TUGAS AKHIR – EE 184801

RANCANG BANGUN PROTOTIPE SENSOR PINTAR *WEARABLE* BERBASIS *INTERNET OF THINGS* UNTUK MONITORING POPOK

Budiman Agung Wibowo
NRP 07111440000169

Dosen Pembimbing
Dr. Ir. Hendra Kusuma, M.Eng.Sc.
Ir. Tasripan, M.T.

DEPATERMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2018

TUGAS AKHIR – EE 184801

RANCANG BANGUN PROTOTIPE SENSOR PINTAR *WEARABLE* BERBASIS *INTERNET OF THINGS* UNTUK MONITORING POPOK

Budiman Agung Wibowo
NRP 07111440000169

Dosen Pembimbing
Dr. Ir. Hendra Kusuma, M.Eng.Sc.
Ir. Tasripan, MT.

DEPARTEMEN TEKNIK ELEKTRO
Fakultas Teknologi Elektro
Institut Teknologi Sepuluh Nopember
Surabaya 2019

FINAL PROJECT – EE 184801

DEVELOPMENT OF IOT BASED WEARABLE SMART SENSOR FOR DIAPER MONITORING

Budiman Agung Wibowo
NRP 07111440000169

Supervisor
Dr. Ir. Hendra Kusuma, M.Eng.Sc.
Ir. Tasripan, MT.

ELECTRICAL ENGINEERING DEPARTMENT
Faculty of Electrical Technology
Sepuluh Nopember Institute of Technology
Surabaya 2019

PERNYATAAN KEASLIAN
TUGAS AKHIR

Dengan ini saya menyatakan bahwa isi sebagian maupun keseluruhan Tugas Akhir saya dengan judul "Rancang Bangun Prototipe Sensor Pintar *Wearable* Berbasis *Internet of Things* untuk Monitoring Popok" adalah benar-benar hasil karya intelektual mandiri, diselesaikan tanpa menggunakan bahan-bahan yang tidak diizinkan dan bukan merupakan karya orang lain yang saya akui sebagai karya sendiri.

Semua referensi yang dikutip maupun dirujuk telah ditulis secara lengkap pada daftar pustaka.

Apabila ternyata pernyataan ini tidak benar, saya bersedia menerima sanksi sesuai peraturan yang berlaku.

Surabaya, 23 Januari 2019

Budiman Agung Wibowo
NRP. 0711 14 40000 169

RANCANG BANGUN PROTOTIPE SENSOR PINTAR *WEARABLE* BERBASIS *INTERNET OF THINGS* UNTUK MONITORING POPOK

TUGAS AKHIR

Diajukan Guna Memenuhi Sebagian Persyaratan Untuk
Memperoleh Gelar Sarjana Teknik
Pada
Bidang Studi Elektronika
Departemen Teknik Elektro
Institut Teknologi Sepuluh Nopember

Menyetujui

Dosen Pembimbing I	Dosen Pembimbing II
Dr. Ir. Hendra Kusuma, M.Eng.Sc.	Ir. Tasripan, M.T.
NIP: 196409021989031003	NIP: 196204181990031004

KEMENTERIAN RISET, TEKNOLOGI, DAN PENDIDIKAN TINGGI
INSTITUT TEKNOLOGI SEPULUH NOPEMBER
FAS TEKNOLOGI ELEKTRO
DEPARTEMEN
TEKNIK ELEKTRO

SURABAYA
JANUARI 2019

RANCANG BANGUN PROTOTIPE SENSOR PINTAR *WEARABLE* BERBASIS *INTERNET OF THINGS* UNTUK MONITORING POPOK

Nama : Budiman Agung Wibowo
Pembimbing 1 : Dr. Ir. Hendra Kusuma, M.Eng.Sc.
Pembimbing 2 : Ir. Tasripan, M.T.

ABSTRAK

Popok adalah suatu jenis pakaian yang berfungsi untuk menampung urine dan feses orang yang tidak bisa mengendalikan pergerakan kandung kemih ataupun pergerakan ususnya. Apabila popok sudah berisi harus segera diganti agar tidak menimbulkan iritasi pada kulit. Akan tetapi, seringkali pengasuh lupa ataupun terlalu sibuk sehingga terlambat dalam mengganti popok. Maka dari itu, dikembangkan suatu sensor pintar yang dapat dikenakan (*wearable*) untuk menyelesaikan permasalahan tersebut. Sensor pintar ini berfungsi untuk mengawasi kondisi kelembapan dan temperatur popok dan dapat memberikan informasi tersebut secara *realtime* tanpa harus memeriksa popok secara rutin. Pada tugas akhir ini akan dirancang suatu sensor pintar berbasis *Internet of Things* yang dapat dikenakan (*wearable*). Sistem ini terdiri dari sebuah sensor pintar menggunakan sensor HTU21 untuk mendeteksi kondisi kelembapan dan temperatur popok dan mengirimkan data tersebut ke aplikasi android melalui *bluetooth* dan sebuah aplikasi android untuk menampilkan hasil pembacaan dan juga memberikan notifikasi apabila popok sudah berisi. Sistem menggunakan mikrokontroler Bluno Beetle untuk integrasi sensor dan pengiriman data menuju aplikasi android. Apabila tingkat kelembapan popok sudah melewati 60% dan temperature sudah melewati 33ºC maka aplikasi akan memberikan notifikasi dan membunyikan alarm sebagai pengingat untuk mengganti popok. Dari 25 kali pengujian menggunakan urine, perangkat berhasil mendeteksi keberadaan urine sebanyak 18 kali. Sehingga persentasi keberhasilannya adalah sebesar 72%.

Kata kunci: *Bluetooth Low Energy, Diaper, Internet of Things, Smart Device, Wearable Sensors.*

.....Halaman ini sengaja dikosongkan.....

DEVELOPMENT OF IOT BASED WEARABLE SMART SENSOR FOR DIAPER MONITORING

Name : Budiman Agung Wibowo
1st Advisor : Dr. Ir. Hendra Kusuma, M.Eng.Sc.
2nd Advisor : Ir. Tasripan, M.T.

ABSTRACT

Diapers are a type of clothing that serves to contain the urine and feces of people who cannot control their bladder or their bowel movement. If the diaper has been filled it must be replaced immediately so as not to cause irritation to the skin. However, caregivers often forget or are too busy until it is too late to change the diapers. Therefore, a wearable smart sensor are developed to solve the problem. This smart sensor serves to monitor the humidity and temperature of diapers and can provide that information in realtime without having to manually check diapers regularly. In this final project will be developed an Internet of Things based wearable smart sensor. This system consists of a smart sensor using the HTU21 sensor to detect humidity and temperature of the diaper and then send the data to an android application via bluetooth and an android application to display the reading results and also provide notifications when the diaper is filled. The system uses the Bluno Beetle microcontroller for sensor integration and data transmission towards the android application. If the humidity level of the diaper has exceeded 60% and the temperature has passed 33^0C, the application will give a notification and sound the alarm as a reminder to change the diaper. From 25 times testing using urine, the device successfully detected the presence of urine 18 times. Hence, the percentage of success is 72%.

***Keywords**: Bluetooth Low Energy, Diaper, Internet of Things, Smart Device, Wearable Sensors.*

.....Halaman ini sengaja dikosongkan.....

KATA PENGANTAR

Puji syukur atas rahmat yang telah diberikan oleh Tuhan Yang Maha Esa. Karena berkat rahmat, berkat, dan bimbingan-Nya penulis dapat menyelesaikan tugas akhir dengan judul **"RANCANG BANGUN PROTOTIPE SENSOR PINTAR *WEARABLE* BERBASIS *INTERNET OF THINGS* UNTUK MONITORING POPOK"**. Penulis sadar bahwa dengan tanpa adanya bantuan dari pihak lain, penulis tidak dapat menyelesaikan laporan tugas akhir ini. Dengan segala hormat dan rasa syukur yang sedalam-dalamnya penulis mengucapkan terima kasih kepada:

1. Kedua orang tua beserta keluarga yang senantiasa memberikan doa, nasihat, motivasi, dukungan, dan pengingat sehingga memungkinkan penulis untuk menyelesaikan tugas akhir ini.
2. Dr. Ir. Hendra Kusuma, M.Eng.Sc. dan Ir. Tasripan, MT., selaku dosen pembimbing, atas bimbingan, inspirasi, pengarahan, dan motivasi yang diberikan selama pengerjaan penelitian ini.
3. Seluruh dosen dan karyawan Departemen Teknik Elektro ITS yang telah memberikan banyak ilmu.
4. Mohammad Reza Maulana, Nicolas Rezadhi Pradipta, Fidelis Prasetyo, dan Rephando Amstrabena selaku teman-teman terdekat penulis yang senantiasa membantu memberikan bantuan moral dalam menyelesaikan tugas akhir ini.
5. Ahmad Naufal Firdaus yang sangat berjasa dalam membantu penyelesaian tugas akhir ini.
6. Mitha Mulia Virdianty, yang senantiasa menemani dan mendampingi penulis dalam pengerjaan tugas akhir ini.

Terima kasih juga kepada semua pihak yang mungkin tidak dapat penulis sebutkan satu persatu. Penulis juga sadar bahwa banyaknya kekurangan dari penulis dalam penulisan laporan ini. Penulis dengan terbuka menerima masukan, kritik, dan saran agar laporan ini dapat bermanfaat kedepannya.

Surabaya, 20 Januari 2019

Budiman Agung Wibowo
NRP. 07111440000169

.....Halaman ini sengaja dikosongkan.....

DAFTAR ISI

DAFTAR GAMBAR

.....Halaman ini sengaja dikosongkan.....

DAFTAR TABEL

.....Halaman ini sengaja dikosongkan.....

BAB I
PENDAHULUAN

Tugas akhir merupakan suatu penelitian yang dilakukan sebagai persyaratan akademik untuk mendapatkan gelar sarjana teknik di Institut Teknologi Sepuluh Nopember (ITS) Surabaya. Tugas akhir ini mengambil topik perancangan dan pembangunan sebuah sensor pintar berbasis *internet of things* yang dapat dikenakan untuk memonitor kondisi popok.

Pada bab ini membahas mengenai hal-hal yang mendahului pelaksanaan tugas akhir. Hal tersebut meliputi latar belakang, perumusan masalah, tujuan penulisan, batasan masalah, metodologi penelitian, sistematika penulisan, dan relevansi.

1.1. Latar Belakang

Popok adalah semacam pakaian yang berfungsi untuk menampung urin dan feses agar tidak mengotori lingkungan sekitar. Popok digunakan karena tidak semua manusia memiliki kemampuan untuk mengendalikan pergerakan kandung kemih atau pergerakan ususnya, baik hal itu dikarenakan oleh otak yang belum cukup berkembang untuk mengatur pergerakan kandung kemihnya seperti pada bayi ataupun dikarenakan oleh penyakit yang menyebabkan ketidakmampuan dalam mengendalikan pergerakan kandung kemih seperti kelumpuhan yang bisa dialami pada kasus-kasus orang dewasa. Popok yang berada di pasaran pada umumnya terbagi menjadi dua kategori berdasarkan penggunanya, yaitu popok untuk bayi dan popok untuk orang dewasa.

Popok yang basah ataupun kotor setelah diisi oleh urin dan feses apabila dibiarkan terlalu lama tanpa diganti dapat menyebabkan munculnya penyakit pada kulit seperti iritasi yang disebut juga sebagai ruam popok [1]. Ruam popok ini adalah masalah yang umum ditemui apabila sedang merawat bayi ataupun orang tua yang mengenakan popok. Biasanya ruam popok hanya menyebabkan iritasi ringan pada kulit, akan tetapi dalam beberapa kasus unik dapat pada infeksi bateri dan juga komplikasi yang dapat berujung kepada kematian. Ruam popok ini dapat diatasi dengan rutin mengganti popok, memberikan bedak anti-jamur, atau krim *hydrocortisone* [2].

Beberapa merek popok yang ada di pasaran sekarang ini sudah menggunakan bahan yang dapat mendeteksi urin dalam popok, dimana

bahan ini akan berubah warna ketika terjadi kontak antara bahan tersebut dengan urin dalam jumlah tertentu. Perawat hanya perlu melihat popok untuk memastikan apakah kondisi popok sudah kotor atau tidak. Akan tetapi popok seperti ini juga masih kurang baik dikarenakan pada saat bahan pada popok mendeteksi keberadaan urin, perawat tidak akan langsung mengetahuinya sehingga kemungkinan terjadinya ruam popok akan tetap ada.

Untuk menjawab permasalahan tersebut, pada penelitian ini akan dibangun suatu sensor pintar yang dapat dikenakan pada popok yang mampu mengawasi dan memberitahu kondisi popok secara *real-time* kepada perawat dan juga memberikan peringatan apabila kondisi kelembapan dan temperature popok telah melewati batas tertentu.

1.2. Perumusan Masalah

Permasalahan yang dibahas dalam tugas akhir ini adalah:

1. Bagaimana sensor pintar mampu mendeteksi kondisi kelembapan dan temperatur popok dan mengirimkan data ke basis data.
2. Bagaimana sistem mampu memberikan peringatan apabila kondisi kelembapan dan temperatur popok sudah melewati batas tertentu.
3. Perancangan sebuah *Mobile App* untuk menunjukkan hasil pembacaan sistem.

1.3. Tujuan Penelitian

Penelitian pada penelitian ini bertujuan sebagai berikut:

1. Merancang sensor pintar dengan sensor temperatur dan kelembapan untuk mengetahui kondisi popok
2. Merancang *Mobile App* yang mampu menunjukkan hasil pembacaan sensor dan mampu memberikan notifikasi kepada perawat.
3. Mengurangi kecenderungan terjadinya kasus ruam popok pada pengguna popok.

1.4. Batasan Masalah

Batasan masalah dari tugas akhir ini adalah sebagai berikut:

1. Pengujian pengambilan data oleh sensor pintar pada tugas akhir ini hanya dilakukan pada bayi.
2. Uji coba pembacaan sensor terhadap popok menggunakan air dan urin tetapi tidak menggunakan feses.

1.5. Metodologi Penelitian

Langkah-langkah yang dikerjakan pada tugas akhir ini adalah sebagai berikut:

1. Studi literatur

Studi literatur merupakan tahap pengumpulan informasi dan referensi dari berbagai sumber sebagai acuan dalam menjalankan proses Tugas Akhir. Referensi dan informasi yang dimaksud diperoleh dari buku, jurnal, tesis, paper, artikel, internet, dan hasil konsultasi dengan dosen pembimbing.

2. Perancangan Perangkat Keras

Pada tahap ini dilakukan perancangan perangkat keras dari sensor pintar berbasis *Internet of Things* yang mampu dikenakan. Tahap yang dilakukan meliputi:

1. Desain elektronik penempatan sensor, mikrokontroler, dan baterai.
2. Desain rangka untuk menyatukan sistem.

3. Perancangan Perangkat Lunak

Pada tahap ini dilakukan perancangan dan pembuatan perangkat lunak sensor dan juga *Mobile App*. Tahap ini meliputi:

1. Merancang program Bluno untuk mendapatkan data temperature dan kelembapan dari sensor dan mengirimkannya ke *Mobile App*.
2. Merancang sebuah *Mobile App* untuk menampilkan hasil pembacaan sensor yang dapat mengirimkan data ke *databank* dan mengambil data dari *databank* dan juga dapat memberikan peringatan kepada perawat.

4. Persiapan Alat dan Bahan

Pada tahap ini dilakukan persiapan alat-alat dan bahan yang dibutuhkan untuk merealisasi sensor pintar berbasis *Internet of Things* yang telah dirancang pada tahap sebelumnya. Hal ini dilakukan agar proses pembuatan dan realisasi komponen-komponen lebih jelas dan lebih terstruktur, sehingga mengurangi kemungkinan terjadinya kesalahan rancang yang dapat berujung kepada kerugian secara materiil. Sistem ini membutuhkan beberapa materi di dalamnya, beberapa komponen utama yang dibutuhkan antara lain: Bluno Beetle, Sensor Temperatur dan Kelembapan HTU21, Battery Socket, dan Baterai Kancing 3,6V. Pembelian bahan-bahan tersebut dilakukan pada: Berkat Elektronik, Isee, dan beberapa toko *Online*.

5. Realisasi Sistem

Pada bagian ini dilakukan proses pembuatan alat. Proses realisasi ini termasuk perancangan rangkaian elektronika, perangkat keras, dan juga perangkat lunak yang dibutuhkan agar sistem dapat bekerja dengan semestinya.

Tahap realisasi ini dilakukan pada laboratorium elektronika dasar B-202 Departemen Teknik Elektro ITS dan tempat kediaman pribadi penulis

6. Tahap Pengujian Sistem

Pada tahap ini dilakukan pengujian akhir sistem secara keseluruhan. Pengujian sistem meliputi pengujian pembacaan temperatur dan kelembapan pada popok, pengiriman data ke *Mobile App*, pengiriman data dari *Mobile App* ke Databank, dan pengambilan data dari Databank ke *Mobile App*.

7. Analisa dan Evaluasi

Tahap ini dimaksudkan untuk menganalisis hasil data pengujian dan mengevaluasi kelebihan dan kekurangan sistem. Pada tahap ini juga diharapkan dapat mengidentifikasi masalah-masalah yang ditemui selama proses jalannya penelitian serta mengusulkan solusi terhadap masalah tersebut.

8. Penyusunan Laporan Tugas Akhir

Setelah dilakukan pengujian dan pengambilan data, dilanjutkan dengan penulisan laporan penelitian berdasarkan hasil pengujian

sistem untuk merangkum seluruh rangkaian kegiatan penelitian. Laporan tugas akhir merangkum seluruh hal yang berkaitan dengan tugas akhir yaitu meliputi pendahuluan, tinjauan pustaka, perancangan dan pembuatan sistem, pengujian dan Analisa data, serta penutup.

1.6. Sistematika Penulisan

Laporan tugas akhir ini terdiri dari lima bab dengan sistematika penulisan sebagai berikut:

- **BAB I: Pendahuluan**
  Bab ini meliputi latar belakang penelitian, perumusan masalah, tujuan penelitian, batasan masalah, metodologi, sistematika penulisan penelitian dan relevansi.
- **BAB II: Tinjauan Pustaka**
  Bab ini menjelaskan tentang dasar teori yang dibutuhkan dalam pengerjaan tugas akhir ini. Dasar teori yang menunjang meliputi teknologi *wearable*, *Internet of Things,* Bluno Beetle, Sensor temperatur dan kelembapan HTU21, Arduino IDE, Android Studio, Firebase, AutoDesk EAGLE (Student Version).
- **BAB III: Perancangan Sistem**
  Bab ini menjelaskan tentang perencanaan sistem yang meliputi perangkat keras (*hardware*) dan perangkat lunak (*software*) untuk pembuatan alat ini.
- **BAB IV: Pengujian dan Analisis**
  Bab ini menjelaskan tentang hasil yang didapat dari pengujian tiap blok sistem baik masing-masing maupun secara keseluruhan.
- **BAB V: Penutup**
  Bab ini menjelaskan tentang kesimpulan yang diperoleh terhadap hasil realisasi alat dan juga saran untuk pengembangan alat di kemudian hari

1.7. Relevansi

Teknologi sensor pintar wearable berbasis *Internet of Things* merupakan hal yang cukup relevan dengan kehidupan masyarakat sekarang ini, dimana masyarakat semakin sibuk dengan pekerjaannya sehingga menimbulkan kebutuhan akan hal yang praktis. Dan dengan adanya *Smartphone* dan Internet di genggaman, maka penerapan *Internet of Things* dalam kehidupan sehari-hari menjadi semakin relevan.

.....Halaman ini sengaja dikosongkan.....

BAB II
TINJAUAN PUSTAKA

Suatu penelitian memerlukan teori-teori yang sudah ada sebelumnya untuk dikaji lebih dalam memperkuat argumen penulis. Teori tersebut digunakan untuk membantu penulis dan sebagai dasar dalam membuat suatu penelitian.

Pada bab ini terdapat teori dasar yang menjadi landasan untuk merumuskan dan menyelesaikan masalah yang akan dibahas pada penelitian ini. Pada bagian ini terdapat tinjauan pustaka tentang komponen-komponen yang akan digunakan untuk membuat alat pada penelitian ini.

2.1. Teknologi *Wearable*

Teknologi *Wearable* adalah perangkat-perangkat elektronik dan computer yang diimplementasikan kepada sebuah pakaian ataupun aksesoris yang dapat dikenakan pada bagian tubuh [3]. Perangkat *wearable* cenderung lebih rumit daripada perangkat *handheld* yang biasa ada pada pasaran karena alat *wearable* biasanya memiliki fungsi sensor dan scan yang tidak biasa ada pada perangkat *handheld* seperti pada *mobile phone* ataupun laptop.

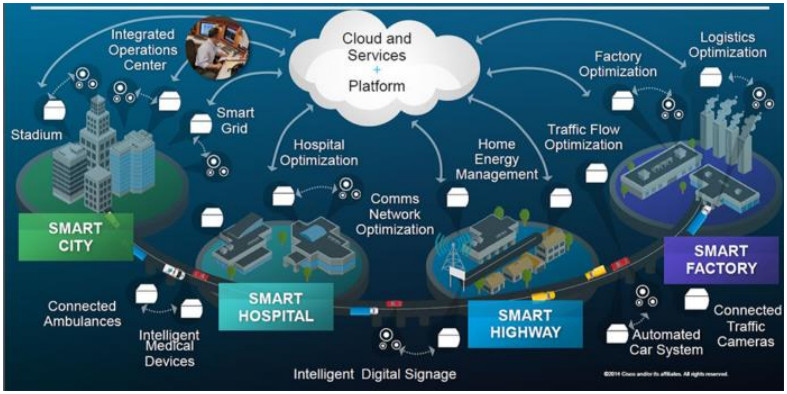
Pada umumnya, teknologi *wearable* memiliki kemampuan untuk berkomunikasi data sehingga memungkinkan penggunanya untuk mengakses informasi secara *realtime.*

Gambar 2.1 Contoh alat *wearable* ***[4]***

2.2. *Internet of Things*

Internet of Things atau yang biasa disingkat dengan IoT merupakan sebuah konsep sistem jaringan fisik yang terdiri dari komponen elektronik, perangkat lunak, sensor, actuator dan terhubung dalam suatu jaringan internet, sehingga objek tersebut dapat saling bertukar data [5].

IoT melibatkan konektivitas dari perangkat seperti *Smartphone, Tablet,* dan laptop, kepada perangkat-perangkat ataupun objek sehari-hari yang biasanya tidak dapat terhubung ke internet. Sehingga dengan ditambahnya teknologi IoT ini, perangkat-perangkat ini dapat berkomunikasi dan berinteraksi melalui internet.



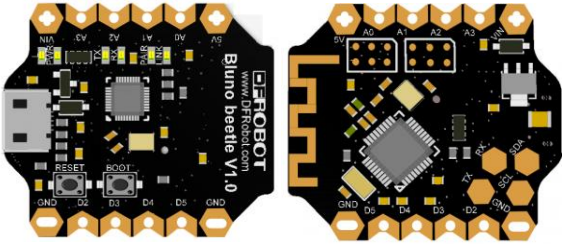
Gambar 2.2 Aplikasi dari IoT ***[6]***

Skema dari hubungan antar komponen dapat dilihat pada gambar 2.2 dimana aplikasi penggunaan IoT mencakup hampir seluruh aspek kehidupan manusia. Dengan adanya konsep IoT ini, aspek-aspek tersebut dapat dikontrol oleh suatu alat pengendali yang terhubung dengan internet, yang kemudian juga dapat terhubung satu sama lain, sehingga menciptakan adanya integrase untuk meningkatkan efisiensi, akurasi, dan faktor ekonomi.

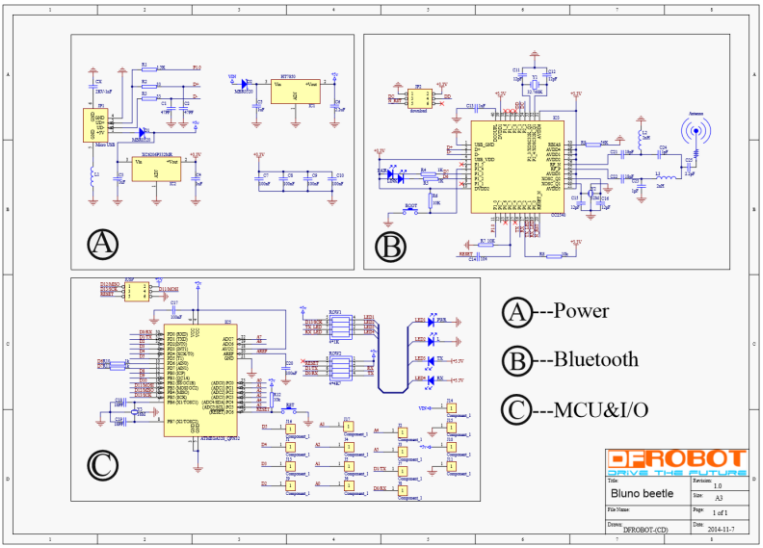
2.3. Bluno Beetle

Bluno Beetle adalah board *microcontroller* berbasis Arduino UNO terintegrasi modul BLE yang ditujukan untuk penggunaan perangkat *wearable* [7]. *Microcontroller* adalah sebuah komputer kecil dalam

sebuah IC (*integrated circuit*), yang memiliki satu atau lebih CPU, memori, dan GPIO (*general purpore input output*) yang dapat diprogram. Bluno menggunakan bahasa pemrograman dan *library* yang didasari dari bahasa pemrograman C++ dan perangkat lunak Arduino (IDE). Untuk memprogram Bluno Beetle dapat dilakukan dengan dua cara, yaitu dengan menggunakan kabel USB atau dengan cara nirkabel menggunakan Bluetooth Low Energy (BLE).



Gambar 2.3 Bluno Beetle ***[7]***



Gambar 2.4 Skematik Bluno Beetle ***[8]***

Diagram pinout Bluno Beetle dapat dilihat pada tabel 2.1 berikut ini.

Tabel 2.1 Diagram Pinout Bluno Beetle **[7]**

Silkscreen	Digital Pin	PWM Channel	Analog Channel	UART	I2C
RX	0			Serial1	
TX	1				
SDA	A4				SDA
SCL	A5				SCL
D2	2				
D3	3	3			
D4	4				
D5	5	5			
A0	A0		A0		
A1	A1		A1		
A2	A2		A2		
A3	A3		A3		

Sedangkan power interface bluno beetle dapat dilihat pada table 2.2 berikut ini.

Tabel 2.2 Power Interface Bluno Beetle **[7]**

Silkscreen	Description
VIN	external power supply<8V
5V	5V positive supply
GND	GND

Spesifikasi Bluno Beetle secara keseluruhan dapat dilihat pada table 2.3 berikut ini.

Tabel 2.3 Spesifikasi Bluno Beetle **[7]**

Bluetooth Chip	CC2540
Sensitivity	(-93dBm)
Working Temperature :	(-10 ℃ ~ +85 ℃)
Maximun Distance	50m(Open field)
Microcontroller:	ATmega328P

Clock frequency:	16 MHz
Working voltage:	5V DC
Digital Pin	x4
Analog Pin	x4
PWM Output	x2
UART interface	x1
I2C interface	x1
Micro USB interface	x1
Power port	x2

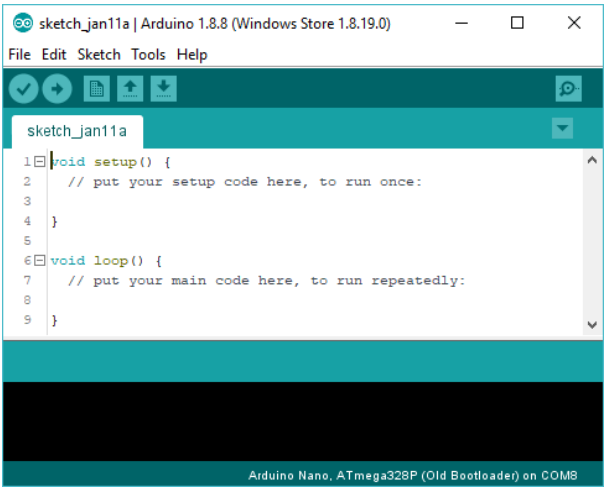
2.4. Bluetooth Low Energy (BLE)

Bluetooth Low Energy atau yang biasa disingkat BLE adalah salah satu bentuk komunikasi nirkabel yang didesain untuk komunikasi jarak dekat. BLE pada awalnya berfokus untuk mendesain sebuah standar radio dengan konsumsi daya seminimal mungkin, dioptimisasi untuk biaya rendah, *band-width* rendah, daya rendah, dan kompleksitas rendah [9].

Kebanyakan *Smartphone* dan Tablet sekarang ini mempunyai kompatibilitas dengan BLE, sehingga memungkinan BLE menjadi bagian penting dalam kehidupan sehari-hari.

2.5. Arduino IDE (Integrated Development Environment)

Arduino IDE (Integrated Development Environment) pada dasarnya diperuntukkan bagi pengguna Mikrokontroller berbasis Arduino untuk melakukan fungsi penulisan program yang tepat dan juga menyediakan teknik kompilasi yang mudah. Arduino IDE menyediakan dukungan untuk Bahasa *Embedded* C dengan penyediaan *library built-in* untuk membantu mempermudah dan menyederhanakan penulisan program. Arduino IDE menyediakan *interface* untuk mengunggah program ke dalam *microcontroller* dan juga menampilkan *display* komunikasi serial dari *microcontroller* [10]. *Software* ini dapat digunakan pada semua perangkat keras berbasis Arduino, ditambah dengan beberapa perangkat keras *microcontroller* lainnya yang kompatibel.



Gambar 2.5 Tampilan *Interface* dari *Software* Arduino IDE.

2.6. Sensor Suhu dan Temperatur HTU21

HTU21 merupakan sebuah sensor temperatur dan kelembapan. IC HTU21 ini sudah mencakup 4 buah pin hubungan yaitu Vcc, Ground, SDA, dan SCL. Perhitungan nilai kelembapan dan temperatur didapat dari:

$$RH = -6 + 125 \times \frac{S_{RH}}{2^{16}} \text{ [11]}$$

$$Temp = -46.85 + 175.72 \times \frac{S_{Temp}}{2^{16}} \text{ [11]}$$

Gambar 2.6 IC HTU21 ***[12]***

Spesifikasi keseluruhan dari IC HTU21 dapat dilihat pada tabel 2.5 dan 2.6 berikut ini.

Tabel 2.5 Spesifikasi HTU21 **[11]**

Ratings	Symbol	Value	Unit
Storage Temperature	Tstg	-40 to 125	°C
Supply Voltage (Peak)	Vcc	3.8V	Vdc
Humidity Operating Range	RH	0 to 100	%RH
Temperature Operating Range	Ta	-40 to +125	°C
VDD to GND		-0.3 to 3.6V	V
Digital I/O pins (DATA/SCK) to VDD		-0.3 to VDD+0.3	V
Input current on any pin		-10 to +10	mA

Tabel 2.6 Karakteristik kelistrikan HTU21 **[11]**

Characteristics		Symbol	Min	Typ	Max	Unit
Voltage Supply		VDD	1.5	3.0	3.6	V
Current consumption	Sleep Mode	idd		0.02	0.14	µA
	Measuring		300	450	500	µA
Power Dissipation	Sleep mode			0.06	0.5	µW
	Average 8bit			2.7		µW
Communication		digital 2-wire interface, I²C protocol				
Heater	VDD=3V	5.5mW/ΔT=+0.5-1.5°C				
Storage		-40°C/125°C				

2.7. Kelembapan Relatif

Kelembapan adalah jumlah uap air yang ada pada udara. Ada tiga jenis pengukuran utama kelembapan yang sering digunakan: kelembapan absolut, kelembapan relatif, dan kelembapan spesifik. [13] Pada tugas akhir ini, pengukuran yang digunakan adalah kelembapan relatif.

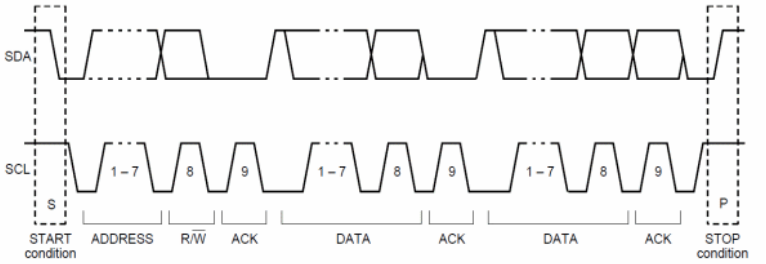
Kelembapan relatif, diekspresikan dalam persen(%), merupakan indikasi kondisi kelembapan absolut saat pengukuran terhadap kelembapan maksimum pada temperatur yang sama. Kelembapan relatif (RH atau ϕ) dari campuran air-udara didefiniskan sebagai rasio dari tekanan parsial uap air (p_{H_2O}) dalam campuran dengan kesetimbangan tekanan uap air($p^*_{H_2O}$) di atas permukaan air murni pada temperatur tertentu. [14]

$$\phi = \frac{p_{H_2O}}{p^*_{H_2O}}$$

2.8. Komunikasi I2C

Komunikasi *Inter-Integrated Circuit* atau yang biasa disingkat dengan komunikasi I2C adalah protokol komunikasi serial untuk antarmuka dua kabel untuk menghubungkan perangkat yang memiliki kecepatan rendah seperti mikrokontroler, EEPROM, A/D dan D/A *Converter.* [15] Komunikasi I2C memungkinkan banyak IC digital "*slave"* untuk berkomunikasi dengan satu atau lebih IC digital "*master"*.

Pada komunikasi I2C, data dikirimkan dalam bentuk pesan, dimana pesan ini terbagi menjadi beberapa bingkai data. Setiap pesan memiliki bingkai *address* yang menyimpan *address* biner dari IC "*slave"*, dan satu atau lebih bingkai data yang menyimpan data yang akan dikirimkan. Pesan ini juga menyimpan kondisi untuk mulai ataupun berhenti, *read/write bits*, ACK/NACK *bits* di antara setiap bingkai data.



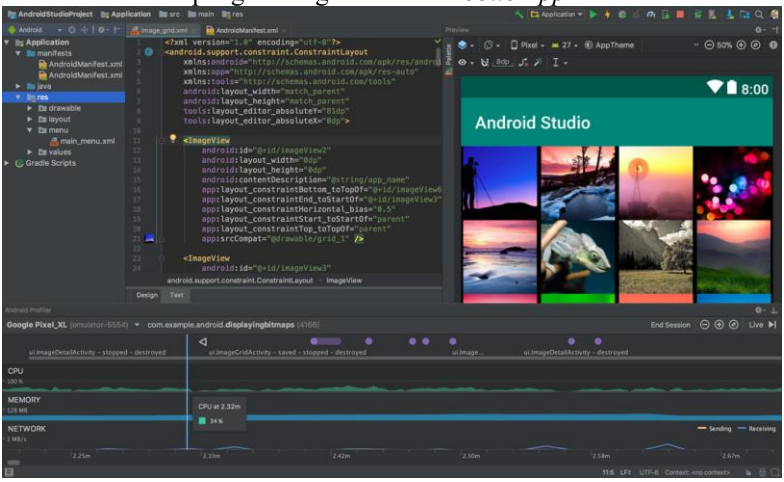
Gambar 2.7 Komunikasi I2C ***[16]***

2.9. Android Studio

Sistem yang akan direalisasikan pada penelitian ini memiliki komponen yang bernama *transceiver* dan *receiver,* dimana *transceiver* dan *receiver* ini adalah penghubung antara sensor dengan pengguna/perawat. *Transceiver* dan *receiver* pada sistem ini berupa *Mobile Application* Android yang berfungsi untuk menampilkan hasil pembacaan sensor dan mengirimkan ataupun mengambil data dari *Database* melalui jaringan internet. Proses pembuatan *Mobile App* Android ini menggunakan *Development Environment* yang bernama Android Studio.

Android Studio adalah *Integrated Development Environment* (IDE)

resmi untuk sistem operasi Android dari google, yang dibangun diatas perangkat lunak JetBrains IntelliJ IDEA dan didesain khusus untuk pengembangan Android [17]. Android Studio ini merupakan pengganti dari Eclipse Android Development Tools (ADT) yang sebelumnya adalah IDE utama untuk pengembangan sebuah *Mobile App* Android.



Gambar 2.8 *Software* Android Studio ***[18]***

Android Studio ini menggunakan Bahasa pemrograman Java dan Kotlin dan pada Android Studio yang terbaru sudah ada dukungan untuk penggunaan Bahasa C++.

2.10. AutoDesk EAGLE (Student Version)

Dalam perancangan dan pembuatan sistem dalam penelitian ini dibutuhkan sebuah PCB (*Printed Circuit* Board) untuk rangkaian elektronika dengan komponen-komponen yang dibutuhkan. Pada penelitian ini, desain dari PCB yang akan digunakan dibuat menggunakan *software* EAGLE yang merupakan salah satu produk dari AutoDesk.

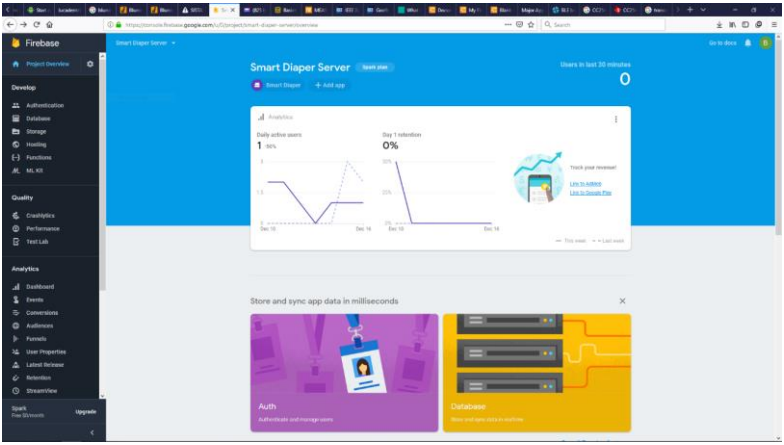
Gambar 2.9 Logo AUTODESK EAGLE ***[19]***

EAGLE merupakan *software* yang digunakan oleh banyak kalangan *engineer* maupun mahasiswa, dikarenakan cara pengoperasian EAGLE yang tergolong mudah dipahami dan ada fitur yang lengkap. Pada *software* ini dilakukan perancangan salah satu sub-sistem yaitu sensor.

2.11. Google Firebase

Sistem yang akan direalisasikan pada penelitian ini memiliki komponen yang bernama *Database*, dimana *Database* ini adalah tempat penyimpanan data hasil pembacaan sensor yang diteruskan oleh *transceiver* untuk kemudian diambil kembali datanya oleh *receiver.*

Database pada sistem ini berbasis pada Google Firebase.



Gambar 2.10 Google Firebase

Firebase adalah *Development Platform* untuk aplikasi *Mobile* dan *Web* yang dimiliki oleh Google. Firebase sendiri memiliki 18 produk dengan fungsinya masing-masing [20]. Pada penelitian ini, produk Firebase yang digunakan adalah Cloud Firestore Beta, dimana Cloud Firestore Beta ini merupakan sebuah tempat penyimpanan data yang terhubung dengan Internet, sehingga semua aplikasi yang terhubung dengan internet dapat mengakses data yang ada pada Cloud Firestore Beta tersebut.

2.12. *Moving Average*

Moving Average adalah perhitungan untuk menganalisa data dengan cara menciptakan sekumpulan rata-rata dari *subset* yang berbeda dari keseluruhan data. *Moving Average* disebut juga sebagai *Moving Mean* atau *Rolling Mean* [21] dan juga merupakan sebuah filter *finite impulse response.*

Moving Average biasanya digunakan pada data yang berjalan seiring waktu untuk menghaluskan fluktuasi jangka pendek. Batas antara jangka pendek dan jangka panjang bergantung pada aplikasi dan parameter dari *Moving Average* dapat diatur sedemikian rupa.

Gambar 2.11 *Moving Average **[22]***

Pada penelitian kali ini *Moving Average* yang digunakan adalah *Standard Moving Average* dengan rumus sebagai berikut:

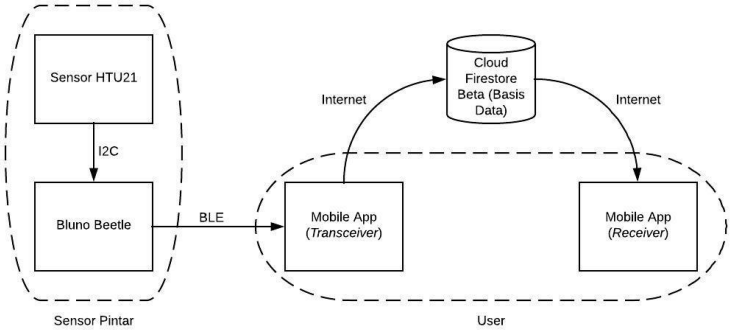
$$SMA = \frac{t1 + t2 + t3 + \cdots + tn}{n}$$

BAB III
PERANCANGAN SISTEM

Pada bab ini akan dijelaskan sistem yang dirancang secara keseluruhan, yakni perangkat keras (*hardware*) dan perangkat lunak (*software*). Perangkat keras yang digunakan dan dirancang dalam sistem ini antara lain *Smart Sensor* yang terdiri dari Sensor HTU21, Bluno Beetle, dan Baterai. Perangkat lunak yang akan diimplementasikan pada sistem ini antara lain pemrosesan hasil pembacaan sensor menggunakan *Moving Average*, pembuatan aplikasi untuk *transceiver* dan *receiver*, dan koneksi antara aplikasi dengan *database*.

3.1. Diagram Blok Sistem

Diagram blok sistem Sensor Pintar *wearable* berbasis *Internet of Things* secara keseluruhan ditunjukkan pada gambar 3.1.



Gambar 3.1 Diagram blok sistem.

Pada tugas akhir ini, digunakan Bluno Beetle sebagai tempat mengolah hasil pembacaan sensor, sedangkan *Mobile App* pada sistem digunakan sebagai tempat menunjukkan hasil pembacaan sensor dan juga mengirimkan data dari *database* ataupun mengambil data dari *database* pengolah data citra uang dengan perangkat lunak MATLAB. Perangkat keras yang digunakan dalam sistem ini meliputi:

- *Smart Sensor*
  Smart Sensor terdiri dari sensor temperatur dan kelembapan HTU21, *microcontroller* Bluno Beetle, dan satu buah baterai kancing LIR2032 sebesar 3.6v. Seluruh perangkat keras *Smart Sensor* berfungsi untuk mendeteksi tingkat kelembapan dan temperatur popok, menghaluskan hasil pembacaan sensor, mengirimkan data hasil pembacaan ke *Mobile App*.
- Bodi luar sistem
  Agar sensor tidak mudah rusak karena tertekan oleh pemakai popok dan juga untuk membuat nyaman pemakai popok, maka dibutuhkan bodi sistem yang mampu menahan tekanan dan juga tidak mengganggu pergerakan pemakai.

3.2. Perancangan Perangkat Keras

Perangkat keras pada tugas akhir ini meliputi *Smart Sensor* dan bodi luar sistem.

3.2.1. *Smart Sensor*

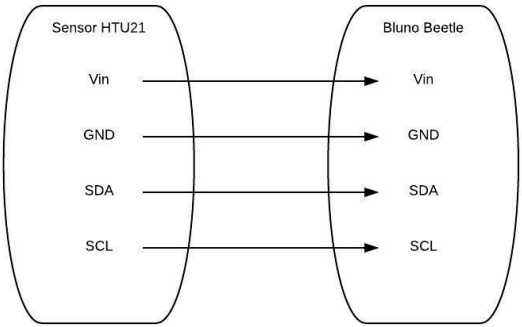
Smart Sensor dimaksudkan sebagai komponen sistem yang bertugas untuk mendeteksi tingkat kelembapan dan temperatur popok secara *realtime*, melakukan penghalusan pembacaan dengan *moving average,* mengkonversikan hasil pembacaan menjadi biner, mengirimkan data ke *mobile app*. *Smart Sensor* ini terdiri dari sensor temperatur dan kelembapan HTU21, *microcontroller* Bluno Beetle, dan satu buah baterai kancing LIR2032 3.6v.

3.2.1.1 Desain dan Rangkaian Sensor HTU21

Sensor yang digunakan pada sistem ini adalah sensor kelembapan dan temperature HTU21, sebuah IC digital yang memberikan *output digital* yang berubah-ubah seiring dengan berubahnya tingkat kelembapan dan suhu di sekitarnya, nilai yang dihasilkan berupa tingkat kelembapan dalam bentuk persen (%) dan temperatur dalam bentuk celcius.

Gambar 3.2 Sensor temperatur dan kelembapan HTU21

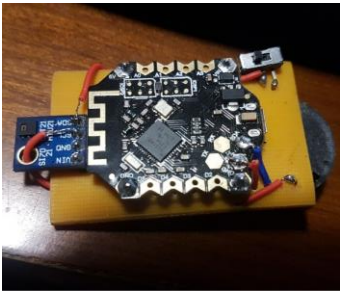
Sensor HTU21 pada sistem ini menggunakan tegangan *supply* sebesar 3.6v yang diambil dari pin Vin Bluno Beetle. Keluaran yang dihasilkan oleh HTU21 akan menjadi input digital pada sistem untuk mengetahui keadaan suhu dan temperatur di sekitar sensor



Gambar 3.3 Tata koneksi antara sensor HTU21 dengan Bluno Beetle

Dari gambar 3.3 terlihat bahwa sensor HTU21 dengan Bluno Beetle memiliki 3 macam hubungan dalam sistem, sensor disuplai dengan tegangan sebesar 3,6v dari Arduino melalui pin Vin, lalu pin GND sensor dihubungkan ke pin GND pada Bluno Beetle untuk membentuk suatu rangkaian.Output dari sensor HTU21 akan berkomunikasi dengan pin SDA dan SCL, menjadi input pada Bluno Beetle dan akan diperoleh dengan menggunakan komunikasi protocol I2C.

3.2.1.2 Bluno Beetle

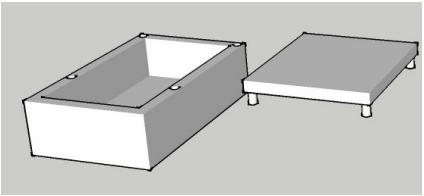


Gambar 3.4 Desain peletakkan Bluno Beetle pada *Smart Sensor*

Bluno Beetle pada sistem ini adalah sebuah *microcontroller* yang berfungsi untuk menerima data dari sensor, mengolah data yang diterima dari sensor dan menghubungkan sensor ke *mobile app* melalui BLE

3.2.2. Bodi Luar Sistem

Untuk mencegah gangguan dalam proses pembacaan temperatur dan kelembapan pada sistem dalam tugas akhir ini, maka pengambilan data harus terhindar dari udara luar yang dapat menganggu. Maka dari itu dibutuhkan suatu bodi luar yang mampu mengurangi udara luar agar tidak mengganggu kinerja sistem. Selain itu, bodi luar juga menjadi tempat penempatan seluruh komponen dalam sistem yang berfungsi untuk melindungi sensor pintar dari tekanan karena berat badan pengguna popok, baik itu bayi maupun orang dewasa. Bodi luar pada tugas akhir ini dibuat menggunakan teknik 3D Printing. Pada tampak atas bodi luar sistem terdapat lubang-lubang kecil yang berada tepat di atas sensor HTU21, hal ini memungkinkan agar udara bisa masuk dan mengenai sensor. Rancangan bodi luar sistem memiliki dimensi 6,5 x 5,2 x 3,3 cm. Desain rancangan dari bodi luar sistem ditunjukkan pada gambar 3.5.



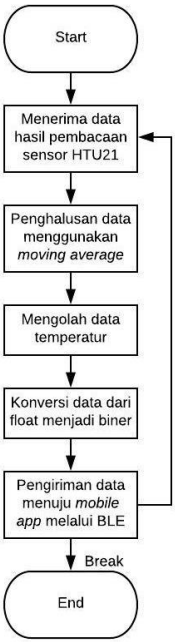
Gambar 3.5 Rancangan bodi luar sistem

3.3. Perancangan Perangkat Lunak

Perangkat lunak yang diimplementasikan pada tugas akhir ini meliputi program pada Arduino IDE yang akan menjadi program utama dari Bluno Beetle yang terdiri dari program akuisisi data temperatur dan kelembapan dari sensor kelembapan dan temperatur HTU21, pemrosesan data temperatur dan kelembapan menggunakan *moving average*, konversi data temperatur dan kelembapan dari float menjadi byte, pengiriman data menuju *mobile app*, dan program pada *mobile app* yang diciptakan menggunakan aplikasi Android Studio menggunakan bahasa pemrograman Java, dimana program terbagi menjadi dua fitur utama yaitu *transceiver* dan *receiver,* dimana *transceiver* meliputi sistem penerimaan data dari bluno, konversi data dari biner menjadi float, menampilkan data, mengirimkan data menuju *database,* dan juga memberi peringatan ketika popok sudah mencapai kondisi tertentu, sedangkan pada *receiver* meliputi program untuk mengambil data dari *database,* menampilkan data yang telah diambil, dan memberikan peringatan apabila popok sudah mencapai kondisi tertentu.

3.3.1. Perancangan Perangkat Lunak Arduino

Perancangan program utama Bluno Beetle pada Arduino IDE yang terdiri dari program akuisisi data temperatur dan kelembapan dari sensor HTU21, pemrosesan data temperatur dan kelembapan menggunakan *moving average*, konversi data temperatur dan kelembapan dari float menjadi byte, pengiriman data menuju *mobile app.* Pola kerja dari program utama sensor akan diilustrasikan oleh sebuah *flowchart* pada gambar berikut ini:



Gambar 3.6 Flowchart Program Utama Arduino

3.3.1.1 Akusisi Data Temperatur dan Kelembapan

Program akusisi data temperature dan kelembapan dilakukan melalui perangkat lunak Arduino IDE. Akusisi dilakukan oleh sensor HTU21 yang dihubungkan ke Bluno Beetle.

```
#include <SHT21.h>
SHT21 sht;
float temp;
float humidity;
void setup() {
  Wire.begin();
  Serial.begin(115200);
}

void loop() {
```

```
        temp = sht.getTemperature();
        humidity = sht.getHumidity();
      }
```

3.3.1.2 Pemrosesan Data menggunakan *Moving Average*

Pemrosesan data menggunakan *moving average* dilakukan untuk memperhalus data dari *spike* yang bisa terjadi.

```
    float tSMA;
    float t1, t2, t3, t4, t5, t6, t7, t8, t9 = 0;
    float hSMA;
    float h1, h2, h3, h4, h5, h6, h7, h8, h9 = 0;
    tSMA = (temp + t1 + t2 + t3 + t4 + t5 + t6 + t7 + t8 + t9) /
10;
      hSMA = (humidity + h1 + h2 + h3 + h4 + h5 + h6 + h7 + h8 +
h9) / 10;
      t9 = t8;
      t8 = t7;
      t7 = t6;
      t6 = t5;
      t5 = t4;
      t4 = t3;
      t3 = t2;
      t2 = t1;
      t1 = temp;
      h9 = h8;
      h8 = h7;
      h7 = h6;
      h6 = h5;
      h5 = h4;
      h4 = h3;
      h3 = h2;
      h2 = h1;
      h1 = humidity;
```

3.3.1.3 Konversi Data dari Float menjadi Byte

Setelah didapatkan data hasil pembacaan sensor yang sudah dihaluskan, data akan dikonversikan menjadi satu buah data dalam bentuk byte. Akan tetapi sebelum dikonversikan ke byte, data temperatur akan diolah lagi karena diinginkan data pembacaan temperatur ada satu angka di belakang koma. Setelah temperatur diolah, barulah data temperatur dan kelembapan diubah menjadi byte.

```
    data1 = (tSMA - 20) * 10;
```

```
data2 = hSMA;

int i = 0;
for(byte mask = 0x80; mask; mask >>=1){
  if(mask & data1)
    data[i] = '1';
  else
    data[i] = '0';
  i++;
}
i = 8;
for(byte mask = 0x80; mask; mask >>=1){
  if(mask & data2)
    data[i] = '1';
  else
    data[i] = '0';
  i++;
}
data[16] = '\0';
```

3.3.1.4 Pengiriman Data Menuju *Mobile App*

Setelah data dikonversi menjadi data sebesar 16 bit. Data dapat langsung dikirim ke *mobile app* dengan cara komunikasi serial.

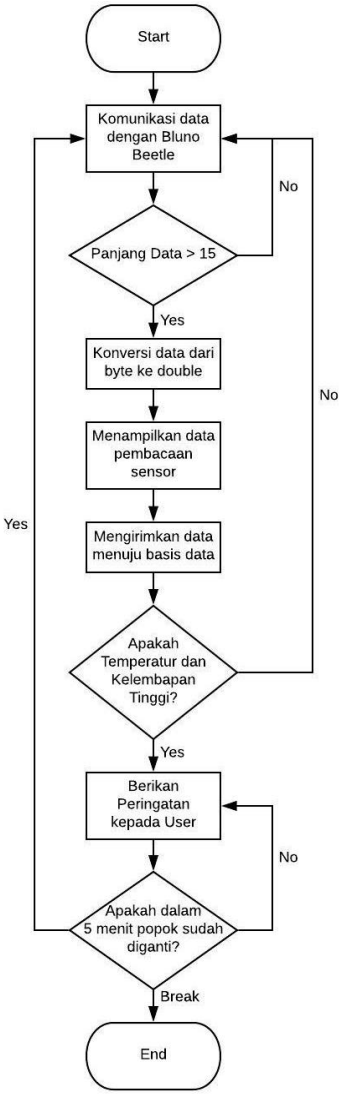
```
if(millis() - timer > 500){
     Serial.write(data);
     timer = millis();
  }
```

Data akan dikirimkan sekali setiap 500ms

3.3.2. Perancangan *Mobile App Transceiver*

Program pada *mobile app transceiver* dirancang menggunakan software Android Studio versi 3.2 dengan menggunakan Bahasa pemrograman java. Aplikasi diuji coba menggunakan perangkat *smartphone* milik penulis, yaitu ponsel Samsung Galaxy S7 Edge dengan versi android 8.0.0. Program keseluruhan meliputi sistem penerimaan data dari bluno, konversi data dari biner menjadi float, menampilkan data, mengirimkan data menuju *database,* memberikan peringatan apabila popok sudah mencapai kondisi tertentu, memberikan peringatan kedua apabila popok belum diganti dalam 5 menit.

Pola kerja program *mobile app transceiver* secara keseluruhan akan digambarkan pada *flowchart* berikut ini:



Gambar 3.7 Flowchart program utama *mobile app transceiver*.

3.3.2.1 Penerimaan Data dari Bluno Beetle

Pertama-tama yang akan diperiksa adalah jumlah panjang data yang diterima oleh *mobile app*. Apabila panjang data yang diterima lebih besar dari 15 berarti data berhasil terkirim tanpa adanya error dan program akan melanjutkan ke proses selanjutnya. Akan tetapi apabila panjang data yang diterima lebih sedikit atau sama dengan 15, berarti ada data yang hilang saat pengiriman dan aplikasi tidak akan melanjutkan ke proses selanjutnya, melainkan akan mengulang terus selama ada data serial yang diterima oleh *mobile app*.

```
@Override
public void onSerialReceived(String theString)
{                                   //Once connection data received,
this function will be called
    // TODO Auto-generated method stub
    char[] dataarray;
    double Temperature = 0;
    double Humidity = 0;

    if (theString.length() > 15) {
        //….
        }
        //….          } else {
        tempstatus.setText(R.string.tempoff);
        humidstatus.setText(R.string.humidoff);
```

3.3.2.2 Konversi Data dari Byte ke Double

Apabila data sudah diterima oleh *mobile app* langkah selanjutnya adalah dengan mengubah kembali data yang berbentuk byte menjadi double.

```
if (theString.length() > 15) {
        tempstatus.setText(R.string.tempon);
        humidstatus.setText(R.string.humidon);
        dataarray = theString.toCharArray();
        for (int i = dataarray.length - 9; i >= 0; i--) {
            if (dataarray[i] == '1') {
```

```
                Temperature = Temperature + Math.pow(2, 7 - i);

            }
        }

        for (int i = dataarray.length - 1; i > 7; i--) {
            if (dataarray[i] == '1') {
                Humidity = Humidity + Math.pow(2, 15 - i);

            }
        }

        //…
    }
```

3.3.2.3 Menampilkan Data

Setelah data dikonversi menjadi double, data temperature harus diolah terlebih dahulu dikarenakan pada saat pengiriman data temperatur dari bluno sudah bukan dalam pembacaan aslinya sehingga harus dikembalikan lagi ke bentuk aslinya. Setelah data temperatur diolah, barulah data hasil konversi tadi bisa ditampilkan.

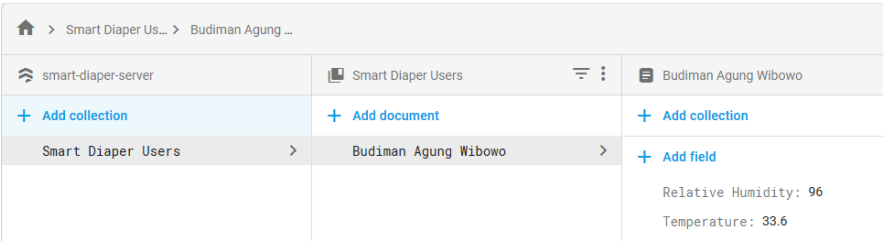
```
Temperature = (Temperature / 10) + 20;
temp.setText(String.format("%.1fc", Temperature));
humid.setText(String.valueOf((int) Humidity + "%"));
```

Gambar 3.8 Tampilan pada *mobile app transceiver*

3.3.2.4 Mengirimkan Data menuju Database

Setelah data ditampilkan, proses selanjutnya pada *mobile app* adalah mengirimkan data hasil pembacaan sensor menuju *database* melalui internet.

```
Map<String, Double> KirimData = new HashMap<String,
Double>();
KirimData.put("Temperature", Temperature);
KirimData.put("Relative Humidity", Humidity);
mDocRef.set(KirimData);
```



Gambar 3.9 Hasil penyimpanan data pada Cloud Firestore Beta

3.3.2.5 Memberikan Peringatan kepada User

Selagi semua proses yang sudah disebutkan berjalan terus, apabila *mobile app* mendeteksi tingkat kelembapan dan temperature yang tinggi (pada kasus ini dimisalkan batas temperatur = 30 dan kelembapan = 60%) maka *mobile app* akan berbunyi dan menampilkan notifikasi sebagai bentuk pengingat kepada user.

```
if (Humidity > 60 && Temperature > 30) {

    if(state == 0) {
    ChangeDiaper();
}
}

public void ChangeDiaper(){

                    if(alertb != null && alertb.isShowing() ) return;
                    Alarm.start();
                    AlertDialog.Builder b_builder = new
AlertDialog.Builder(MainActivity.this);
                        b_builder.setTitle("Time to Change
Diaper")
                                .setMessage(R.string.gantipopok
)
                                .setCancelable(false)
                                .setPositiveButton("Close", new
DialogInterface.OnClickListener() {
                                    @Override
                                    public void
onClick(DialogInterface dialogInterface, int i) {

dialogInterface.cancel();
                                        Alarm.stop();
                                    }
                                });
                        alertb = b_builder.create();
        if (!alertb.isShowing()) {
            alertb.show();
        }
```

}

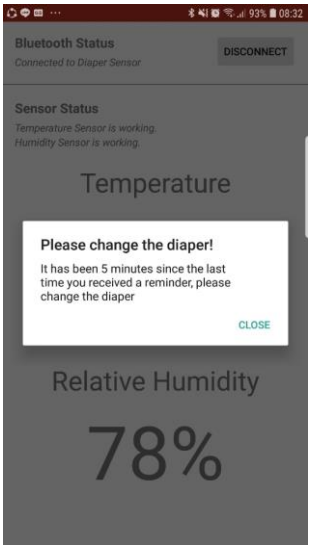
Gambar 3.10. Tampilan *mobile app* ketika muncul notifikasi

3.3.2.6 Memberikan peringatan kedua

Apabila dalam 5 menit setelah pemberian peringatan pertama popok masih belum diganti, maka akan diberikanan peringatan kedua, lalu apabila 5 menit setelah peringatan kedua popok masih belum diganti, akan diberikan peringatan selanjutnya, dan akan begitu seterusnya sampai popok diganti

```
Handler timerHandler = new Handler();
Runnable timerRunnable = new Runnable() {
    @Override
    public void run() {
        long millis = System.currentTimeMillis() - startTime;
        int seconds = (int) (millis / 1000);
        int minutes = seconds / 60;
        seconds = seconds % 60;
```

```
                timerHandler.postDelayed(this, 500);
                if (minutes >= 5) {
                    if (state == 1) {
                        Reminder();
                    }
                }
            }
        };

public void Reminder(){

            if(reminder != null && reminder.isShowing() ) return;
            Buzzer.start();
            Buzzer.setLooping(true);
            AlertDialog.Builder b_builder = new
AlertDialog.Builder(MainActivity.this);
            b_builder.setTitle("Please change the diaper!")
                    .setMessage("It has been 5 minutes since the last
time you received a reminder, please change the diaper")
                    .setCancelable(false)
                    .setPositiveButton("Close", new
DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface
dialogInterface, int i) {
                            dialogInterface.cancel();
                            Buzzer.stop();
                            Buzzer.prepareAsync();
                            startTime = System.currentTimeMillis();
                        }
                    });
            reminder = b_builder.create();
            if (!reminder.isShowing()) {
                reminder.show();
            }
        }
```

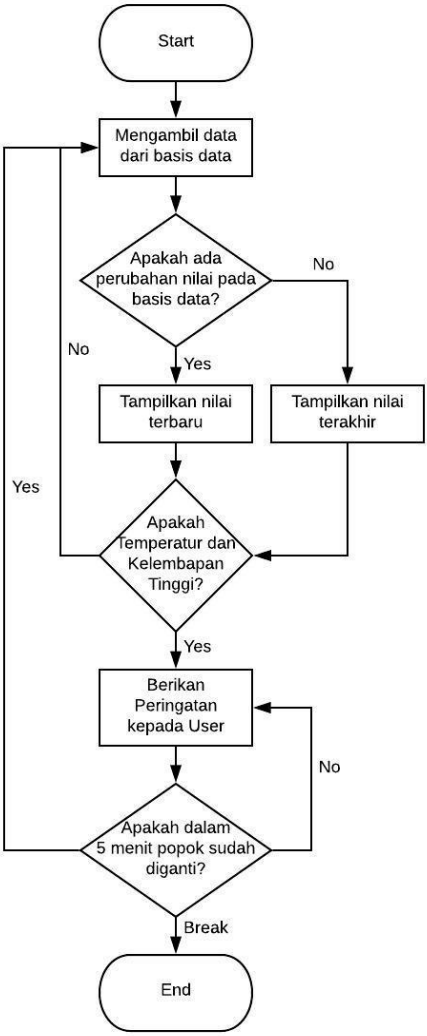


Gambar 3.11. Tampilan *mobile app* ketika muncul notifikasi kedua

3.3.3. Perancangan *Mobile App Receiver*

Program pada *mobile app receiver* dirancang menggunakan software Android Studio versi 3.2 dengan menggunakan Bahasa pemrograman java. Aplikasi diuji coba menggunakan perangkat *smartphone* milik penulis, yaitu ponsel Samsung Galaxy S7 Edge dengan versi android 8.0.0. Program keseluruhan meliputi sistem pengambilan data dari *database*, menampilkan data, memberikan peringatan apabila popok sudah mencapai kondisi tertentu, dan memberikan notifikasi kedua apabila belum diganti dalam 5 menit.

Pola kerja program *mobile app receiver* secara keseluruhan akan digambarkan pada *flowchart* berikut ini:



Gambar 3.11 Flowchart program utama *mobile app receiver*

3.3.3.1 Pengambilan Data dari Database

Pertama-tama *mobile app* membaca data-data yang tersimpan di *database* apabila tidak ada perubahan nilai pada database maka akan ditampilkan nilai yang terakhir diambil, akan tetapi apabila dideteksi ada perubahan pada nilai-nilai tersebut, maka yang akan ditampilkan adalah nilai-nilai yang paling baru.

```
mDocRef.addSnapshotListener(this, new
EventListener<DocumentSnapshot>() {
    @Override
    public void onEvent(@Nullable DocumentSnapshot
documentSnapshot, @Nullable FirebaseFirestoreException e) {
        if (documentSnapshot.exists()) {
            intstatus.setText("Connected to Databank");
            long Humidityint =
documentSnapshot.getLong("Relative Humidity");
            Double Temperatureint =
documentSnapshot.getDouble("Temperature");
            //…
        } else {
            intstatus.setText("Not Connected to Databank");
            humidint.setText("00%");
            tempint.setText("00.0c");
        }
    }
});
```

3.3.3.2 Menampilkan Data

Setelah *mobile app* mengambil data dari Cloud Firestore, proses selanjutnya adalah untuk menampilkan data yang telah diambil tersebut

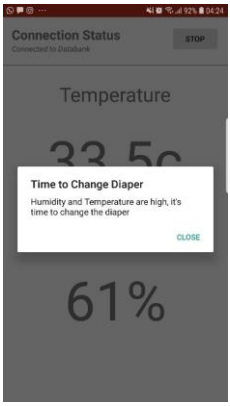
```
humidint.setText(Humidityint + "%");
tempint.setText(Temperatureint + "c");
```

3.3.3.3 Memberikan Peringatan kepada User

Setelah data diambil dan ditampilkan, sama seperti pada *mobile app*

transceiver, dilakukan pengecekan apakah temperatur dan kelembapan sudah mencapai nilai yang tinggi (dalam kasus ini temperatur = 30 dan kelembapan = 60%). Apabila sudah melebihi nilai tersebut, aplikasi akan berbunyi dan memberikan notifikasi sebagai bentuk pengingat kepada user.

```
if (Humidityint > 60 && Temperatureint > 30) {

    if(state == 0) {
        ChangeDiaper();
    }
}

public void ChangeDiaper(){

        if(alertb != null && alertb.isShowing() ) return;
                Alarm.start();
        AlertDialog.Builder b_builder = new
AlertDialog.Builder(SecondActivity.this);
        b_builder.setTitle("Time to Change Diaper")
                .setMessage(R.string.gantipopok)
                .setCancelable(false)
                .setPositiveButton("Close", new
DialogInterface.OnClickListener() {
                    @Override
                    public void onClick(DialogInterface
dialogInterface, int i) {
                        dialogInterface.cancel();
                            Alarm.stop();
                    }
                });
        alertb = b_builder.create();
        if (!alertb.isShowing()) {
            alertb.show();
        }

    }
```



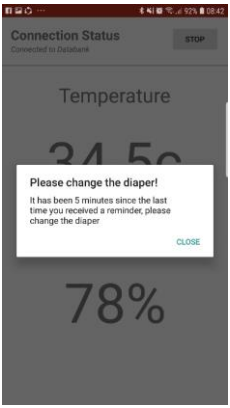
Gambar 3.12 Tampilan *mobile app* ketika muncul notifikasi

3.3.3.4 Memberikan peringatan kedua

Apabila dalam 5 menit setelah pemberian peringatan pertama popok masih belum diganti, maka akan diberikanan peringatan kedua, lalu apabila 5 menit setelah peringatan kedua popok masih belum diganti, akan diberikan peringatan selanjutnya, dan akan begitu seterusnya sampai popok diganti

```
Handler timerHandler = new Handler();
 Runnable timerRunnable = new Runnable() {
            @Override
            public void run() {
                long millis = System.currentTimeMillis() -
startTime;
                int seconds = (int) (millis / 1000);
                int minutes = seconds / 60;
                seconds = seconds % 60;

                timerHandler.postDelayed(this, 500);
                if (minutes >= 5) {
                    if (state == 1) {
                        Reminder();
```

```
                    }
                }
            }
        };

    public void Reminder(){

            if(reminder != null && reminder.isShowing() ) return;
            Buzzer.start();
            Buzzer.setLooping(true);
            AlertDialog.Builder b_builder = new
AlertDialog.Builder(SecondActivity.this);
            b_builder.setTitle("Please change the diaper!")
                    .setMessage("It has been 5 minutes since the
last time you received a reminder, please change the diaper")
                    .setCancelable(false)
                    .setPositiveButton("Close", new
DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface
dialogInterface, int i) {
                            dialogInterface.cancel();
                            Buzzer.stop();
                            Buzzer.prepareAsync();
                            startTime =
System.currentTimeMillis();
                        }
                    });
            reminder = b_builder.create();
            if (!reminder.isShowing()) {
                reminder.show();
            }
        }
```



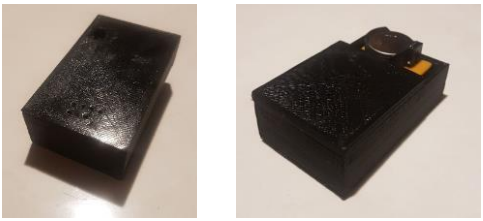
Gambar 3.12 Tampilan *mobile app* ketika muncul notifikasi kedua

BAB IV
PENGUJIAN DAN ANALISIS

Pada bab ini akan dibahas mengenai pengujian dari sistem yang telah dirancang pada bab sebelumnya. Bab ini bertujuan untuk mendapatkan data yang kemudian dilakukan analisa pada masing-masing pengujian. Pengujian yang dilakukan meliputi pengujian sensor HTU21, pengujian metode *moving average*, pengujian supply tegangan, pengujian sistem secara keseluruhan, pengambilan data pada popok, dan pengujian sistem sensor pintar *wearable* berbasis i*nternet of things.*

4.1. Realisasi Sistem

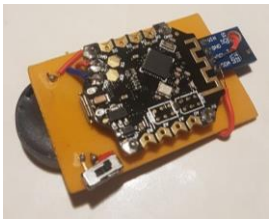
Pada subbab ini akan ditunjukkan realisasi sistem secara keseluruhan.



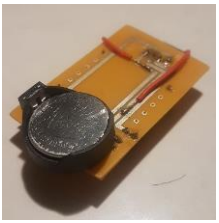
(a) (b)

Gambar 4.1 Realisasi *Smart Sensor* secara keseluruhan: (a) tampak atas dan (b) tampak bawah.

Pada gambar 4.1 diperlihatkan realisasi desain fisik *smart sensor* pada sistem. *Case* pada *smart sensor* yang menjadi wadah bagi rangkaian elektroniknya terbuat dari bahan plastik. Desain *case* pada *smart sensor* dilakukan per bagian secara 3 Dimensi menggunakan Aplikasi SketchUp. Setelah desain per bagian selesai, maka dilakukan pencetakan desain yang telah dibuat menggunakan teknologi *3D Printing*. Bagian-bagian yang sudah dicetak tadi diletakkan satu sama lain membentuk sebuah wadah seperti yang sudah diperlihatkan pada gambar 4.1



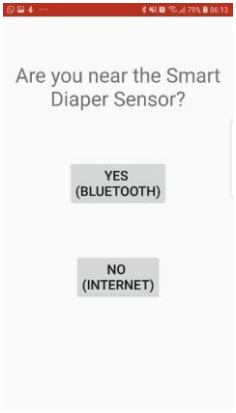
(a)



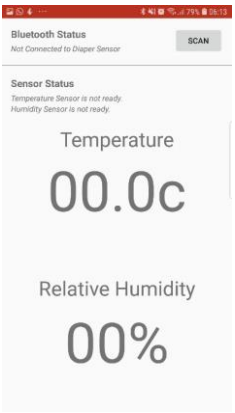
(b)

Gambar 4.2 Tata pengaturan rangkaian elektronika pada *smart sensor*: (a) tampak atas dan (b) tampak bawah.

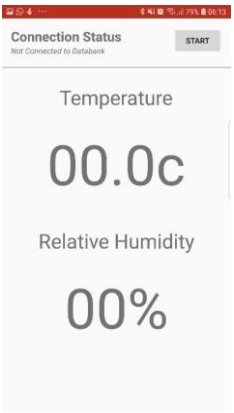
Gambar 4.2 memperlihatkan realisasi dari perancangan sistem elektronik pada *smart sensor.* Bluno Beetle diletakkan di tengah sebagai pusat dari *smart sensor*. Bluno Beetle kemudian dihubungkan dengan sensor HTU21 di sampingnya sebagai sumber pengambilan data. Kemudian di bagian lainnya lagi, Bluno Beetle dipasangkan dengan *switch* geser yang menghubungkan baterai kancing 3,6v dengan rangkaian.



(a)



(b)



(c)

Gambar 4.3 Realisasi tampilan Aplikasi Android *"Smart Diaper"*: (a) Menu Utama, (b) *Transceiver mode* dan (c) *Receiver mode.*

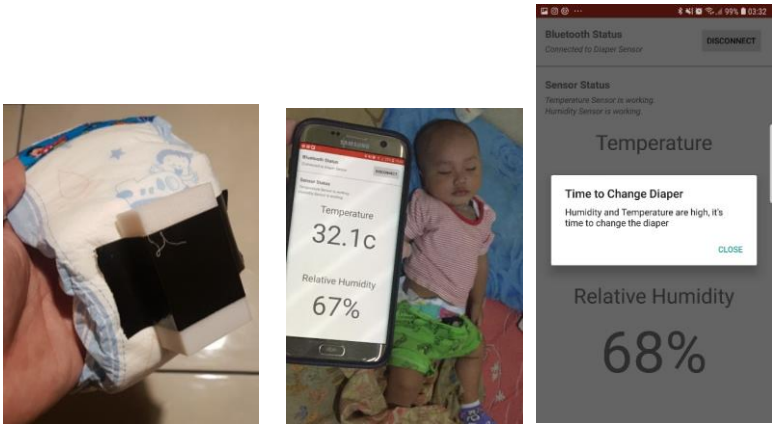
Gambar 4.3 memperlihatkan tampilan dari aplikasi android yang telah dibuat yaitu *"Smart Diaper"*. Aplikasi ini memiliki 3 halaman yaitu menu utama, *transceiver mode,* dan *receiver mode.* Menu utama adalah

hal yang pertama kali ditemui pada saat kita membuka aplikasi *"Smart Diaper",* pada menu ini user akan dihadapkan dengan dua pilihan dimana kedua pilihan ini akan mengarahkan user ke mode *transceiver* ataupun *receiver*. Mode *transceiver* adalah halaman yang akan ditemui apabila user menekan tombol "YES (BLUETOOTH)" pada menu utama, pada halaman ini user hanya akan dihadapkan dengan pilihan untuk menghubungkan aplikasi dengan *smart sensor*, setelah terhubung user dapat memantau kondisi popok secara *realtime.* Mode *receiver* adalah halaman yang akan ditemui apabila user menekan tombol "NO (INTERNET)" pada menu utama, pada halaman ini user hanya akan dihadapkan dengan satu tombol saja, dimana tombol ini memiliki fungsi untuk memulai pengambilan data melalui internet secara *realtime.*

(a) (b) (c)

Gambar 4.4 Posisi peletakkan sensor pintar pada popok: a. depan popok, b. bawah popok, c. belakang popok.

Gambar 4.4 menunjukkan peletakkan sensor pintar pada popok, dimana sensor pintar ditempelkan pada popok dan direkatkan menggunakan perekat jenis apapun. Sensor pintar dapat diletakkan di bagian manapun pada popok, dengan posisi sensor HTU21 berada dekat dengan bagian bawah popok.



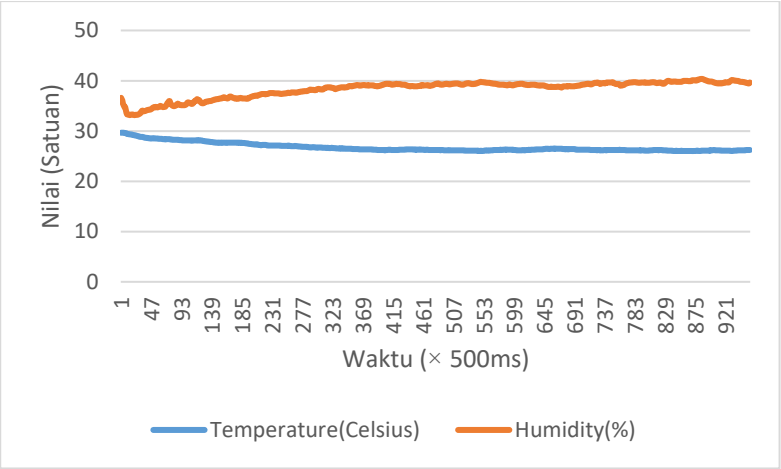
(1) (2) (3)

Gambar 4.5 Skenario pemakaian sensor pintar untuk monitoring popok.

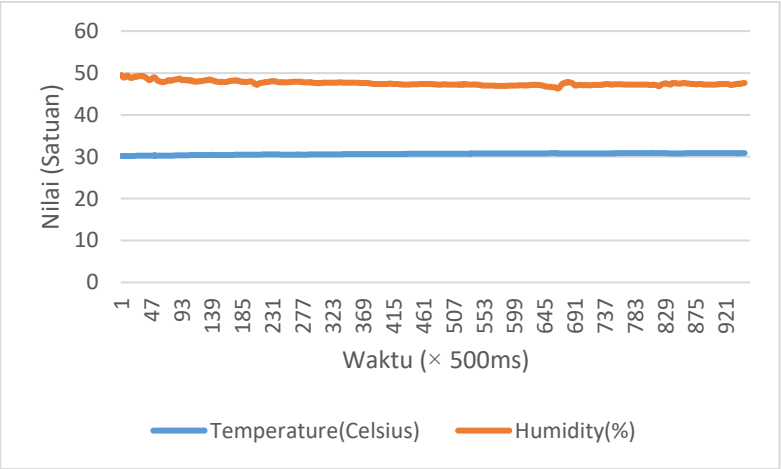
Pada gambar 4.5 diperlihatkan skenario penggunaan sensor pintar pada subjek. Gambar 4.5 (1) menunjukkan pemasangan sensor pintar pada popok menggunakan perekat. Gambar 4.5 (2) menunjukkan penggunaan sistem keseluruhan dimana sensor pintar yang sudah direkatkan pada popok akan dikenakan oleh bayi, lalu sensor tersebut dihubungkan dengan aplikasi untuk mulai melakukan kegiatan monitoring. Gambar 4.5 (3) menunjukkan tampilan aplikasi ketika hasil pembacaan tingkat kelembapan dan temperatur pada popok bayi sudah melewati 60% dan 33^0C, dimana pada saat itu aplikasi akan memberikan notifikasi dan membunyikan alarm sebagai bentuk pengingat untuk mengganti popok.

4.2. Pengujian Sensor HTU21

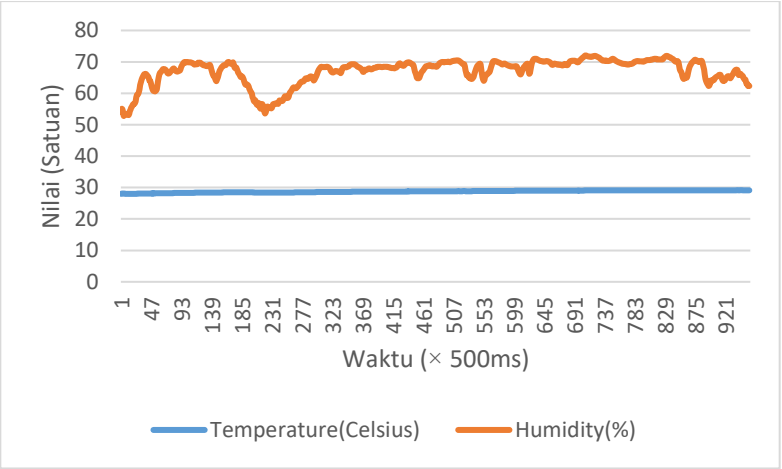
Sensor HTU21 digunakan dalam deteksi tingkat kelembapan dan temperatur. Pengujian sensor HTU21 perlu dilakukan untuk mengetahui apakah pembacaan sensor HTU21 sesuai dengan kebutuhan atau tidak. Pengujian dilakukan dengan cara mengukur *output* HTU21 yang terukur pada berbagai kondisi. Pengukuran besaran dilakukan dengan mengamati serial monitor Arduino IDE yang menampilkan *output* sensor dengan *baud rate* 115200 bps selama 8 menit dan 20 detik. Hasil pengujian sensor HTU21 dapat dilihat di bawah ini.



Gambar 4.6 Hasil pembacaan HTU21 pada ruangan dengan AC



Gambar 4.7 Hasil pembacaan HTU21 pada ruangan tanpa AC



Gambar 4.8 Hasil pembacaan HTU21 pada kamar mandi

Tabel 4.1 Hasil pembacaan terakhir sensor HTU21

No.	Kondisi	Suhu (°c)	Kelembapan (%)
1.	Ruangan dengan AC	26,2	40
2.	Ruangan tanpa AC	30,8	48
3.	Kamar Mandi	29.1	62

Dari hasil pembacaan dapat disimpulkan bahwa *output* sudah sesuai seperti dengan yang diinginkan, yaitu perubahan pembacaan suhu dari ruangan tanpa AC dengan ruangan dengan AC, dan juga perubahan tingkat kelembapan pada ruangan biasa dengan kamar mandi.

4.3. Pengujian Subsistem sensor pintar

Pada tahap pengujian subsistem sensor pintar ini akan dilakukan pengujian pada subsistem sensor pintar, dimana akan dilakukan pengamatan terhadap hasil pembacaan sensor pintar pada berbagai kondisi. Kondisi pengujian adalah kamar dengan AC, kamar tanpa AC, dan kamar mandi. Pengujian dilakukan selama 8 menit dan 20 detik. Hasil pembacaan sensor dapat dilihat pada tabel berikut.

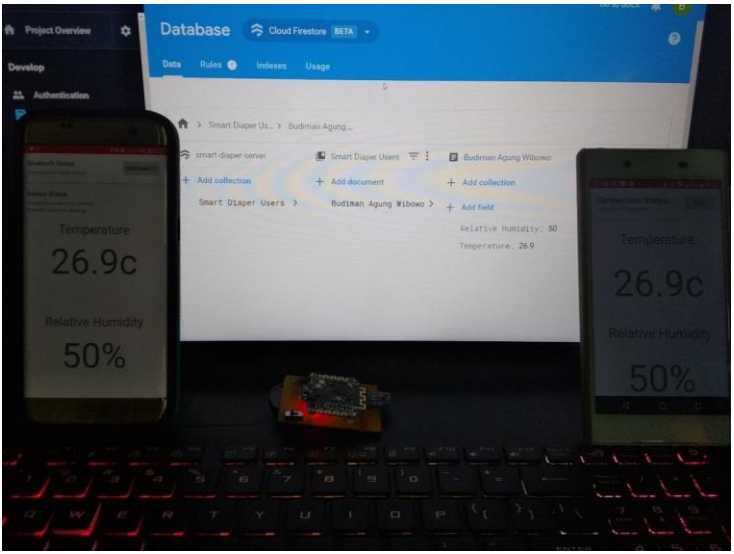
Tabel 4.2 Hasil pembacaan sensor pintar

No.	Kondisi	Suhu (°c)	Kelembapan (%)
1.	Ruangan tanpa AC	27,6	45
2.	Ruangan dengan AC	24,4	39
3.	Kamar Mandi	30.3	51

Dari hasil pembacaan dapat ditarik kesimpulan bahwa packaging tidak mempengaruhi hasil pembacaan kelembapan dan temperatur.

4.4. Pengujian Komunikasi Antar Subsistem

Pada pengujian komunikasi antar sistem akan dilakukan pengamatan data yang ditampilkan pada tiap subsistem dan perbandingan hasil data yang ditampilkan pada *transceiver, database,* dan *receiver* pada satu waktu. Hasil perbandingan data yang ditampilkan antara *transceiver, database,* dan *receiver* dapat dilihat pada gambar berikut



Gambar 4.9 Perbandingan data yang ditampilkan pada *transceiver,* basis data, dan *receiver.*

Berdasarkan hasil pengamatan penulis, data yang ditampilkan antara aplikasi *transceiver*, basis data, aplikasi *receiver* adalah sama. Akan tetapi selama pengamatan, terjadi *delay* komunikasi data sedikit.

4.5. Pengambilan Data Pada Popok

Pada tahap ini dilakukan pengambilan data secara langsung pada popok yang digunakan. Pengambilan data dilakukan dengan cara menempelkan *smart sensor* kepada popok. Tujuan pengambilan data adalah untuk menentukan batas bawah kondisi kotor popok dan juga untuk melihat pengaruh kondisi luar terhadap pembacaan. Pengujian pertama adalah pada popok dewasa yang kering, dimana pengukuran akan dilakukan pada ruangan tanpa AC, ruangan dengan AC, dan kamar mandi. Hasil pembacaan pada popok dewasa dengan berbagai kondisi ditunjukkan pada tabel berikut.

Tabel 4.3 Pengukuran Popok Dewasa

No.	Kondisi	Suhu (°c)	Kelembapan (%)
1.	Kamar Mandi	32,6	53
2.	Ruangan tanpa AC	33,5	48
3.	Ruangan dengan AC	33,0	49

Dari tabel di atas dapat dilihat bahwa tidak terjadi perubahan tingkat kelembapan dan temperatur yang signifikan. Sehingga dapat ditarik kesimpulan bahwa tingkat kelembapan dan temperatur di dalam celana berbeda dengan tingkat kelembapan dan temperatur udara luar.

Pengukuran yang kedua dilakukan pada popok bayi, sampel yang digunakan ada 3 orang anak. Dimana dilakukan pemasangan *smart sensor* pada popok bayi selama 8 menit. Hasil pembacaan sensor pada popok bayi ditunjukkan pada tabel berikut.

Tabel 4.4 Pengujian Popok Bayi

Subjek	Kondisi Popok	Suhu (°c)	Kelembapan (%)	Kondisi Pengukuran
Aisyah (1 Tahun)	Kotor	33,8	67	Berlari
	Bersih	33,7	56	Berlari
Riskina (2 Tahun)	Kotor	33.7	61	Tertidur
	Bersih	35,8	53	Tertidur

Syahrul (8 Bulan)	Kotor	33,5	66	Tertidur
	Bersih	33,3	54	Tertidur

Dari tabel di atas penulis mengambil nilai-nilai terendah sebagai parameter penentuan kondisi popok mulai berisi. Sehingga nilai-nilai yang diambil sebagai parameter adalah sebagai berikut:

Temperatur = 33,0

Kelembapan = 60

4.6. Pengujian Sistem Sensor Pintar *Wearable* Berbasis *Internet of Things*

Setelah didapat batas bawah dari pengujian sebelumnya, batas bawah langsung diimplementasikan ke dalam program aplikasi android. Pengujian keseluruhan sistem dilakukan dengan cara memasang *smart sensor* pada popok kemudian diberikan cairan dan membuktikan apakah aplikasi android akan memberikan notifikasi ketika diberikan air urin. Pengujian dilakukan menggunakan 1 merek popok dewasa dan 4 merek popok bayi. Hasil pengujian sistem sensor pintar dapat dilihat pada tabel berikut:

Tabel 4.5 Pengujian Sistem Sensor Pintar berbasis IoT

Jenis Popok	Jumlah Pengujian	Jumlah Keberhasilan	Persentase Keberhasilan
Indomaret(Popok Dewasa)	5	4	80%
Sweety(Popok Bayi)	5	4	80%
Popoku(Popok Bayi)	5	3	60%
Merries(Popok Bayi)	5	3	60%
Mamypoko(Popok Bayi)	5	4	80%
Total	25	18	72%

.....Halaman ini sengaja dikosongkan.....

BAB V
PENUTUP

5.1. Kesimpulan

Berdasarkan percobaan yang telah dilakukan pada pelaksanaan tugas akhir ini didapat beberapa kesimpulan sebagai berikut:

a. Berdasarkan pengamatan, sensor pintar memberikan sedikit ketidaknyamanan apabila diletakkan di bawah popok, dan tidak memberikan ketidaknyamanan apabila diletakkan di bagian depan ataupun belakang
b. Tingkat kelembapan dan temperatur di dalam celana berbeda dengan tingkat kelembapan dan temperatur udara luar sehingga sensor pintar tepat digunakan pada kondisi tersebut
c. Sistem sensor pintar *wearable* berbasis *Internet of Things* yang dibuat mampu memberikan pengingat ketika tingkat kelembapan dan temperatur melewati batas tertentu
d. Sensor pintar mampu membedakan air biasa dengan air urin

5.2. Saran

Sebagai sarana pengembangan Sensor pintar *wearable* berbasis *Internet of Things*, maka terdapat beberapa saran dari penulis berdasarkan hasil yang diperoleh saat percobaan, yaitu sebagai berikut:

a. Pengujian dan analisa lebih lanjut dengan jumlah dan variasi data lebih banyak untuk meningkatkan akurasi
b. Menggunakan *fuzzy logic* agar notifikasi lebih akurat terhadap kondisi kelembapan dan temperatur popok
c. Pembuatan subsistem sensor pintar yang lebih kecil dari sebelumnya agar pengguna popok lebih merasa nyaman.
d. Melakukan pengujian dan pengambilan data pada popok dewasa

.....Halaman ini sengaja dikosongkan.....

DAFTAR PUSTAKA

[1] H. T. Shin, "Diagnosis and Management of Diaper Dermatitis," Pediatric Clinics of North America, vol. 61, no. 2, pp. 367-382, 2014.

[2] M. F. John Mersch, "Diaper Rash," [Online]. Available: https://www.medicinenet.com/diaper_rash/article.htm#what_causes_diaper_rash. [Accessed 31 Agustus 2018].

[3] J. O'Donoghue and J. Herbert, "Data Management within mHealth Environments: Patient Sensors, Mobile Devices, and Databases," Journal of Data and Information Quality, vol. 4, no. 1, 2012.

[4] Tehrani, Kiana and A. Michael, "Introduction to Wearable Technology," [Online]. Available: http://www.wearabledevices.com/what-is-a-wearable-device/. [Accessed 31 Agustus 2018].

[5] R. Zafalon, "Smart System Design: Industrial Challenges and Perspectives," 2013 IEEE 14th International Conference on Mobile Data Management, vol. 1, pp. 3-3, 2013.

[6] A. Banafa, "Datafloq," [Online]. Available: https://datafloq.com/read/internet-of-things-more-than-smart-things/1060. [Accessed 29 November 2018].

[7] "DFRobot," 29 June 2017. [Online]. Available: https://www.dfrobot.com/wiki/index.php/Bluno_Beetle_SKU:DFR0339. [Accessed 2 December 2018].

[8] "RobotShop," [Online]. Available: https://www.robotshop.com/media/files/pdf/DFR0339-Bluno-beetle-V1.0.pdf. [Accessed 2 December 2018].

[9] R. Davidson, Akiba, C. Cufi and K. Townsend, Getting Started with Bluetooth Low Energy, O'Reilly Media, Inc., 2014.

[10] "Arduino Software (IDE)," 07 September 2015. [Online]. Available: https://www.arduino.cc/en/Guide/Environment. [Accessed 3 December 2018].

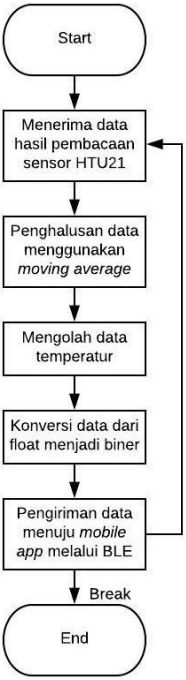
[11] "Datasheet HTU21D RH/T SENSOR," TE Connectivity, [Online]. Available: https://www.te.com/commerce/DocumentDelivery/DDEController?Action=showdoc&DocId=Data+Sheet%7FHPC199_6%7FA6%7Fpdf%7FEnglish%7FENG_DS_HPC199_6_A6.pdf%7FHPP845E031. [Accessed 4 December 2018].

[12] "HTU21D RH/T SENSOR," TE Connectivity, [Online]. Available:

https://www.te.com/usa-en/product-HPP845E031.html. [Accessed 2 December 2018].
[13] R. H. Perry and D. W. Green, "Psychrometry, Evaporative Cooling and Solids Drying," in Perry's Chemical Engineers' Handbook (8th Edition), McGraw-Hill, 2007.
[14] D. Lide, in CRC Handbook of Chemistry and Physics (85th Edition), CRC Press, 2005, pp. 15-25.
[15] "I2C Info," [Online]. Available: http://i2c.info/. [Accessed 4 December 2018].
[16] "I2C Bus Specification," [Online]. Available: http://i2c.info/i2c-bus-specification. [Accessed 4 December 2018].
[17] X. Ducrohet, T. Norbye and K. Chou, "Android Studio: An IDE built for Android," 15 May 2013. [Online]. Available: https://android-developers.googleblog.com/2013/05/android-studio-ide-built-for-android.html. [Accessed 4 December 2018].
[18] "Android Studio," [Online]. Available: https://developer.android.com/studio/. [Accessed 4 December 2018].
[19] S. Sattel, "Autodesk EAGLE has landed: The easy PCB design software – now under the Autodesk wing," AUTODESK, [Online]. Available: https://www.autodesk.com/products/eagle/blog/autodesk-eagle-has-landed-the-easy-pcb-design-software-now-under-the-autodesk-wing/. [Accessed 4 December 2018].
[20] "Google Firebase Products," [Online]. Available: https://firebase.google.com/products/. [Accessed 4 December 2018].
[21] E. Booth, J. Mount and J. H. Viers, "Hydrologic Variability of the Cosumnes River Floodplain," San Francisco Estuary and Watershed Science, vol. 4, no. 2, 2016.
[22] "Moving Averages," [Online]. Available: http://cns.bu.edu/~gsc/CN710/fincast/Technical%20_indicators/Moving%20Averages.htm. [Accessed 4 December 2018].

LAMPIRAN A

Program Arduino Sensor Pintar



Program ini adalah program yang ada pada Bluno Beetle, dimana program ini bertujuan untuk menerima hasil pembacaan sensor, mengolahnya, dan kemudian mengirimkan hasilnya ke aplikasi melalui Bluetooth Low Energy. Pertama, Bluno Beetle akan menerima hasil pembacaan dari sensor HTU21. Setelah hasil pembacaan tersebut diterima, maka hasil pembacaan tersebut akan diperhalus menggunakan moving average. Setelah data diperhalus, maka data pembacaan temperatur akan diolah menggunakan perhitungan $Temp = (Temp_{SMA} - 20) \times 10$ dikarenakan pembacaan temperatur diinginkan memiliki 1 angka dibelakang koma. Setelah data temperatur diolah, maka hasil pembacaan kelembapan dan temperatur akan diubah menjadi bentuk byte untuk memudahkan pengiriman. Setelah data diubah, barulah data tersebut dikirimkan menuju aplikasi melalui Bluetooth Low Energy.

Program:

```
#include <SHT21.h>

SHT21 sht;

float temp;
float tSMA;
float t1, t2, t3, t4, t5, t6, t7, t8, t9 = 0;
float humidity;
float hSMA;
float h1, h2, h3, h4, h5, h6, h7, h8, h9 = 0;
byte data1 = 0;
byte data2 = 0;
char data[16];
long timer = 0;

void setup() {
  Wire.begin();
  Serial.begin(115200);
}

void loop() {

  temp = sht.getTemperature();
  humidity = sht.getHumidity();
  tSMA = (temp + t1 + t2 + t3 + t4 + t5 + t6 + t7 + t8 + t9) / 10;
  hSMA = (humidity + h1 + h2 + h3 + h4 + h5 + h6 + h7 + h8 + h9) / 10;
  t9 = t8;
  t8 = t7;
  t7 = t6;
  t6 = t5;
  t5 = t4;
  t4 = t3;
  t3 = t2;
  t2 = t1;
  t1 = temp;
  h9 = h8;
```

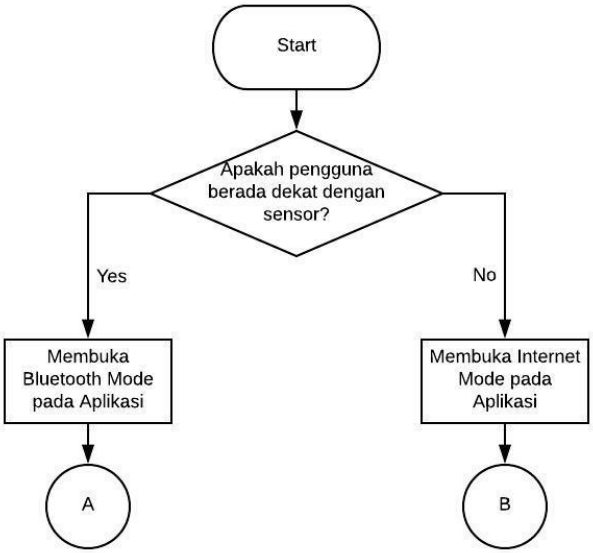
```
    h8 = h7;
    h7 = h6;
    h6 = h5;
    h5 = h4;
    h4 = h3;
    h3 = h2;
    h2 = h1;
    h1 = humidity;

    data1 = (tSMA - 20) * 10;
    data2 = hSMA;

    int i = 0;
    for(byte mask = 0x80; mask; mask >>=1){
       if(mask & data1)
          data[i] = '1';
       else
          data[i] = '0';
       i++;
    }
    i = 8;
    for(byte mask = 0x80; mask; mask >>=1){
       if(mask & data2)
          data[i] = '1';
       else
          data[i] = '0';
       i++;
    }
    data[16] = '\0';

    if(millis() - timer > 500){
          Serial.print(data);
          timer = millis();
    }
}
```

.....Halaman ini sengaja dikosongkan.....

LAMPIRAN B

Program Mobile App

Main Menu

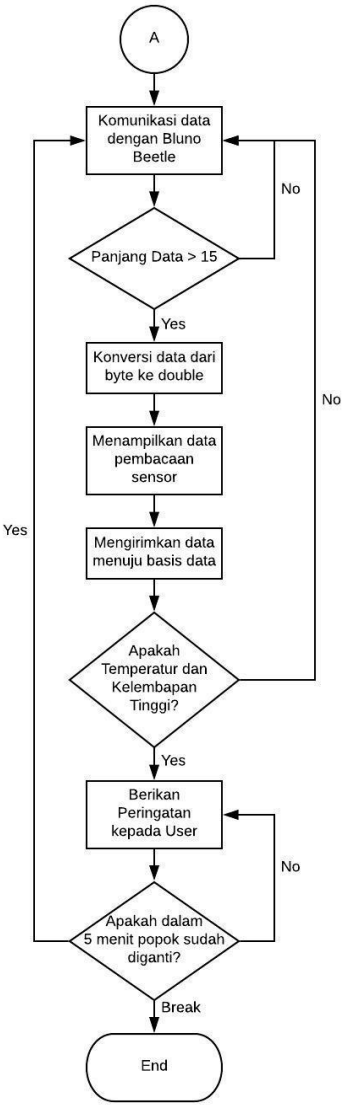


Main menu ini adalah tampilan pertama yang akan dihadapkan oleh pengguna ketika pengguna pertama kali membuka aplikasi "Smart Diaper". Pada tampilan ini, pengguna akan dihadapkan dengan dua pilihan bergantung pada mode yang ingin digunakan pada aplikasi. Dimana ada dua mode yang dapat digunakan pada aplikasi ini, yaitu mode Bluetooth apabila pengguna berada dekat dengan sensor pintar dan juga mode Internet apabila pengguna berada jauh dengan sensor pintar. Dapat dilihat pada flowchart di atas, ketika pengguna membuka Bluetooth mode pada aplikasi maka program akan diteruskan menuju program mode Bluetooth yang akan dijelaskan pada poin selanjutnya, dan apabila pengguna membuka Internet mode pada aplikasi maka program akan diteruskan menuju program mode Internet yang akan dijelaskan pada poin terakhir dari lampiran ini.

Program:

```
package com.dfrobot.angelo.blunobasicdemo;

import android.content.Intent;
import android.os.Bundle;
import android.support.v7.app.AppCompatActivity;
import android.view.View;

public class MainMenu extends AppCompatActivity {

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main_menu);
    }

    public void BluetoothMode(View view) {
        Intent intent = new Intent(this, MainActivity.class);
        startActivity(intent);
    }

    public void InternetMode(View view) {
        Intent intent = new Intent(this, SecondActivity.class);
        startActivity(intent);
    }
}
```

Bluetooth Mode (*Transceiver*)



Apabila pada main menu pengguna memilih untuk menggunakan mode Bluetooth, maka aplikasi akan menjalankan program MainActivity. Pada mode ini, aplikasi akan menerima data hasil pembacaan sensor melalui Bluetooth Low Energy. Setelah data diterima, akan dilakukan pengecekan apakah panjang data yang diterima lebih besar dari 15 atau tidak, apabila panjang data yang diterima 15 atau ke bawah maka program tidak akan berlanjut dan akan terus menerima data dari Bluno Beetle, akan tetapi apabila panjang data yang diterima adalah 16, maka akan dilanjutkan ke tahap selanjutnya. Pada tahap ini, akan dilakukan pengubahan jenis data dari byte menjadi double. Setelah data diubah, maka data tersebut akan ditampilkan pada aplikasi. Setelah data ditampilkan pada aplikasi, data tersebut akan dikirimkan menuju basis data. Setelah itu akan dilakukan pengecekan apakah tingkat kelembapan sudah melewati 60% dan temperatur sudah melewati 33^0C, apabila belum maka proses akan kembali ke awal, akan tetapi apabila sudah melewati batas tersebut maka aplikasi akan memberikan notifikasi dan membunyikan alarm sebagai bentuk pengingat. Setelah muncul notifikasi pertama, akan dilakukan pengecekan kembali apakah dalam 5 menit selanjutnya popok sudah diganti atau belum, apabila popok sudah diganti maka proses akan kembali ke awal, akan tetapi apabila dalam 5 menit popok masih belum diganti maka akan diberikan notifikasi kedua dan seterusnya setiap 5 menit sekali selama popok belum diganti.

Program:

```
package com.dfrobot.angelo.blunobasicdemo;

import android.Manifest;
import android.app.AlertDialog;
import android.content.DialogInterface;
import android.content.Intent;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.os.Handler;
import android.view.View;
import android.view.View.OnClickListener;
import android.widget.Button;
import android.widget.TextView;
import android.widget.Toast;
```

```
import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.FirebaseFirestore;

import java.util.HashMap;
import java.util.Map;

import pub.devrel.easypermissions.AfterPermissionGranted;
import pub.devrel.easypermissions.EasyPermissions;

public class MainActivity    extends BlunoLibrary {
       private Button buttonScan;
       private TextView btstatus;
       private TextView tempstatus;
       private TextView humidstatus;
       private TextView temp;
       private TextView humid;
       int state = 0;
       public AlertDialog alert;
       public AlertDialog reminder;
       MediaPlayer Alarm;
       MediaPlayer Buzzer;
       private final int REQUEST_LOCATION_PERMISSION = 1;
       private DocumentReference mDocRef =
FirebaseFirestore.getInstance().document("Smart Diaper Users/Budiman
Agung Wibowo");
       long startTime = 0;

       Handler timerHandler = new Handler();
       Runnable timerRunnable = new Runnable() {
              @Override
              public void run() {
                     long millis = System.currentTimeMillis() - startTime;
                     int seconds = (int) (millis / 1000);
                     int minutes = seconds / 60;
                     seconds = seconds % 60;

                     timerHandler.postDelayed(this, 500);
                     if (minutes >= 5) {
                            Reminder();
```

```
            }
        }
    };


    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        requestLocationPermission();
        onCreateProcess();
                        //onCreate Process by BlunoLibrary

        serialBegin(115200);
                    //set the Uart Baudrate on BLE chip to 115200

        btstatus = (TextView) findViewById(R.id.btStatus);
        tempstatus = (TextView) findViewById(R.id.tempstatus);
        humidstatus = (TextView) findViewById(R.id.humidstatus);
        temp = (TextView) findViewById(R.id.temp);
        humid = (TextView) findViewById(R.id.humid);
        buttonScan = (Button) findViewById(R.id.buttonScan);
        Alarm = MediaPlayer.create(MainActivity.this,
R.raw.alarm_beep);
        Buzzer = MediaPlayer.create(MainActivity.this,
R.raw.alarm_buzzer);


        buttonScan.setOnClickListener(new OnClickListener() {

            @Override
            public void onClick(View v) {
                // TODO Auto-generated method stub

                buttonScanOnClickProcess();
                        //Alert Dialog for selecting the BLE
device
            }
        });
```

```
    }

    @Override
    public void onRequestPermissionsResult(int requestCode, String[]
permissions, int[] grantResults) {
        super.onRequestPermissionsResult(requestCode, permissions,
grantResults);

        // Forward results to EasyPermissions
        EasyPermissions.onRequestPermissionsResult(requestCode,
permissions, grantResults, this);
    }

    @AfterPermissionGranted(REQUEST_LOCATION_PERMISSIO
N)
    public void requestLocationPermission() {
        String[] perms =
{Manifest.permission.ACCESS_FINE_LOCATION};
        if(EasyPermissions.hasPermissions(this, perms)) {
            Toast.makeText(this, "Permission already granted",
Toast.LENGTH_SHORT).show();
        }
        else {
            EasyPermissions.requestPermissions(this, "Please grant
the location permission", REQUEST_LOCATION_PERMISSION,
perms);
        }
    }


    protected void onResume(){
        super.onResume();
        System.out.println("BlUNOActivity onResume");
        onResumeProcess();
                        //onResume Process by BlunoLibrary
    }
```

```
    @Override
    protected void onActivityResult(int requestCode, int resultCode,
Intent data) {
        onActivityResultProcess(requestCode, resultCode, data);
                //onActivityResult Process by BlunoLibrary
        super.onActivityResult(requestCode, resultCode, data);
    }

    @Override
    protected void onPause() {
        super.onPause();
        onPauseProcess();
                        //onPause Process by BlunoLibrary
    }

    protected void onStop() {
        super.onStop();
        onStopProcess();
                        //onStop Process by BlunoLibrary
    }

    @Override
    protected void onDestroy() {
        super.onDestroy();
        onDestroyProcess();
                        //onDestroy Process by BlunoLibrary
    }

    @Override
    public void onConectionStateChange(connectionStateEnum
theConnectionState) {//Once connection state changes, this function will
be called
        switch (theConnectionState) {
                        //Four connection state
        case isConnected:
            buttonScan.setText("Disconnect");
            btstatus.setText(R.string.kalautersambung);
            break;
        case isConnecting:
```

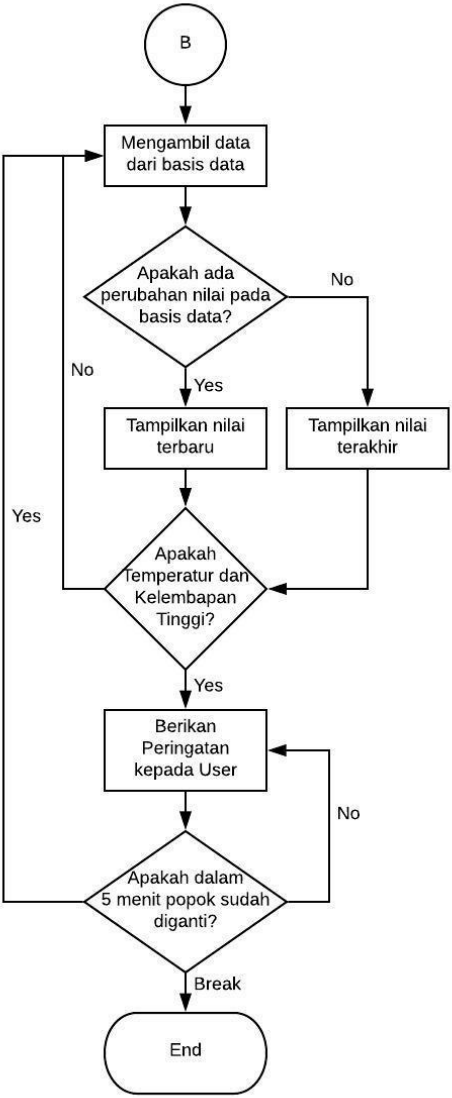
```
                buttonScan.setText("Connecting");
                break;
            case isToScan:
                buttonScan.setText("Scan");
                tempstatus.setText(R.string.tempoff);
                humidstatus.setText(R.string.humidoff);
                btstatus.setText(R.string.kalaugatersambung);
                temp.setText("00.0c");
                humid.setText("00%");
                break;
            case isScanning:
                buttonScan.setText("Scanning");
                break;
            case isDisconnecting:
                buttonScan.setText("isDisconnecting");
                break;
            default:
                break;
            }
        }

        @Override
        public void onSerialReceived(String theString) {
    //Once connection data received, this function will be called
            // TODO Auto-generated method stub
            char[] dataarray;
            double Temperature = 0;
            double Humidity = 0;

            if (theString.length() > 15) {
                tempstatus.setText(R.string.tempon);
                humidstatus.setText(R.string.humidon);
                dataarray = theString.toCharArray();
                for (int i = dataarray.length - 9; i >= 0; i--) {
                    if (dataarray[i] == '1') {
                        Temperature = Temperature + Math.pow(2, 7
    - i);

                    }
```

```
                }
                Temperature = (Temperature / 10) + 20;
                temp.setText(String.format("%.1fc", Temperature));
                for (int i = dataarray.length - 1; i > 7; i--) {
                        if (dataarray[i] == '1') {
                                Humidity = Humidity + Math.pow(2, 15 - i);

                        }
                }

                humid.setText(String.valueOf((int) Humidity + "%"));

                if (Humidity > 60 && Temperature > 30) {

                        if(state == 0) {
                        ChangeDiaper();
                        }
                } else {
                        state = 0;
                        startTime = System.currentTimeMillis();
                        timerHandler.removeCallbacks(timerRunnable);
                }
        } else {
                tempstatus.setText(R.string.tempoff);
                humidstatus.setText(R.string.humidoff);
        }
        Map<String, Double> KirimData = new HashMap<String,
Double>();
        KirimData.put("Temperature", Temperature);
        KirimData.put("Relative Humidity", Humidity);
        mDocRef.set(KirimData);
    }

    public void ChangeDiaper(){

                        if(alert != null && alert.isShowing() ) return;
                        Alarm.start();
                        Alarm.setLooping(true);
                        state = 1;
```

```
                        AlertDialog.Builder b_builder = new
AlertDialog.Builder(MainActivity.this);
                    b_builder.setTitle("Time to change diaper!")
                            .setMessage(R.string.gantipopok)
                            .setCancelable(false)
                            .setPositiveButton("Close", new
DialogInterface.OnClickListener() {
                                @Override
                                public void
onClick(DialogInterface dialogInterface, int i) {
                                    dialogInterface.cancel();
                                    state = 0;
                                    Alarm.stop();
                                    Alarm.prepareAsync();
                                      startTime =
System.currentTimeMillis();

    timerHandler.postDelayed(timerRunnable, 0);
                                }
                            });
                    alert = b_builder.create();
        if (!alert.isShowing()) {
            alert.show();
        }

    }

    public void Reminder(){

        if(reminder != null && reminder.isShowing() ) return;
        Buzzer.start();
        Buzzer.setLooping(true);
        AlertDialog.Builder b_builder = new
AlertDialog.Builder(MainActivity.this);
        b_builder.setTitle("Please change the diaper!")
                .setMessage("It has been 5 minutes since the last
time you received a reminder, please change the diaper")
                .setCancelable(false)
                .setPositiveButton("Close", new
```

```
DialogInterface.OnClickListener() {
                                @Override
                                public void onClick(DialogInterface
dialogInterface, int i) {
                                        dialogInterface.cancel();
                                        Buzzer.stop();
                                        Buzzer.prepareAsync();
                                        startTime = System.currentTimeMillis();
                                }
                        });
                reminder = b_builder.create();
                if (!reminder.isShowing()) {
                        reminder.show();
                }
        }
}
```

Internet Mode (*Receiver*)



Apabila pada main menu pengguna memilih untuk menggunakan mode Internet, maka aplikasi akan menjalankan program SecondActivity. Pada mode ini, aplikasi akan mengambil data hasil pembacaan sensor dari basis data. Pertama, akan dilakukan pengecekan apakah ada perubahan nilai pada basis data atau tidak, apabila tidak ada perubahan maka nilai yang akan ditampilkan adalah nilai terakhir, akan tetapi apabila ada perubahan data maka yang akan ditampilkan adalah nilai terbaru. Setelah data ditampilkan pada aplikasi, akan dilakukan pengecekan apakah tingkat kelembapan sudah melewati 60% dan temperatur sudah melewati 33^0C, apabila belum maka proses akan kembali ke awal, akan tetapi apabila sudah melewati batas tersebut maka aplikasi akan memberikan notifikasi dan membunyikan alarm sebagai bentuk pengingat. Setelah muncul notifikasi pertama, akan dilakukan pengecekan kembali apakah dalam 5 menit selanjutnya popok sudah diganti atau belum, apabila popok sudah diganti maka proses akan kembali ke awal, akan tetapi apabila dalam 5 menit popok masih belum diganti maka akan diberikan notifikasi kedua dan seterusnya setiap 5 menit sekali selama popok belum diganti.

Program:

```
package com.dfrobot.angelo.blunobasicdemo;

import android.app.AlertDialog;
import android.content.DialogInterface;
import android.media.MediaPlayer;
import android.os.Bundle;
import android.os.Handler;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.Button;
import android.widget.TextView;

import com.google.firebase.firestore.DocumentReference;
import com.google.firebase.firestore.DocumentSnapshot;
import com.google.firebase.firestore.EventListener;
import com.google.firebase.firestore.FirebaseFirestore;
import com.google.firebase.firestore.FirebaseFirestoreException;

import javax.annotation.Nullable;
```

```
public class SecondActivity extends AppCompatActivity {
    private TextView humidint;
    private TextView tempint;
    private TextView intstatus;
    TextView timer;
    private Button Tombol;
    int state = 0;
    boolean kondisi = false;
    public AlertDialog alert;
    public AlertDialog reminder;
    MediaPlayer Alarm;
    MediaPlayer Buzzer;
    private final int REQUEST_LOCATION_PERMISSION = 1;
    private DocumentReference mDocRef =
FirebaseFirestore.getInstance().document("Smart Diaper Users/Budiman
Agung Wibowo");
    long startTime = 0;

    Handler timerHandler = new Handler();
    Runnable timerRunnable = new Runnable() {
        @Override
        public void run() {
            long millis = System.currentTimeMillis() - startTime;
            int seconds = (int) (millis / 1000);
            int minutes = seconds / 60;
            seconds = seconds % 60;

            timerHandler.postDelayed(this, 500);
            if (minutes >= 5) {
                Reminder();
            }
        }
    };

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_second);
```

```
        humidint = (TextView) findViewById(R.id.humidint);
        tempint = (TextView) findViewById(R.id.tempint);
        intstatus = (TextView) findViewById(R.id.intstatus);
        Tombol = (Button) findViewById(R.id.Tombol);
        Alarm = MediaPlayer.create(SecondActivity.this,
R.raw.alarm_beep);
        Buzzer = MediaPlayer.create(SecondActivity.this,
R.raw.alarm_buzzer);

    }

    @Override
    protected void onStart() {
        super.onStart();
    }

    public void Gantimode (View view) {
        if (kondisi == false) {
            kondisi = true;
            state = 0;
            Tombol.setText("Stop");
            mDocRef.addSnapshotListener(this, new
EventListener<DocumentSnapshot>() {
                @Override
                public void onEvent(@Nullable DocumentSnapshot
documentSnapshot, @Nullable FirebaseFirestoreException e) {
                    if (documentSnapshot.exists()) {
                        intstatus.setText("Connected to
Databank");
                        long Humidityint =
documentSnapshot.getLong("Relative Humidity");
                        Double Temperatureint =
documentSnapshot.getDouble("Temperature");
                        humidint.setText(Humidityint + "%");
                        tempint.setText(Temperatureint + "c");
                        if (Humidityint > 60 && Temperatureint >
30) {

                            if(state == 0) {
```

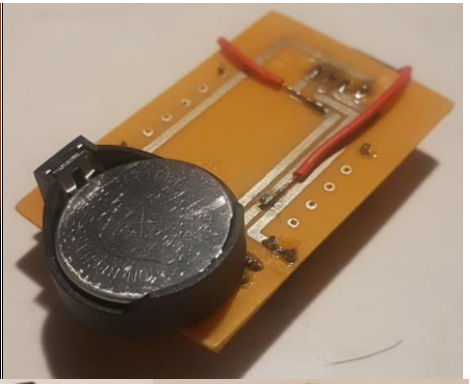
```
                                        ChangeDiaper();
                                    }
                                } else {
                                    state = 0;
                                    startTime =
System.currentTimeMillis();

timerHandler.removeCallbacks(timerRunnable);
                                }
                            } else {
                                intstatus.setText("Not Connected to
Databank");
                                humidint.setText("00%");
                                tempint.setText("00.0c");

                            }
                        }
                    });
            } else {
                kondisi = false;
                state = 1;
                Tombol.setText("Start");
                mDocRef.addSnapshotListener(this, new
EventListener<DocumentSnapshot>() {
                    @Override
                    public void onEvent(@Nullable DocumentSnapshot
documentSnapshot, @Nullable FirebaseFirestoreException e) {
                        if (documentSnapshot.exists()) {
                            intstatus.setText("Not Connected to
Databank");
                            humidint.setText("00%");
                            tempint.setText("00.0c");
                        }
                    }
                });
            }
    }

    public void ChangeDiaper(){
```

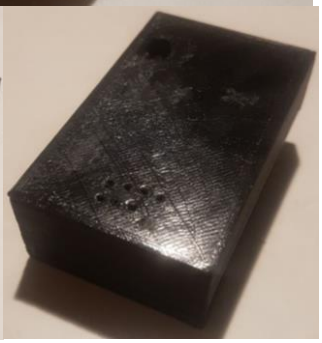
```
            if(alert != null && alert.isShowing() ) return;
            Alarm.start();
            Alarm.setLooping(true);
            state = 1;
            AlertDialog.Builder b_builder = new
AlertDialog.Builder(SecondActivity.this);
            b_builder.setTitle("Time to change diaper!")
                        .setMessage(R.string.gantipopok)
                        .setCancelable(false)
                        .setPositiveButton("Close", new
DialogInterface.OnClickListener() {
                              @Override
                              public void onClick(DialogInterface
dialogInterface, int i) {
                                    dialogInterface.cancel();
                                    Alarm.stop();
                                    Alarm.prepareAsync();
                                    startTime = System.currentTimeMillis();
                                    timerHandler.postDelayed(timerRunnable,
0);
                              }
                        });
            alert = b_builder.create();
            if (!alert.isShowing()) {
                  alert.show();
            }
      }

      public void Reminder(){

            if(reminder != null && reminder.isShowing() ) return;
            Buzzer.start();
            Buzzer.setLooping(true);
            AlertDialog.Builder b_builder = new
AlertDialog.Builder(SecondActivity.this);
            b_builder.setTitle("Please change the diaper!")
                        .setMessage("It has been 5 minutes since the last
time you received a reminder, please change the diaper")
```

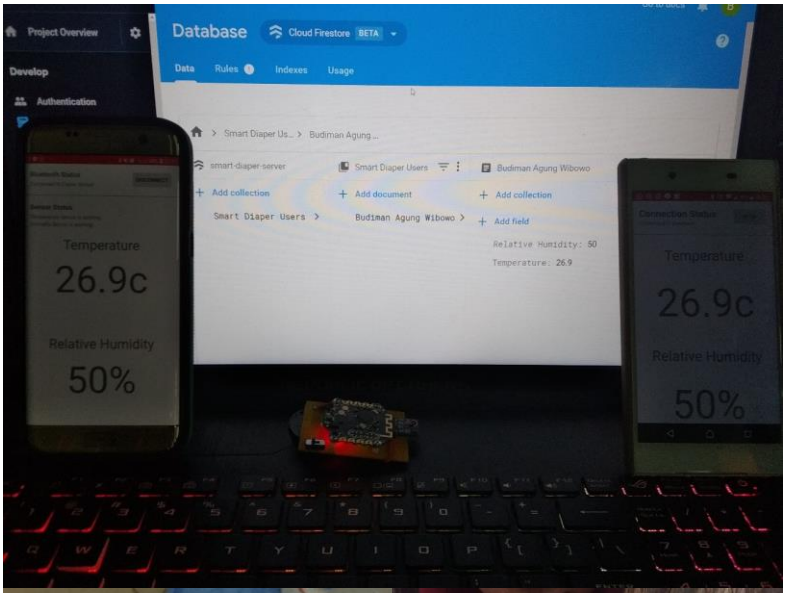
```
                    .setCancelable(false)
                    .setPositiveButton("Close", new
DialogInterface.OnClickListener() {
                        @Override
                        public void onClick(DialogInterface
dialogInterface, int i) {
                            dialogInterface.cancel();
                            Buzzer.stop();
                            Buzzer.prepareAsync();
                            startTime = System.currentTimeMillis();
                        }
                    });
            reminder = b_builder.create();
            if (!reminder.isShowing()) {
                reminder.show();
            }
        }
}
```

.....Halaman ini sengaja dikosongkan.....

LAMPIRAN C
Dokumentasi Kegiatan Tugas Akhir





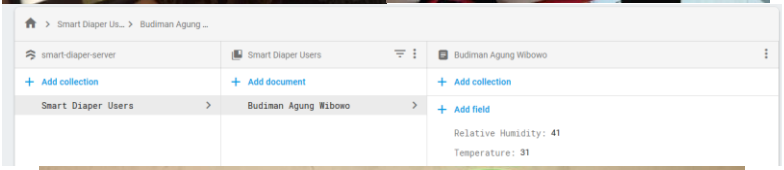

Project Overview
Develop
Authentication
Database
Cloud Firestore BETA
Data
Rules
Indexes
Usage
smart-diaper-server
Smart Diaper Users
Budiman Agung Wibowo
Add collection
Add document
Add collection
Add field
Relative Humidity: 50
Temperature: 26.9
Temperature
26.9c
Relative Humidity
50%
Temperature
26.9c
Relative Humidity
50%

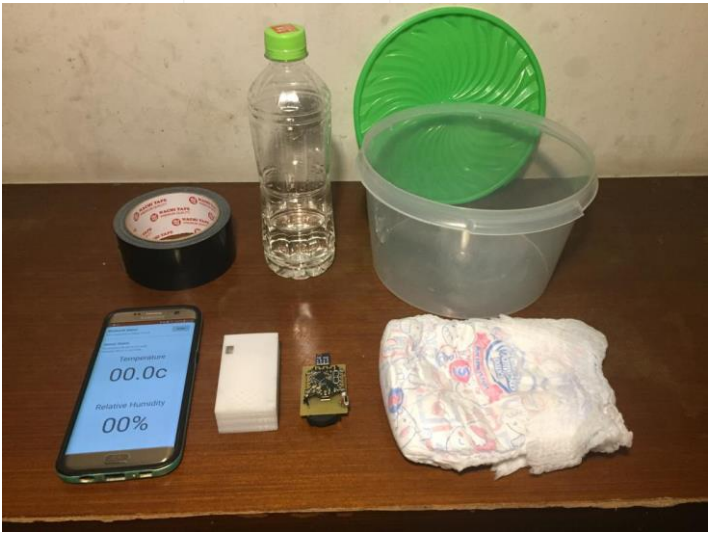

SAMSUNG
Bluetooth Status
DISCONNECT
Sensor Status
Temperature
32.1c
Relative Humidity
67%


Temperature
33.5c
Relative Humidity
65%


Temperature
33.6c
Relative Humidity
62%


3.93
LCR Multimeter


Smart Diaper Us...
Budiman Agung ...
smart-diaper-server
Add collection
Smart Diaper Users
Smart Diaper Users
Add document
Budiman Agung Wibowo
Budiman Agung Wibowo
Add collection
Add field
Relative Humidity: 41
Temperature: 31


Temperature
00.0c
Relative Humidity
00%

.....Halaman ini sengaja dikosongkan.....

BIODATA PENULIS

Budiman Agung Wibowo lahir di Jakarta pada tanggal 29 November 1996 merupakan anak ketiga dari tiga bersaudara. Penulis menyelesaikan pendidikan dasar di SD Yasporbi II Pancoran, dilanjutkan pendidikan tingkat menengah di SMP Negeri 19 Jakarta dan sekolah tingkat atas di SMA Negeri 14 Jakarta. Penulis memulai kehidupan perkuliahan pada tahun 2014 di Departemen Teknik Elektro, Fakultas Teknologi Elektro, Institut Teknologi Sepuluh Nopember Surabaya. Selama masa perkuliahan, penulis aktif dalam berbagai kegiatan kepanitiaan, organisasi, dan marching band. Selain itu, penulis juga menjadi asisten praktikum di bidang studi elektronika.

Email	: budiman_agungwibowo@yahoo.com
Hp/WA	: 085718267230
Facebook	: Budiman Agung Wibowo
Line	: dimdimie