodeMCU lalu membaca sensor dan membandingkan hasil pembacaan tersebut dengan data yang diperoleh dari *server*. Jika terdapat perbedaan, maka NodeMCU akan memberikan notifikasi kepada pengguna, berupa bunyi *buzzer* dan mengirim pesan *error* ke *smartphone* pengguna.

Apabila tidak terjadi kesalahan atau kesalahan yang terjadi telah diperbaiki oleh pengguna, maka perangkat lunak akan lanjut ke tahap selanjutnya.

Bagian kedua adalah *looping*. pada bagian ini dimulai dengan menghidupkan LED merah dan mematikan LED hijau sebagai tanda bahwa semua proses inisialisasi telah terselesaikan dengan baik. Proses berikutnya yaitu membaca sensor barang dan mencocokannya dengan hasil pembacaan sebelumnya.

Jika tidak terjadi perbedaan, maka program membaca nilai temperatur dan mengubah nilai *ctr* menjadi 0 dan mematikan LED hijau. Proses berikutnya adalah membaca temperatur lalu membandingkannya dengan nilai temperatur sebelumnya. Jika selisihnya mencapai 0.5°C maka NodeMCU akan mengirim data tersebut ke *home server* dan menyimpan nilai tersebut sebagai nilai temperatur sebelumnya. Apabila selisihnya tidak mencapai 0.5°C, maka program akan kembali membaca sensor barang.



Gambar 3. 5 Diagram alir NodeMCU bagian 2

Sedangkan jika terjadi perbedaan nilai pembacaan sensor barang, maka program akan menambah nilai variabel *ctr* dan menghidupkan LED hijau sebagai tanda bahwa sistem mendeteksi perubahan pembacaan sensor. Apabila nilai *ctr* belum mencapai 20, maka program akan kembali membaca sensor barang. Sedangkan apabila nilai *ctr* telah mencapai 20, maka nodeMCU akan menentukan perubahan yang terjadi valid atau tidak. Jika tidak valid, maka NodeMCU akan

membunyikan *buzzer* sebagai tanda bahwa sesuatu yang salah terjadi. Apabila perubahan yang terjadi bersifat valid, maka NodeMCU akan membaca sensor temperatur lalu mengirimkan semua hasil pembacaan sensor ke *home server*. Setelah data dikirim, maka nodeMCU memperbaharui data barang dan temperatur yang mereka miliki serta mengatur nilai ctr menjadi 0 kembali.

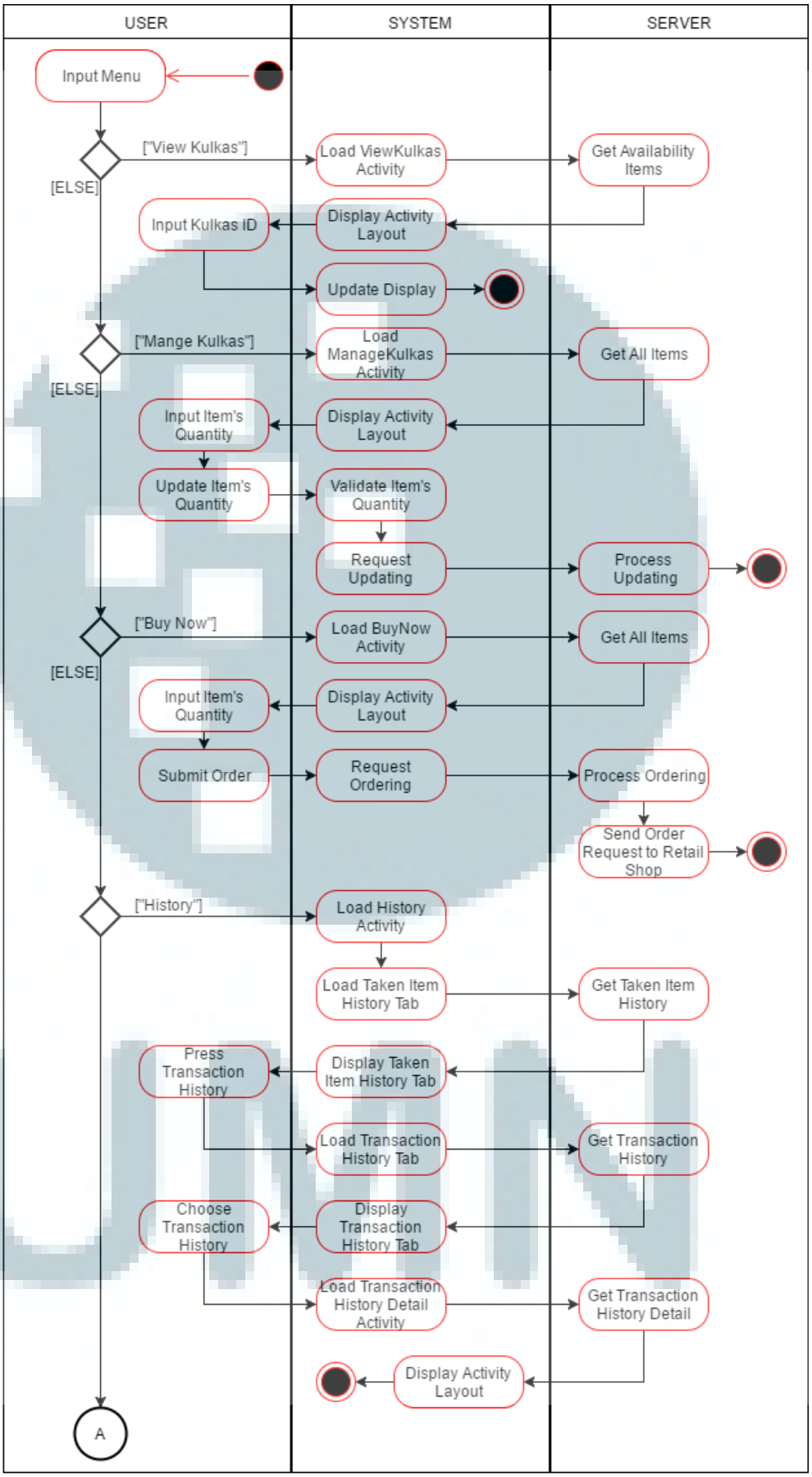
Data yang kirim oleh NodeMCU ke *home server* berubah sebuah JSON dengan format seperti gambar 3.6.

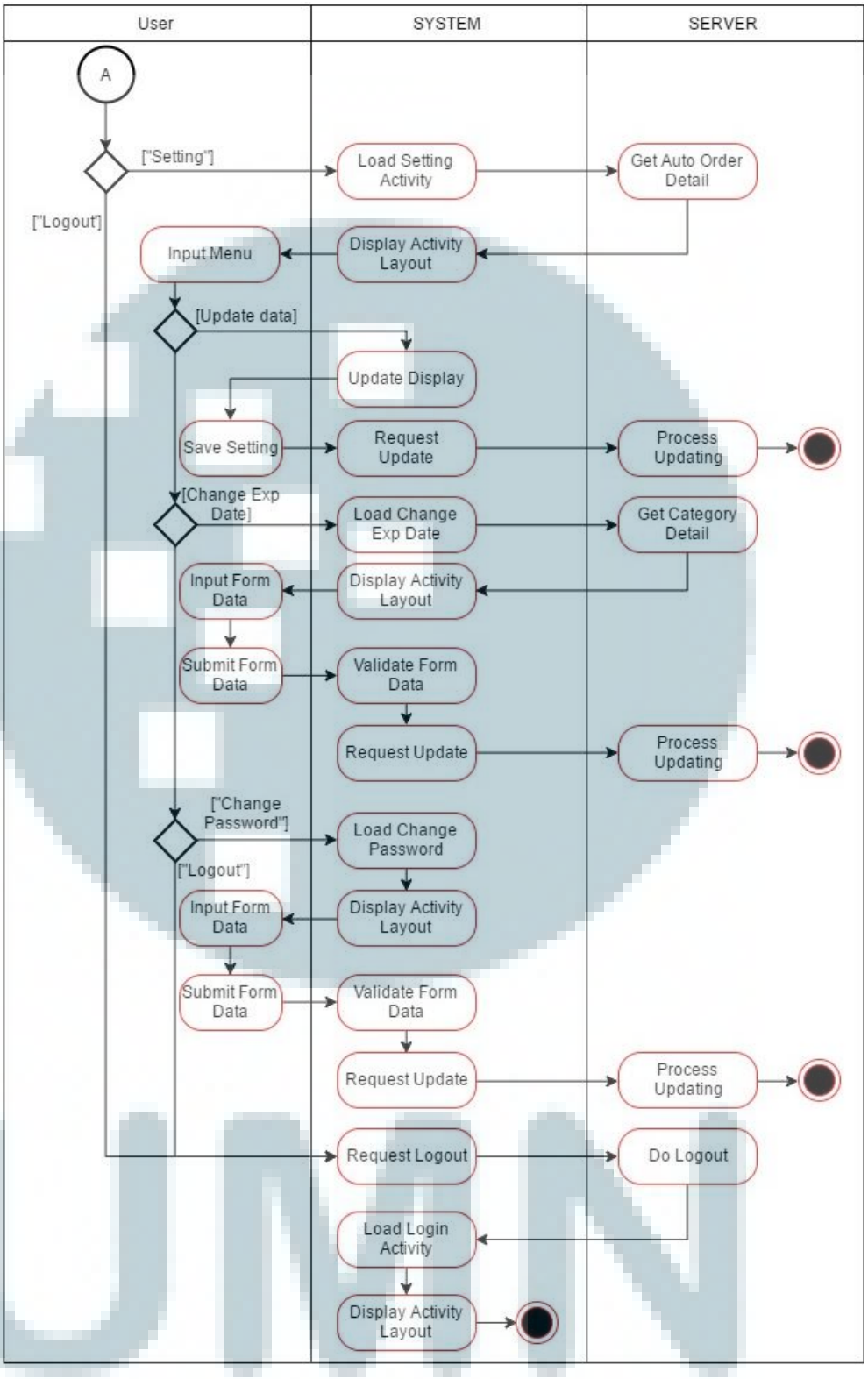
```
{
 "kulkasID":"1",
 “temp”:Xtemp
 "location":
  [
   X0,X1,x2,X3,X4,X5,X6,X7,X8,X9,X10,X11,X12,X13,X14,X15
  ]
}
```

Gambar 3. 6 Format JSON yang dikirim ke *server* untuk memperbarui data

3.3.2.Perancangan Aplikasi pada Perangkat Cerdas Android

Aplikasi android merupakan *front end* yang menghubungkan pengguna dengan sistem kulkas pintar. Untuk mengetahui berbagai macam fitur yang terdapat pada aplikasi, kita dapat mlihat diagram aktivitas pada gambar 3.7.

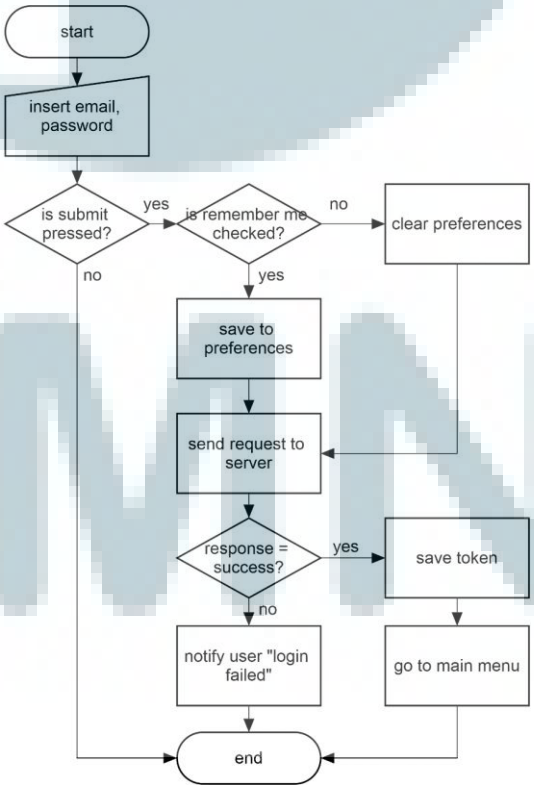

USER
SYSTEM
SERVER
Input Menu
["View Kulkas"]
[ELSE]
Load ViewKulkas Activity
Get Availability Items
Input Kulkas ID
Display Activity Layout
Update Display
["Mange Kulkas"]
[ELSE]
Load ManageKulkas Activity
Get All Items
Input Item's Quantity
Display Activity Layout
Update Item's Quantity
Validate Item's Quantity
Request Updating
Process Updating
["Buy Now"]
[ELSE]
Load BuyNow Activity
Get All Items
Input Item's Quantity
Display Activity Layout
Submit Order
Request Ordering
Process Ordering
Send Order Request to Retail Shop
["History"]
Load History Activity
Load Taken Item History Tab
Get Taken Item History
Press Transaction History
Display Taken Item History Tab
Load Transaction History Tab
Get Transaction History
Choose Transaction History
Display Transaction History Tab
Load Transaction History Detail Activity
Get Transaction History Detail
Display Activity Layout
A



Gambar 3. 7 Diagram aktivitas aplikasi android

A. Login

Login digunakan untuk melakukan autentifikasi terhadap pengguna agar aplikasi tidak diakses oleh sembarang orang. Pada menu ini, pengguna harus mengisi email serta kata sandi yang mereka miliki untuk dapat mengakses menu lain yang terdapat pada aplikasi ini. Untuk mempermudah pengguna, disediakan fitur *remember me* yang akan menyimpan email dan kata sandi pengguna dalam *preferences* aplikasi. Apabila pengguna telah melakukan *login* lalu keluar dari aplikasi tanpa melakukan *logout* terlebih dahulu, maka pada saat pengguna membuka aplikasi kulkas pintar kembali, maka aplikasi akan langsung meneruskan ke menu utama tanpa harus melakukan autentifikasi ulang. Alur kerja dari menu ini tergambar melalui gambar 3.8.

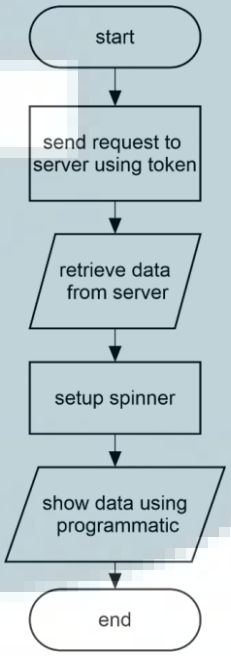


Gambar 3. 8 Diagram alir proses login android

B. Melihat barang yang berada di dalam kulkas

Fitur ini digunakan oleh oleh pengguna ketika hendak melihat benda apa saja yang berada di dalam kulkas mereka. Informasi yang disediakan oleh fitur ini adalah nama barang, tanggal kadaluarsa, serta jumlah dari setiap barang yang ada.

Fitur ini hanya menampilkan informasi tanpa memerlukan masukkan. Alur kerja dari fitur ini tergambar pada gambar 3.9.



Gambar 3. 9 Diagram alir untuk melihat barang pada android

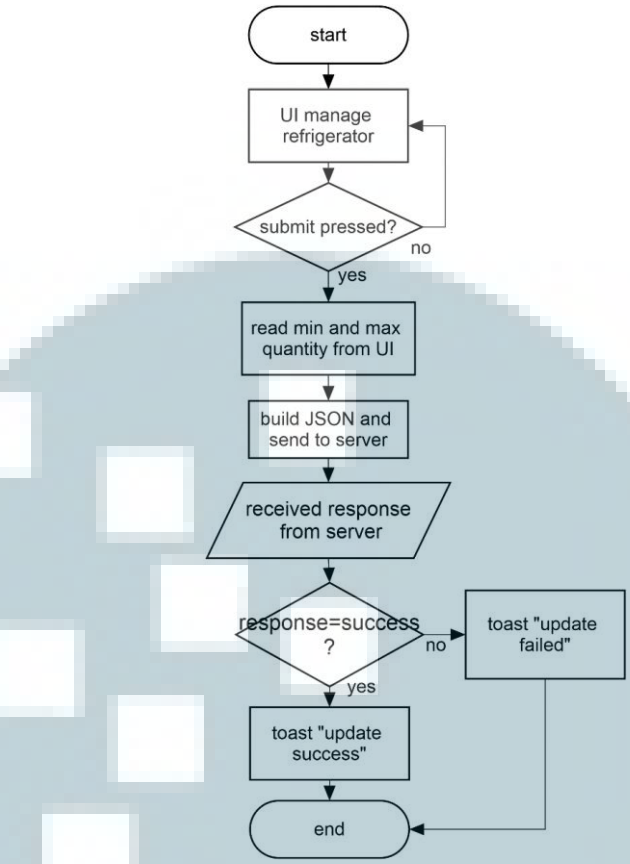
Pada saat menu dipanggil, maka aplikasi akan melakukan request ke server menggunakan *async task*. Setelah mendapat respons, lalu aplikasi mengelompokan setiap data yang diterima dan menampilkannya ke layar menggunakan programmatic layout. Pada menu ini juga terdapat sebuah *spinner* yang dapat digunakan oleh pengguna untuk memfilter data apa yang hendak ditampilkan di layar.

Filter dilakukan berdasarkan kulkas yang ada, sehingga barang yang ada ditampilkan per kulkas.

C. Mengatur batas ambang dari setiap item

Menu ini dipanggil pengguna ketika pengguna ingin melakukan konfigurasi dari setiap barang yang ada di dalam *database*. Melalui menu ini pengguna dapat menentukan batas minimum dan batas normal dari setiap barang. Batas minimum dan batas normal ini digunakan ketika fitur *auto buy* diaktifkan oleh pengguna. Cara kerja dari fitur *auto buy* adalah memesan barang yang jumlahnya kurang dari batas minimum. Jumlah barang dipesan adalah selisih antara jumlah barang saat itu dengan batas normal yang telah ditentukan oleh pengguna.

Alur kerja untuk mendapatkan informasi dari *server* ketika menu ini dipanggil hampir sama dengan diagram alir pada gambar 3.9, namun tidak melakukan setup spinner. Untuk mengatur jumlah dari setiap item, maka pengguna dapat menggunakan tombol "+" dan tombol "-" yang terletak di samping batas ambang. Setelah selesai, pengguna harus menekan tombol "submit" untuk menyimpan konfigurasi yang telah dilakukan di dalam *database home server*. Alur kerga aplikasi saat tombol "submit" ditekan tergambar pada gambar 3.10. Format JSON yang dikirim ke server memiliki format seperti gambar 3.11.



Gambar 3. 10 Diagram alir untuk memperbarui batas ambang barang

```
{
"token":"token",
"items":
  [
   {
    “itemID”:”itemID”,
    “min”:”min”,
    “max”:”max”
   }
  ]
}
```

Gambar 3. 11 Format JSON untuk memperbarui pengaturan kulkas

D. Membeli barang secara langsung

Menu ini digunakan ketika pengguna ingin melakukan pembelian lebih secara *real time*. Ketika menu ini dijalankan pertama kali, maka aplikasi akan melakukan *request* ke *server* untuk mendapat daftar barang yang bisa dibeli secara *async task*. Barang yang dapat dibeli

adalah barang yang telah terdaftar sebelumnya di dalam *database home server*. Respons yang didapat dari *server* berupa sebuah JSON dengan format seperti pada gambar 3.12.

```
{
 "token":"token",
 "items":
  [
   {
    “itemID”:”itemID”
    “qty”:”qty”
   }
  ]
}
```

Gambar 3. 12 Format JSON dari android ke server untuk membeli barang

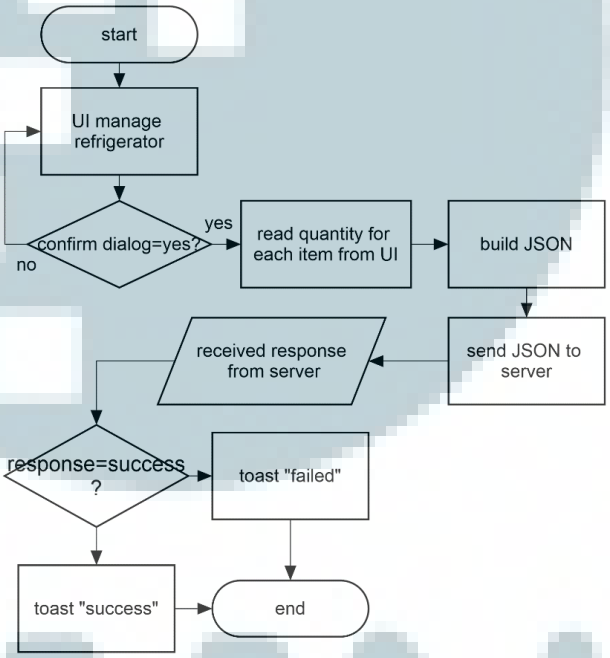
Cara kerja menu ini dengan mengirim *request* ke *server* menggunakan *token*, lalu menampilkan JSON yang diterima ke UI dengan menggunakan programmatic layout.

Setelah menu telah menampilkan semua daftar barang yang dapat dibeli oleh pengguna, maka pengguna dapat menekan tombol ”+” atau “-” untuk menentukan jumlah dari setiap barang yang akan dibeli. Setelah selesai, maka pengguna wajib menekan tombol “Submit”. Setelah tombol “Submit” ditekan, maka akan muncul dialog untuk memastikan bahwa pengguna benar-benar ingin melakukan pemesanan. Apabila pengguna tidak yakin, maka pengguna dapat menekan tombol “No” sedangkan jika pengguna sudah yakin, maka dapat menekan tombol “Yes”. Ketika tombol “Yes” ditekan, maka aplikasi akan membentuk sebuah JSON dengan format seperti gambar 3.13.

```
{
"email":"email",
"password":"password",
 "items":
  [
   {
    “itemID”:”  itemID”
    “qty”:”qty”
   }
  ]
}
```

Gambar 3. 13 Format JSON dari android untuk membeli barang

Alur kerja program ketika tombol “Yes” pada *form dialog* di tekan tergambar pada gambar 3.14.



Gambar 3. 14 Diagram alir untuk mengirim pesanan ke server

E. Melihat riwayat pembelian

Menu ini digunakan ketika pengguna ingin melihat riwayat pembelian yang pernah dilakukan. Informasi yang ditampilkan oleh menu ini meliputi ID transaksi, tanggal pemesanan, dan banyak jenis barang yang dibeli. Ketika menu ini dijalankan pertama kali, maka aplikasi akan melakukan *request* ke *server* secara *async task*. Setelah

mendapat respons dari *server* berupa sebuah JSON dengan format seperti pada gambar 3.15, aplikasi lalu memunculkan informasi tersebut ke tampilan layar dengan menggunakan *programmatic layout*. Ketika salah satu daftar ditekan lama, maka detail dari transaksi tersebut dimunculkan.

```
{
"token":"token",
"month":"mmmm",
"year":"yyyy",
"items":
  [
   {
    "transactionID":"itemID",
    "orderDate":"orderDate",
    "time":"mmmm yyyy",
    "qty":"qty"
   }
  ]
}
```

Gambar 3. 15 Format JSON untuk melihat daftar riwayat ransaksi

Pada menu ini terdapat sebuah spinner yang mempermudah pengguna untuk memfilter daftar riwayat transaksi berdasarkan bulan transaksi itu dilakukan. Transaksi yang ditampilkan pada menu ini adalah transaksi yang terjadi dalam kurun waktu tiga bulan terakhir.

Cara kerja dari menu detail transaksi tak jauh berbeda dari menu riwayat transaksi, yakni melakukan request ke server secara *async task* dengan mengirim ID transaksi dari transaksi yang dipilih. Setelah mendapat respons dari *server* berupa sebuah JSON dengan format seperti pada gambar 3.16, maka aplikasi menampilkan informasi tersebut secara *programmatic*. Informasi yang ditampilkan pada menu ini adalah tanggal transaksi, nama barang, serta jumlahnya.

```
{
"token":"token",
 "items":
   [
    {
     “productName”:”product_name”,
     “qty”:”qty”
    }
   ]
}
```

Gambar 3. 16 Format JSON untuk mengetahui detail riwayat transaksi

F. Melihat riwayat benda yang telah diambil dari kulkas

Menu ini digunakan ketika pengguna ingin melihat daftar riwayat benda yang telah diambil dari kulkas. Melalui menu ini, pengguna mendapat informasi mengenai nama, tanggal masuk, tanggal keluar, serta jumlah barang yang diambil. Cara kerja menu ini tak jauh berbeda dari menu lainnya, yakni melakukan *request* ke *server* secara *async task* dan mendapat respons sebuah JSON dengan format seperti pada gambar 3.17. Setelah menerima JSON, aplikasi lalu menampilkan informasi tersebut ke layar menggunakan *programmatic layout*.

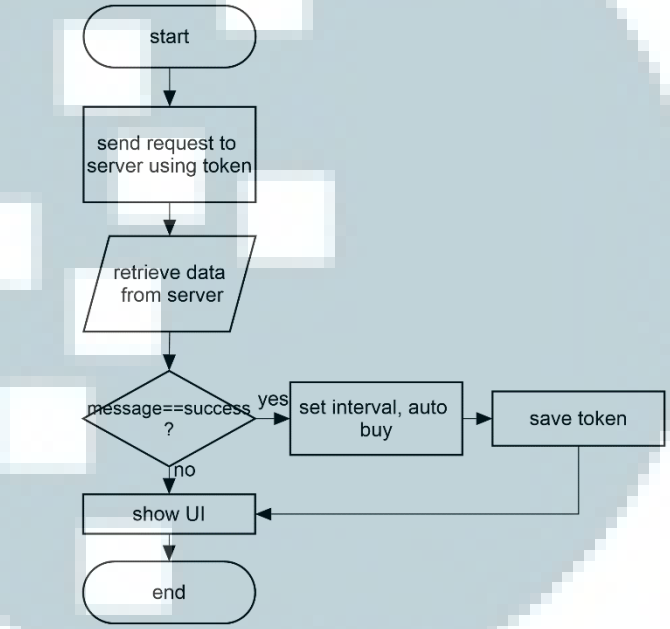
```
{
"token":"token",
"month":"mmmm",
"year":"yyyy",
"items":
  [
   {
    “itemName”:”itemID”,
    “total”:”total”,
    “date_in”:”date_in”,
    “date_out”:”date_out”,
    “date”:”mmmm yyyy”,
    “imageUrl”:”imageUrl”
   }
  ]
}
```

Gambar 3. 17 Format JSON untuk mengetahui riwayat benda yang diambil

G. Melakukan setting pada aplikasi

Menu ini digunakan oleh pengguna untuk melakukan berbagai pengaturan aplikasi kulkas pintar, antara lain menghidup-matikan fitur *auto* buy, mengatur interval pemesanan, mengatur tanggal kadaluarsa, mengganti kata sandi, serta melakukan *logout*.

Alur kerja dari menu ini tergambarkan pada gambar 3.18.



Gambar 3. 18 Diagram alir ketika menu setting dibuka

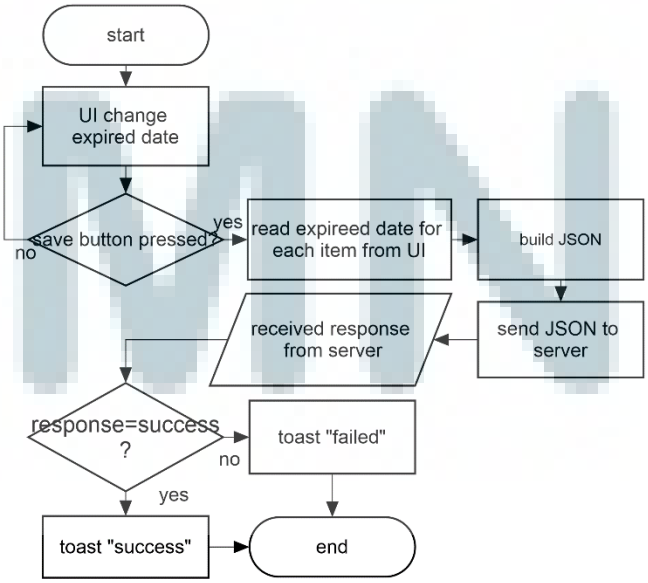
Langkah pertama ketika menu ini diakses adalah melakukan *request* secara *async task* ke *server* untuk mendapatkan informasi mengenai *auto order* serta interval pemesanan. Server memberi response ke aplikasi berupa sebuah JSON dengan format seperti gambar 3.19.

```
{
 "token":"token",
 "autoBuy":"autoBuy",
 "period":"period",
 "message":"message"
}
```

Gambar 3. 19 Format JSON untuk menu setting

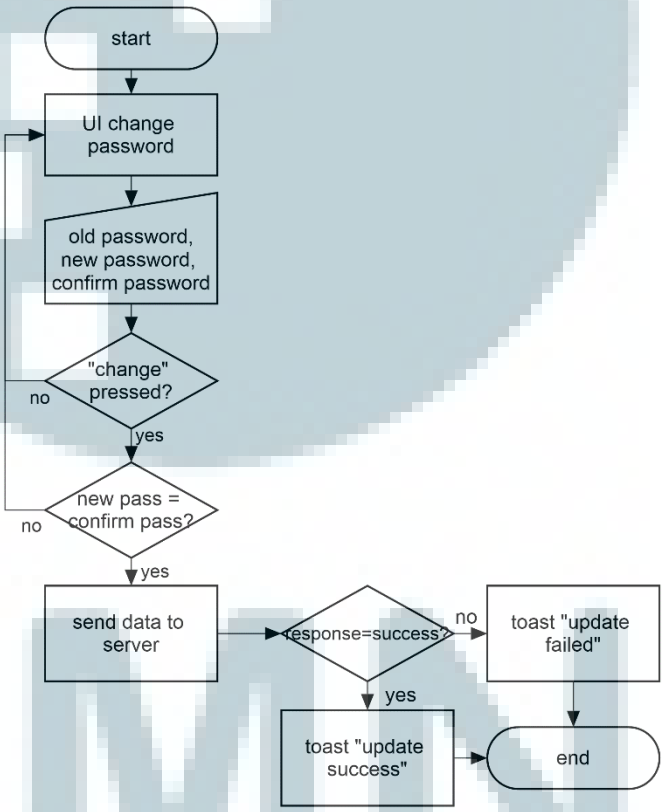
Informasi yang didapat lalu ditampilkan ke layar. Jika pengguna melakukan perubahan pengaturan dan ingin menyimpannya secara permanen di dalam *database home server*, maka pengguna harus menekan tombol "Save". Jika tombol "Save" ditekan, maka aplikasi akan membaca pengaturan yang dilakukan oleh pengguna lalu mengirimkannya ke *home server* melalui HTTP POST.

Pengaturan berikutnya adalah melakukan pengaturan tanggal kadaluarsa dari setiap barang yang ada di dalam kulkas. Pengaturan ini dapat diakses dengan menekan tombol "Change Expired Date". Menu ini akan menampilkan semua barang yang terdapat di dalam kulkas, termasuk posisi, tanggal masuk, dan tanggal kadaluarsa. Dengan menekan salah satu barang yang ada, pengguna dapat mengganti tanggal kadaluarsa melalui *dialog* yang muncul. Setelah selesai, pengguna dapat menyimpan pengaturan tersebut dengan menekan tombol "Save". Alur kerja aplikasi ini saat tombol "Save" ditekan terlihat pada gambar 3.20.



Gambar 3. 20 Diagram alir untuk mengganti tanggal kadaluarsa

Jika pengguna ingin mengganti kata sandi, maka pengguna dapat menekan tombol "Change Password". Setelah masuk ke menu *change password*, pengguna harus mengisi semua *field* yang tersedia lalu menekan tombol "Change" untuk mengganti sandi mereka. Jika tombol "Change" ditekan, maka aplikasi akan mengirimkan data ke server melalui HTTP POST *request*. Alir kerja dari menu ini dapat dilihat pada gambar 3.21. Apabila proses penggantian kata sandi berhasil, maka aplikasi akan memunculkan toast "Update success!", sedangkan jika proses gagal, maka aplikasi akan memunculkan toast "Update failed!".



Gambar 3. 21 Diagram Alir untuk mengganti password

H. Logout

Fitur *logout* digunakan ketika pengguna ingin mengeluarkan akun mereka dari aplikasi. Fitur *logout* dapat diakses melalui tombol pada

menu *setting* atau melalui menu pada *navigation bar*. Ketika *logout* dilakukan, maka aplikasi akan mengirim *token* ke *server*. Jika *token* yang dikirim salah, maka pengguna diharuskan mengisi kata sandi untuk autentifikasi bahwa aplikasi ini tidak sedang dijalankan oleh pengguna lain.

3.3.3. Perancangan Home Server

Sebuah *server* yang mampu menyimpan data secara permanen serta memiliki kemampuan mengolah data yang mumpuni sangat dibutuhkan dalam perancangan kulkas pintar ini. *Server* harus dapat memenuhi beberapa fungsi utama, seperti:

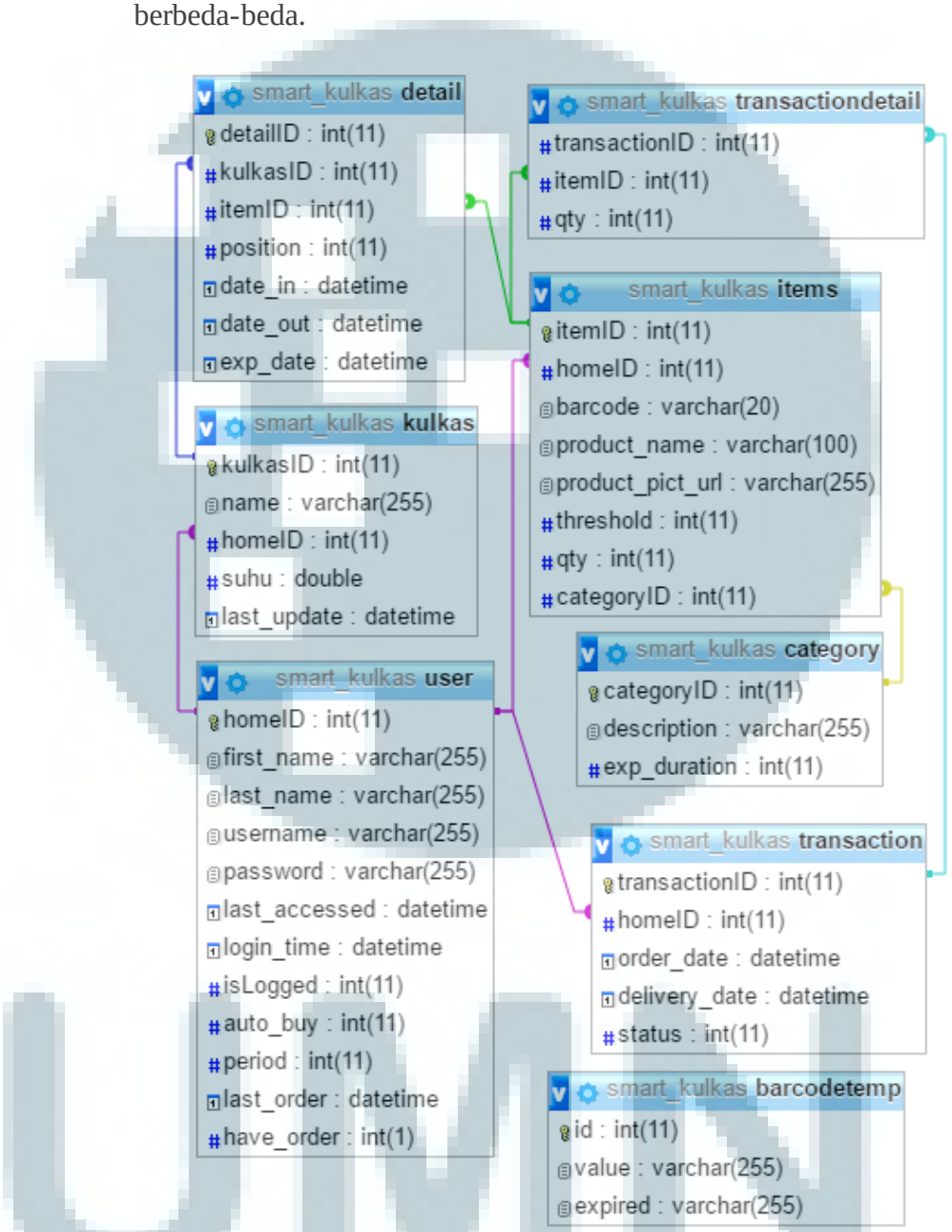
A. Menerima, mengolah, dan menyimpan data dari NodeMCU
B. Menyediakan API bagi perangkat cerdas
C. Mengirim notifikasi ke perangkat cerdas pengguna
D. Menerima input dari *barcode reader*

Selain fungsi di atas, *server* juga harus memiliki kemampuan mengolah data yang mumpuni. Hal ini dikarenakan *server* akan menjadi *server* utama untuk perangkat IoT lainnya yang berada di dalam rumah. Untuk memenuhi segala kebutuhan tersebut, maka *server* yang digunakan adalah sebuah komputer mini Raspberry Pi 3.

Server yang digunakan terdiri dari dua bagian utama, yaitu *database* dan perangkat lunak. Perangkat lunak terbagi menjadi beberapa bagian, yakni perangkat lunak yang berhubungan langsung dengan NodeMCU, perangkat lunak yang berhubungan langsung dengan perangkat cerdas, serta perangkat lunak yang berhubungan dengan *scheduler*.

A. Perancangan Database

Database yang digunakan terdiri dari delapan tabel dengan relasi seperti gambar 3.22. Delapan tabel tersebut memiliki fungsi yang berbeda-beda.



Gambar 3. 22 Database yang akan digunakan

Tabel user berguna untuk menyimpan informasi seperti ID rumah, informasi pengguna seperti nama, *username*, dan *password*, informasi tentang kulkas pintar seperti fitur *auto buy*, periode

pemesanan, serta tanggal pemesanan terakhir. Pada tabel ini juga terdapat informasi waktu pengaksesan terkahir dan waktu *login* terkahir. Informasi ini digunakan sebagai *salt* untuk menghasilkan *token* yang selanjutkan akan digunakan untuk komunikasi antara *smartphone* dengan *server*.

Tabel Kulkas berfungsi untuk menyimpan informasi mengenai kulkas, seperti ID, deskripsi, dan temperatur kulkas serta informasi waktu terakhir temperatur kulkas diperbarui.

Tabel Items berisi informasi mengenai semua barang yang telah dikenali oleh kulkas. Pengguna tidak dapat memasukan barang atau membeli barang yang belum terdaftar pada tabel ini. oleh sebab itu pengguna harus mendaftarkan setiap barang yang belum pernah dimasukan ke dalam *database*.

Tabel Detail berisi informasi mengenai barang apa saja yang pernah dan sedang berada di dalam kulkas. Untuk mendapat informasi tanggal kadaluarsa, maka akan digunakan informasi dari tabel Category.

Tabel Transaction dan Transactiondetail berisi informasi mengenai daftar transaksi yang pernah dilakukan oleh kulkas pintar sehingga pengguna dapat melihat jejak transaksi yang telah terjadi dengan detail.

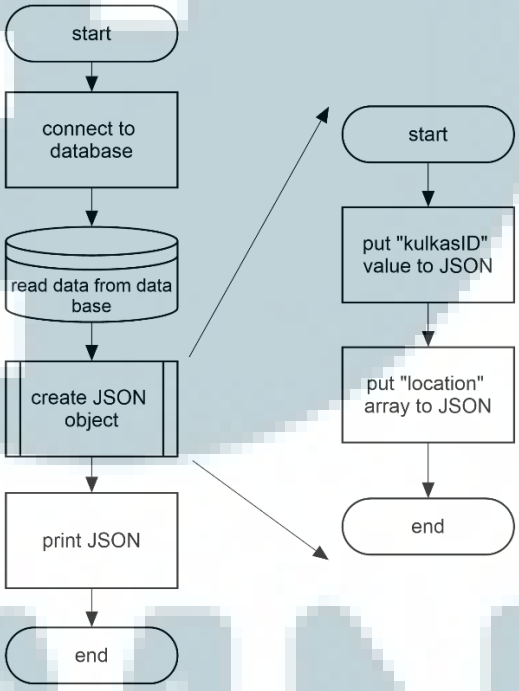
Tabel terakhir dalam *database* adalah Barcodetemp yang berguna untuk menyimpan sementara nilai dari pembacaan *barcode*.

B. Perancangan API untuk NodeMCU

Perangkat lunak yang berhubungan langsung dengan NodeMCU terdiri dari tiga program, yaitu :

B.1. Inisialisasi posisi awal

Program ini berfungsi untuk memberi informasi kepada NodeMCU mengenai posisi barang yang berada di dalam kulkas. Proses dalam program ini tergambar dalam gambar 3.23. Nilai kembalian yang diberikan server kepada NodeMCU berupa sebuah objek JSON seperti pada gambar 3.4.

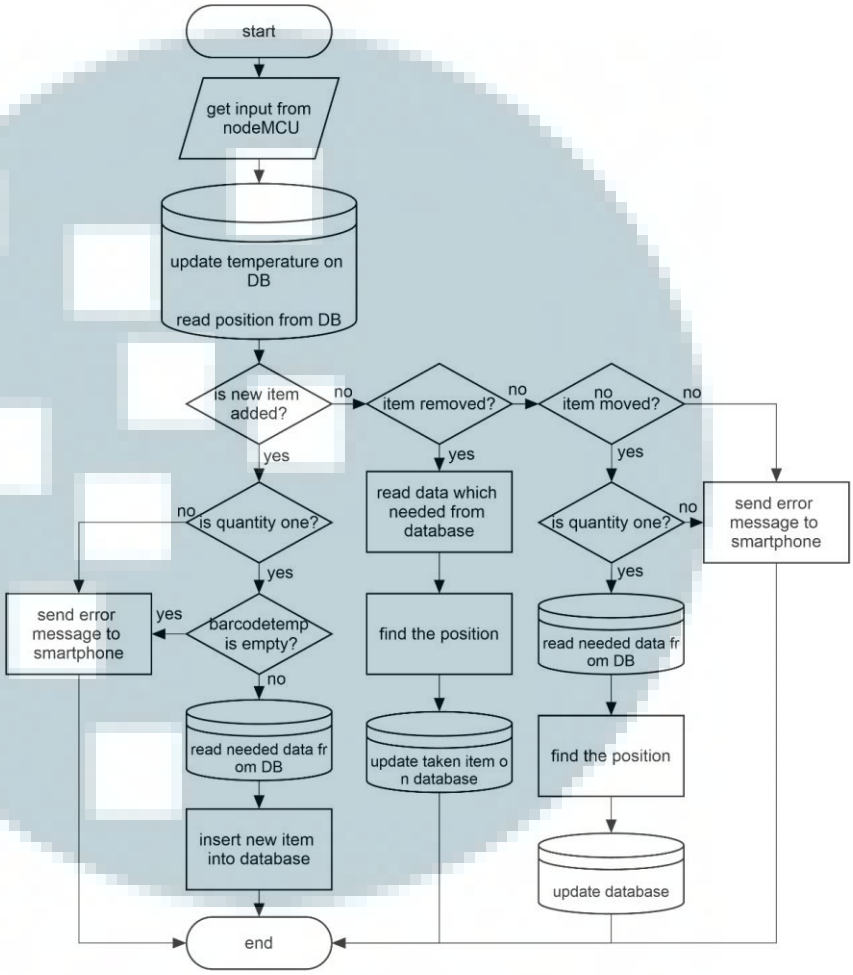


Gambar 3. 23 Diagram alir inisialisasi posisi awal

B.2. Perubahan posisi

Program ini berfungsi untuk menganalisa perubahan yang terjadi serta memperbaharui data di database. Alur kerja dari diagram ini tergambar pada gambar 3.24.

Program ini berjalan dengan menerima input dari NodeMCU. Input yang diterima berupa sebuah objek JSON dengan format seperti pada gambar 3.6.



Gambar 3. 24 Diagram alir perubahan posisi

Setelah menerima input, proses selanjutnya yang dilakukan adalah menyimpan temperatur yang diterima ke dalam *database*. Proses selanjutnya adalah membaca data posisi dari *database*. Setelah semua data yang diperlukan telah lengkap, maka langkah berikutnya adalah menentukan perubahan yang terjadi. Perubahan yang dapat terjadi terdiri dari tiga kemungkinan, yaitu satu barang baru dimasukan, barang dikeluarkan, dan satu barang

dipindahkan. Jika perubahan yang terjadi bukan salah satu dari kemungkinan tersebut, maka *server* akan mengirim notifikasi *error* ke perangkat smartphone pengguna.

Ketika ada barang yang dimasukan ke dalam kulkas, maka *server* akan memastikan terlebih dahulu bahwa tepat hanya ada satu barang yang dimasukan. Apabila barang yang dimasukan lebih dari satu atau data di tabel Barcodetemp tidak ada, maka *server* akan mengirim pesan error ke *smartphone*. Apabila barang yang dimasukan tepat satu buah, maka *server* akan menentukan posisi manakah yang menjadi tempat barang tersebut dan menyimpannya di dalam *database*.

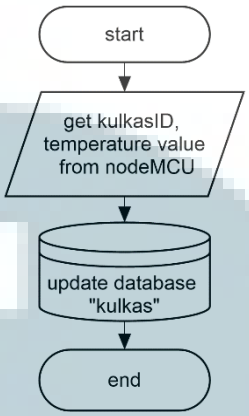
Ketika ada barang yang diambil, maka server akan menentukan posisi mana sajakah yang barangnya diambil lalu memperbarui di dalam database bahwa barang tersebut telah diambil.

Ketika ada barang yang dipindahkan dari satu titik ke titik yang lain, maka program akan menentukan titik asal dan titik akhir dari barang yang dipindah. Setelah itu program akan mengupdate database sesuai data yang baru.

B.3. Perubahan temperatur

Program ini dijalankan ketika hanya terjadi perubahan temperatur tanpa ada perubahan posisi. Program ini menerima input berupa dua buah parameter yang dikirim melalui *HTTP*

request dengan metode POST. Cara kerja dari program ini sesuai dengan diagram alir pada gambar 3.25.

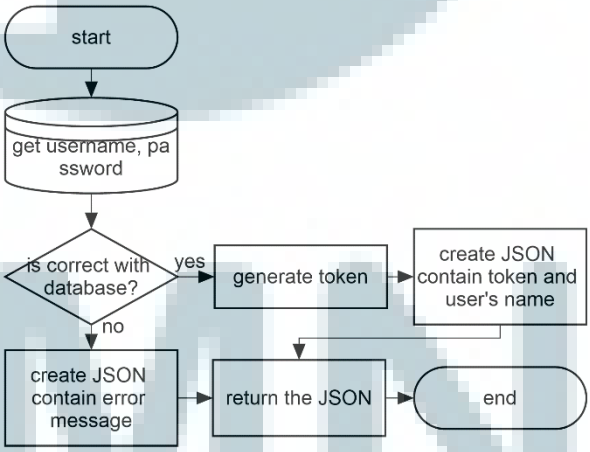


Gambar 3. 25 Diagram alir perubahan suhu

C. Perancangan API untuk Smartphone

Perangkat lunak ini berfungsi sebagai API antara *database home server* dengan perangkat *smartphone*. API ini terdiri dari beberapa bagian, yakni :

C.1. Login



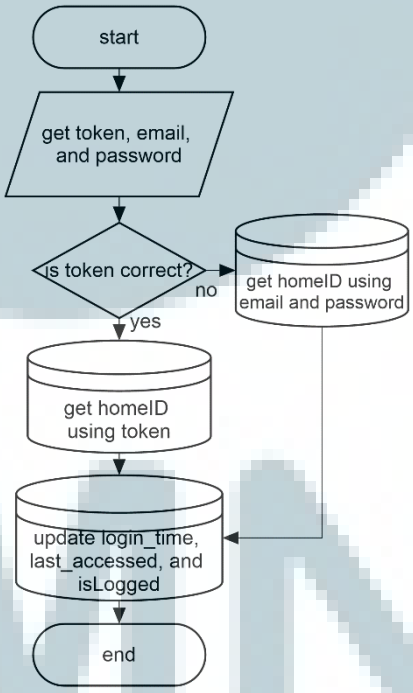
Gambar 3. 26 Diagram alir API login

API ini digunakan untuk verifikasi email dan sandi yang dimasukkan oleh pengguna. Input yang diterima oleh API ini berupa dua buah *field* HTTP POST yang berisi mengenai email

dan sandi pengguna. Apabila login berhasil, maka API ini akan mengembalikan sebuah JSON dengan informasi *token* serta nama pengguna. Cara kerja dari API ini sesuai dengan gambar 3.26.

C.2. Logout

Untuk melakukan *logout*, masukkan yang dapat digunakan terdiri dari dua jenis, yaitu menggunakan token atau menggunakan email dan sandi pengguna. Pada proses logout biasa, input yang digunakan adalah *token*, sedangkan jika terjadi kesalahan perbedaan token, maka logout harus dilakukan dengan menggunakan email dan sandi pengguna. Alur kerja API ini tergambar pada gambar 3.27.



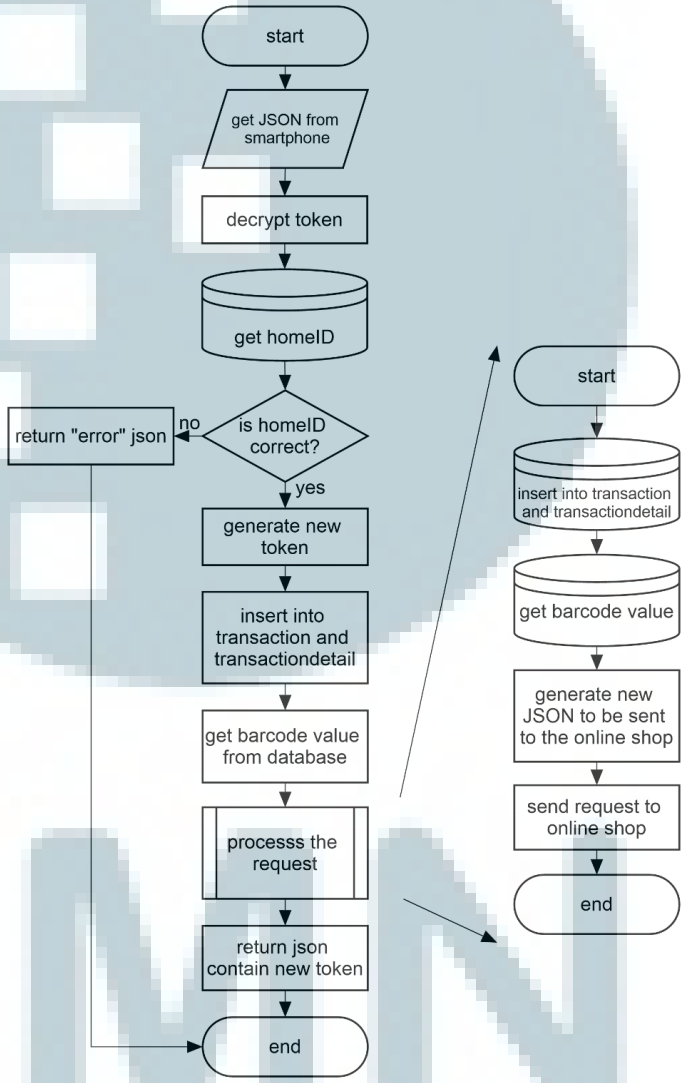
Gambar 3. 27 Diagram alir API logout

C.3. Beli

Fungsi dari API ini adalah menerima *request* dari pengguna mengenai barang apa yang akan dibeli lalu meneruskannya

request tersebut ke penyedia jasa retail secara langsung. API ini menerima masukan berupa JSON dengan format seperti gambar 3.12

Setelah menerima *request* dari pengguna, maka API ini akan memproses *request* tersebut sesuai dengan diagram alir pada gambar 3.28.



Gambar 3. 28 Diagram alir API buy now

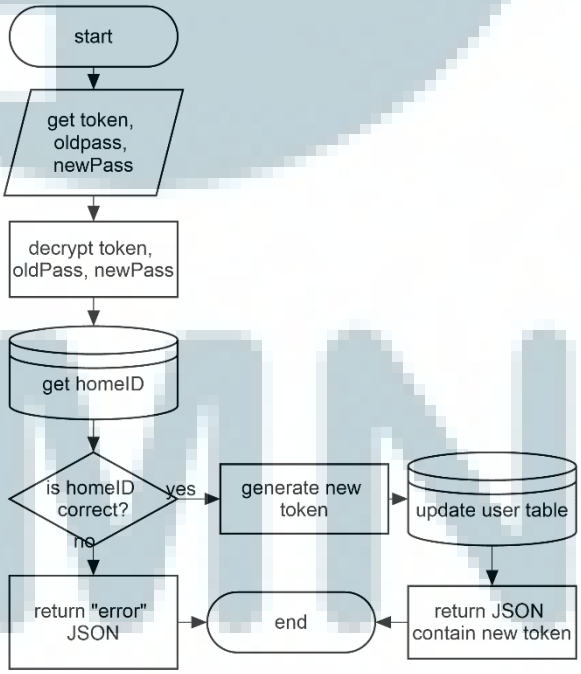
Untuk mengirim request ke penyedia jasa retail, API akan menyusun sebuah JSON dengan format seperti gambra 3.29.

```
{
"email":"email",
"password":"password",
 "items":
  [
   {
    “barcode”:”barcode”
    “qty”:”qty”
   }
  ]
}
```

Gambar 3. 29 Format JSON untuk memesan barang ke toko retail

C.4. Ganti kata sandi

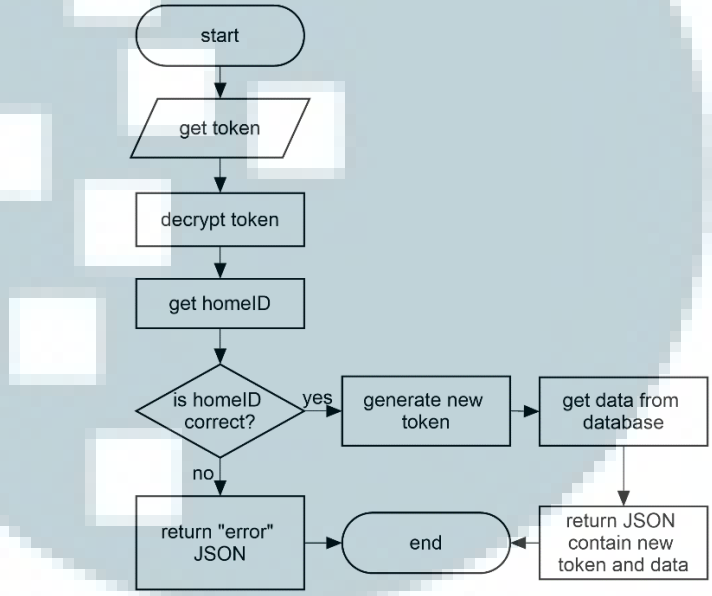
API ini dipanggil ketika pengguna ingin mengganti kata sandi. Masukkan yang diperlukan berupa HTTP POST yang terdiri dari tiga field, yaitu *token*, *newPass*, dan *oldPass*. API ini memberi nilai balik berupa sebuah JSON. Alur kerja API ini tergambar pada gambar 3.30.



Gambar 3. 30 Diagram alir API change password

C.5. Informasi barang dalam kulkas

API ini berfungsi untuk memberi informasi kepada *smartphone* pengguna mengenai daftar barang apa saja yang terdapat di dalam kulkas, termasuk jumlah dan tanggal kadaluarsa dari setiap item. API ini membutuhkan masukkan sebuah *token*. Alur kerja dari API ini terlihat pada gambar 3.31. Sedangkan nilai kembalian dari API ini terlihat pada gambar 3.32.



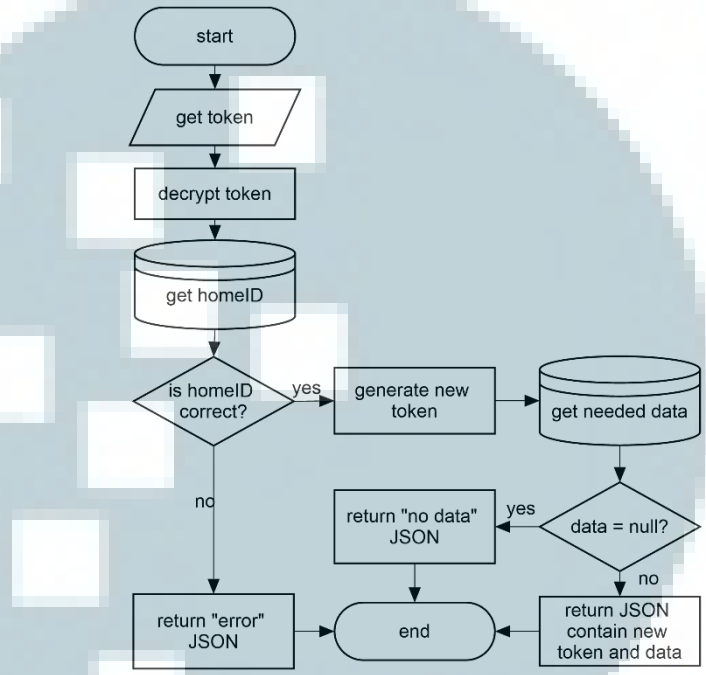
Gambar 3. 31 Diagram alir API get detail

```
{
"token":"token",
"kulkas_ID":[X1, X2],
 "items":
  [
   {
    “kulkasID”:”kulkasID”,
    “itemID”:”itemID”,
    “itemName”:”itemName”,
    “total”:”total”,
    “exp_date”:”exp_date”,
    “imageUrl”:”imageUrl”
   }
  ]
}
```

Gambar 3. 32 Format JSON untuk mengetahui detail barang di dalam kulkas

C.6. Informasi daftar barang yang bisa dibeli

API ini memberi informasi daftar barang yang dapat dibeli melalui aplikasi kulkas pintar. Alur kerja dari API tergambar pada gambar 3.33.



Gambar 3. 33 Diagram alir API get item

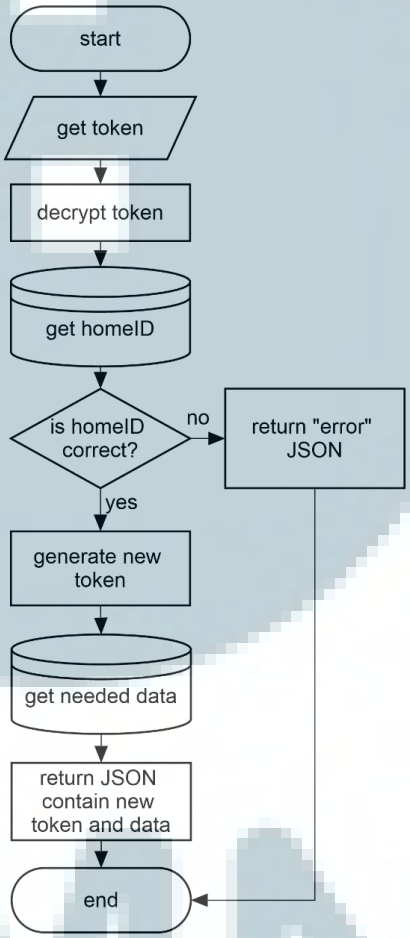
Nilai kembalian yang diberikan oleh API ini berupa sebuah JSON dengan format seperti gambar 3.34.

```
{
"token":"token",
"homeID":"homeID",
 "items":
  [
   {
    "itemID":"itemID",
    "itemName":"itemName",
    "barcode":"barcode",
    "imageUrl":"imageUrl"
   }
  ]
}
```

Gambar 3. 34 Format JSON kembalian dari API get item

C.7. Informasi detail barang

API ini memberikan informasi daftar barang yang terdapat dalam *database*, termasuk jumlah barang minimum dan maksimum. API ini dijalankan saat menu *manage refrigerator* dimuat. Masukkan yang diperlukan adalah sebuah *token*. Alur kerja dari API ini sesuai dengan diagram alir pada gambar 3.35.



Gambar 3. 35 Diagram alir API get detail

Alur kerja dari API ini dimulai dengan menerima *token* dari perangkat cerdas android. Setelah itu, *token* yang diterima akan di dekripsi terlebih dahulu dan digunakan untuk mendapatkan *homeID* klien. *homeID* tersebut digunakan untuk memperoleh

data yang dibutuhkan dari *database*. Data yang dibaca meliputi ID barang, nama barang, jumlah minimum, jumlah maksimum, dan alamat gambar barang. Langkah terakhir dari API ini adalah merancang JSON dan menampilkannya sebagai nilai kembalian.

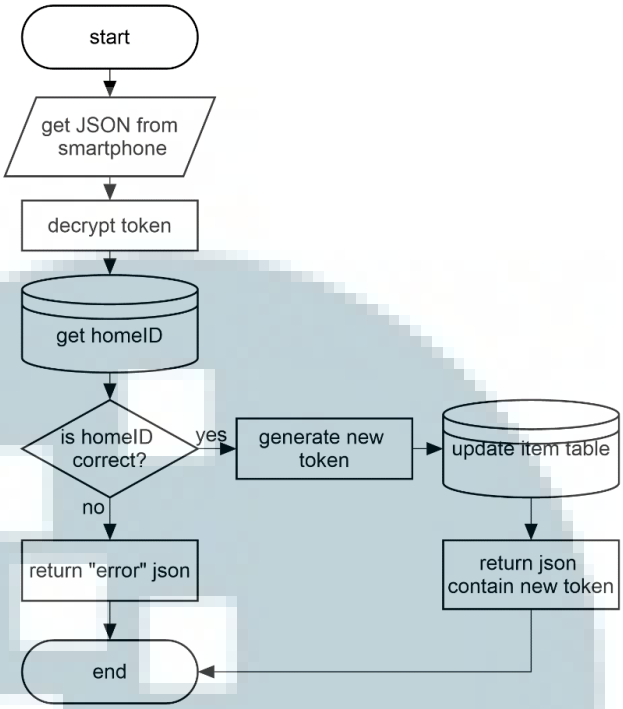
Nilai kembalian dari API ini adalah sebuah JSON dengan format seperti gambar 3.36.

```
{
"token":"token",
"items":
  [
   {
    “homeID”:”homeID”,
    “itemID”:”itemID”,
    “itemName”:”itemName”,
    “min”:”min”,
    “max”:”max”,
    “imageURL”:”imageURL”
   }
  ]
}
```

Gambar 3. 36 Format JSON kembalian dari API get detail

C.8. Perbarui batas ambang barang

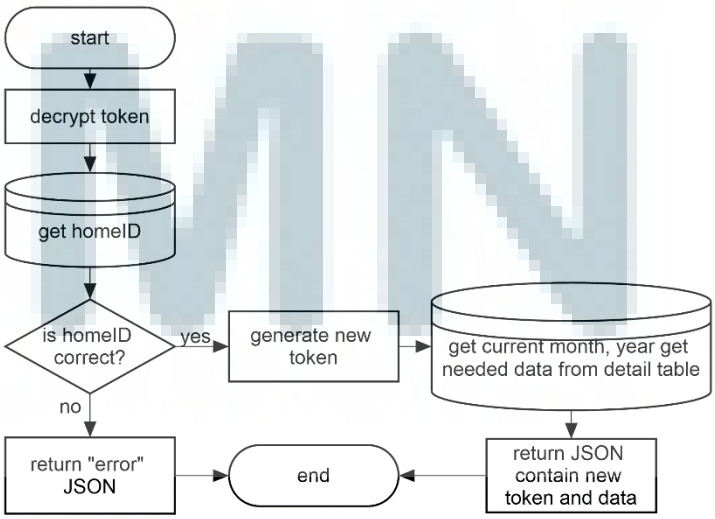
API ini merupakan lanjutan dari API informasi detail barang. API ini dipanggil setelah pengguna selesai melakukan pengaturan pada menu *manage refrigerator*. API yang membutuhkan input JSON seperti gambar 3.11 berfungsi untuk memperbarui nilai batas ambang dari setiap barang di *database home server*. Nilai kembalian yang diberikan oleh API ini adalah sebuah JSON yang berisi nilai *token* yang dapat digunakan pada operasi selanjutnya. Alur kerja dari API ini tergambar pada gambar 3.37.



Gambar 3. 37 Diagram alir API manage refrigerator

C.9. Riwayat benda yang telah di ambil

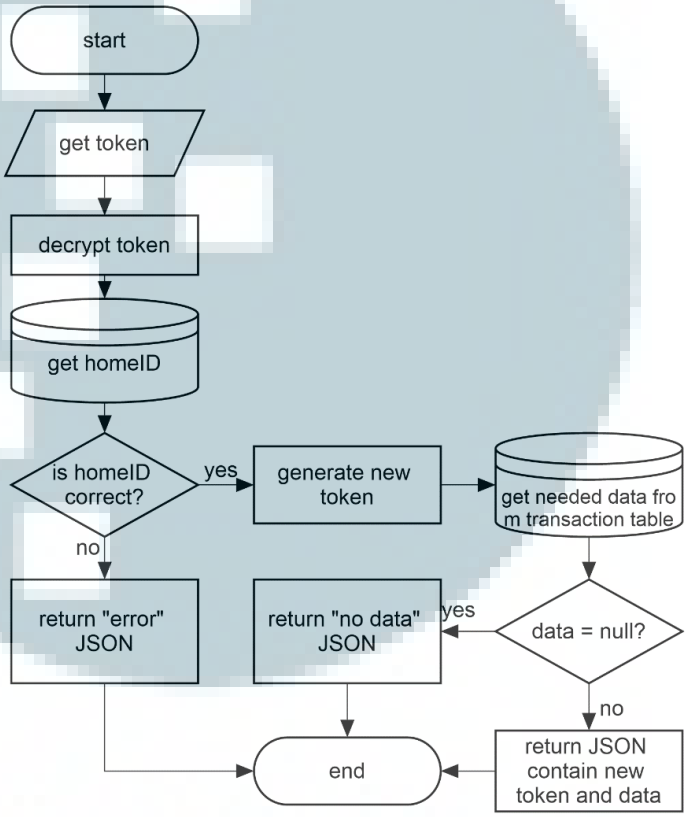
API ini digunakan untuk melihat daftar riwayat benda yang telah diambil dari kulkas dalam kurun waktu tiga bulan terakhir. API ini memerlukan masukkan berupa sebuah *token* dan mengembalikan nilai sebuah JSON dengan format sesuai gambar 3.17. Alur kerja dari API ini tergambar pada gambar 3.38.



Gambar 3. 38 Diagram alir API get item history

C.10. Riwayat transaksi

API ini memberi informasi mengenai riwayat transaksi yang terjadi selama tiga bulan terakhir. API ini memerlukan masukkan sebuah *token* dan memberikan nilai balik sebuah JSON dengan format seperti gambar 3.15. Alur kerja dari API ini digambarkan melalui diagram alir pada gambar 3.39.

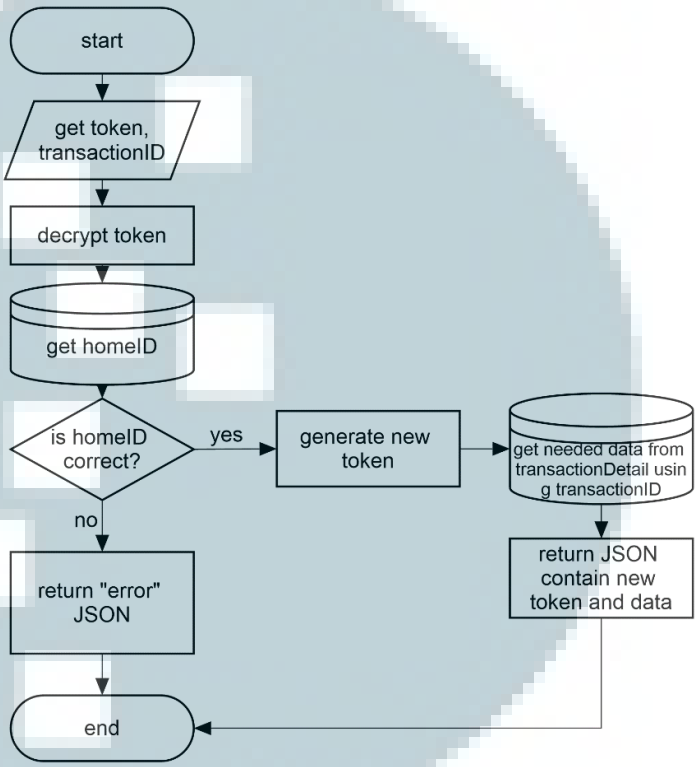


Gambar 3. 39 Diagram alir API get transaction history

C.11. Detail riwayat transaksi

API detail riwayat transaksi dipanggil ketika pengguna ingin melihat detail dari sebuah riwayat transaksi yang mereka pilih. Masukkan yang dibutuhkan API ini adalah *token* dan ID transaksi yang dikirim melalui sebuah HTTP POST *Request* dari perangkat android. Alur kerja API ini tergambar pada gambar

3.40. API ini memberikan sebuah nilai balik berupa sebuah JSON dengan format seperti gambar 3.16. Informasi yang diberikan melalui JSON ini meliputi *token* baru, nama barang, dan jumlah dari setiap barang yang dibeli.



Gambar 3. 40 Diagram alir API get transaction detail

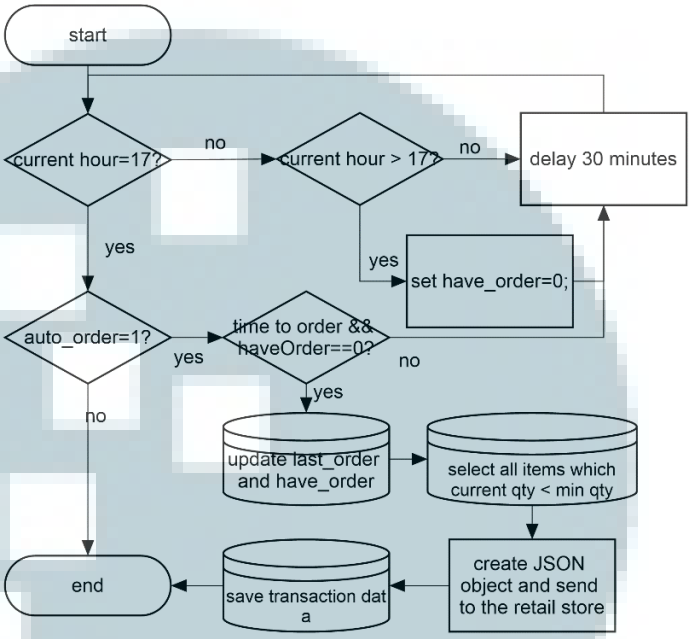
D. Perancangan Perangkat Lunak untuk Scheduler

Perancangan perangkat lunak untuk *scheduler* terdiri dari tiga bagian, yaitu *scheduler* untuk melakukan pemesanan barang ke toko retail, *scheduler* untuk memeriksa temperatur kulkas, dan *scheduler* untuk memeriksa tanggal kadaluarsa.

D.1. Scheduler untuk pemesanan barang

Scheduler ini berfungsi untuk memesan barang yang habis ke toko retail. *Scheduler* ini hanya akan memesan jika pengguna mengaktifkan fitur *auto buy*. Pemesanan dilakukan setiap pukul

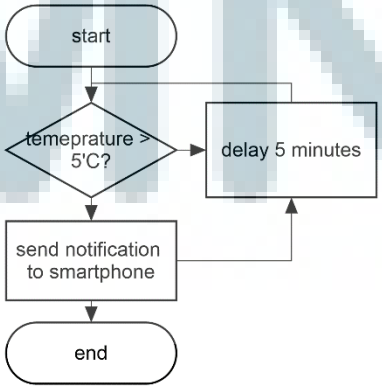
17.00 WIB dengan interval waktu sesuai dengan yang ditentukan oleh pengguna. Cara kerja *scheduler* secara keseluruhan tergambar dari gambar 3.41.



Gambar 3. 41 Diagram alir scheduler untuk auto order

D.2. Scheduler untuk pengawasan temperatur

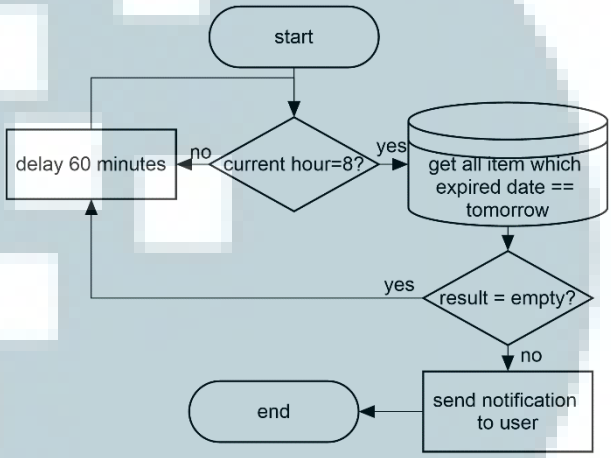
Scheduler ini berfungsi memeriksa temperatur di dalam kulkas secara rutin. Apabila suhu mengalami kenaikan hingga lebih dari 10°C, maka *scheduler* ini akan mengirim notifikasi ke perangkat cerdas pengguna. Cara kerja *scheduler* ini tergambar pada gambar 3.42.



Gambar 3. 42 Diagram alir pemeriksaan suhu kulkas.

D.3. Scheduler untuk pengawasan tanggal kadaluarsa

Sheduler ini berfungsi memeriksa tanggal kadaluarsa dari setiap barang yang tersimpan didalam kulkas. Scheduler ini akan mengirimkan notifikasi kepada pengguna jika terdapat barang dengan tanggal kadaluarsa esok hari. Alur kerja scheduller ini tergambar pada gambar 3.43.



Gambar 3. 43 Diagram alir scheduler pemeriksaan tanggal kadaluarsa