

PENGEMBANGAN SMART CONVEYOR BELT DENGAN ARDUINO

(Penataan Barang dengan *Robot Arm*)

DASAR-DASAR PYTHON

Rolly M. Awangga
Informatics Research Center



Kreatif Industri Nusantara

Penulis:

Rolly Maulana Awangga

ISBN : 978-602-53897-0-2

Editor:

M. Yusril Helmi Setyawan

Penyunting:

Syafrial Fachrie Pane

Khaera Tunnisa

Diana Asri Wijayanti

Desain sampul dan Tata letak:

Deza Martha Akbar

Penerbit:

Kreatif Industri Nusantara

Redaksi:

Jl. Ligar Nyawang No. 2

Bandung 40191

Tel. 022 2045-8529

Email : awangga@kreatif.co.id

Distributor:

Informatics Research Center

Jl. Sariasih No. 54

Bandung 40151

Email : irc@poltekpos.ac.id

Cetakan Pertama, 2019

Hak cipta dilindungi undang-undang
Dilarang memperbanyak karya tulis ini dalam bentuk dan dengan cara
apapun tanpa ijin tertulis dari penerbit

*'Jika Kamu tidak dapat
menahan lelahnya
belajar, Maka kamu harus
sanggup menahan
perihnya Kebodohan.'
Imam Syafi'i*

CONTRIBUTORS

ROLLY MAULANA AWANGGA, Informatics Research Center., Politeknik Pos Indonesia, Bandung, Indonesia

CONTENTS IN BRIEF

1	Python	1
2	<i>Python</i>	3
3	Judul Bagian Kedua	23
4	Fungsi dan Kelas	29

DAFTAR ISI

Daftar Gambar	xi
Daftar Tabel	xiii
Foreword	xvii
Kata Pengantar	xix
Acknowledgments	xxi
Acronyms	xxiii
Glossary	xxv
List of Symbols	xxvii
Introduction	xxix
<i>Rolly Maulana Awangga, S.T., M.T.</i>	
1 Python	1
2 Python	3

2.1	Sejarah <i>Python</i>	3
2.2	Perbedaan <i>Python 2.x</i> dan <i>Python 3.x</i>	4
		ix
x	DAFTAR ISI	
2.3	Instalasi Python	11
2.3.1	Windows (Windows 10)	11
2.4	Instalasi Pip	16
2.4.1	Windows (Windows 10)	16
2.4.2	Linux (Ubuntu 19.04)	17
2.5	Setting Environment	18
2.5.1	Windows (Windows 10)	18
2.5.2	Linux (Ubuntu 19.04)	20
2.6	Command Line Interface/Interpreter	21
2.6.1	Windows (Windows 10)	21
2.6.2	Linux (Ubuntu 19.04)	22
3	Judul Bagian Kedua	23
3.1	Variabel	23
3.2	Input dan Output	24
3.3	Operasi Aritmatika	24
3.4	Perulangan	24
3.4.1	For	25
3.4.2	While	25
3.5	Kondisi	25
3.6	Error	27
3.7	Try Except	28
4	Fungsi dan Kelas	29
4.1	Teori	29
4.1.1	Fungsi	29
4.2	Package	30
4.3	Class, Object, Atribut, and Method	30
4.4	Pemanggilan Class	31
4.5	Pemakaian Package Fungsi Apabila File Didalam Folder	31
4.6	Pemakaian Package Kelas Apabila File didalam Folder	31
	Daftar Pustaka	33

DAFTAR GAMBAR

2.1	Gambar hasil print	4
2.2	Gambar perintah print	5
2.3	Gambar hasil pembagian	6
2.4	Gambar perintah pembagian	6
2.5	Gambar hasil error	7
2.6	Gambar perintah error	7
2.7	Gambar hasil looping	8
2.8	Gambar perintah looping	8
2.9	Gambar hasil unicode (bytes)	9
2.10	Gambar perintah unicode (bytes)	9
2.11	Gambar hasil unicode	10

2.12	Gambar perintah unicode	10
2.13	Run Setup Anaconda	11
		xi

2.14	Setup Loading	12
2.15	Welcome to Anaconda Setup	12
2.16	<i>License Agreement</i>	12
2.17	<i>Just Me(recomended)</i>	13
2.18	<i>Pilih lokasi</i>	13
2.19	<i>Centang Anaconda to my PATH</i>	14
2.20	<i>Installation Complete</i>	14
2.21	<i>Installation Complete</i>	15
2.22	<i>Anaconda+JetBrains</i>	15
2.23	<i>Thanks for install Anaconda</i>	16
2.24	<i>Install pip</i>	16
2.25	<i>Install pip Selesai</i>	17
2.26	<i>Melihat Versi pip</i>	17
2.27	Gambar instal pip	18
2.28	<i>Properties</i>	18
2.29	<i>Advanced system settings</i>	19
2.30	<i>Environment Variables</i>	19
2.31	<i>Path</i>	20
2.32	<i>Edit Environment Variable</i>	20
2.33	Gambar setpath	21
2.34	<i>CLI in Command Prompt</i>	22
2.35	Gambar running script dengan CLI	22

DAFTAR TABEL

Listings

FOREWORD

Sepatah kata dari Kaprodi, Kabag Kemahasiswaan dan Mahasiswa

KATA PENGANTAR

xix

ACKNOWLEDGMENTS

Terima kasih atas semua masukan dari para mahasiswa agar bisa membuat buku ini lebih baik dan lebih mudah dimengerti.

Terima kasih ini juga ditujukan khusus untuk team IRC yang telah fokus untuk belajar dan memahami bagaimana buku ini mendampingi proses Intership.

R. M. A.

ACRONYMS

ACGIH	American Conference of Governmental Industrial Hygienists
AEC	Atomic Energy Commission
OSHA	Occupational Health and Safety Commission
SAMA	Scientific Apparatus Makers Association

Cerdas Menguasai Git, First Edition.

By Rolly M. Awangga Copyright © 2019 Kreatif Industri Nusantara.

xxiii

GLOSSARY

git	Merupakan manajemen sumber kode yang dibuat oleh linus torvald.
bash	Merupakan bahasa sistem operasi berbasiskan *NIX.
linux	Sistem operasi berbasis sumber kode terbuka yang dibuat oleh Linus Torvald

SYMBOLS

- A Amplitude
& Propositional logic symbol
 a Filter Coefficient

 B Number of Beats

PENGANTAR

xxix

BAGIAN 1

PENDAHULUAN-PENGANTAR RFID

1.1 Definisi

Radio Frequency Identification atau disingkat (RFID) (Bahasa Indonesia: Identifikasi frekuensi radio adalah sistem penyimpanan data dari jarak jauh menggunakan nirkabel yang jangkauannya antara 50 KHz sampai 2.5 GHz, jangkauannya ini terdiri dari *Radio Frequency Identification (RFID) tag*, *Radio Frequency Identification (RFID reader)*, dan *the host line of business system* [1], Sebuah *tag Radio Frequency Identification (RFID)* atau transponder, terdiri atas sebuah mikro (*microchip*) dan sebuah antena. *Chip mikro* itu sendiri dapat berukuran sekecil butiran pasir, seukuran 0,4 mm . *Chip* tersebut menyimpan nomor seri yang unik atau informasi lainnya tergantung kepada tipe memorinya. Tipe memori itu sendiri dapat *read-only*, *read-write*, atau *write-once read-many*. Antena yang terpasang pada *chip* mikro mengirimkan informasi dari *chip* ke *reader*[2].

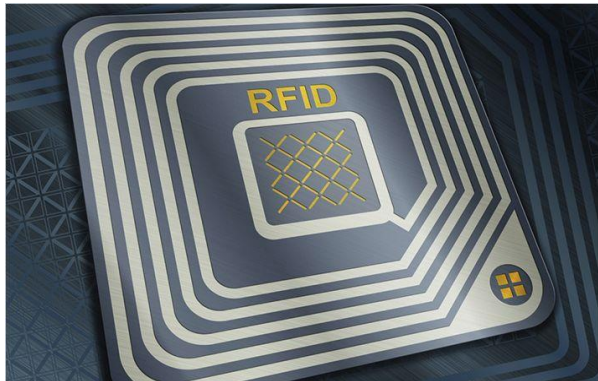
Pada dasarnya *Radio Frequency Identification (RFID)* berperan sama dengan dengan *Barcode*,

magnetik, *smart card*, punchcard, kode/nomor rekening pada buku *check*, label, dll. cara kerja *Radio Frequency Identification* (RFID) yaitu:

1. Sebuah alat pembaca *Radio Frequency Identification* (RFID) akan selalu memancarkan *signal* / freq tertentu secara terus-menerus sampai terdapat sebuah *chip Radio Frequency Identification* (RFID) menerima signal tersebut pada jarak jangkauan tertentu tergantung dengan antenna yang terpasang.
2. Sebuah *chip Radio Frequency Identification* (RFID) melintasi area dari pembaca *Radio Frequency Identification* (RFID) tersebut, dimana *chip* tersebut akan secara otomatis aktif jika freq yang dipancarkan sesuai dengan freq yang di set didalam *chip Radio Frequency Identification* (RFID) tersebut yaitu membalas dengan cara mengirimkan data yang terdapat didalamnya.
3. Alat pembaca yang mengirimkan freq tersebut akan menerima data yang dikirimkan oleh *chip Radio Frequency Identification* (RFID), lalu

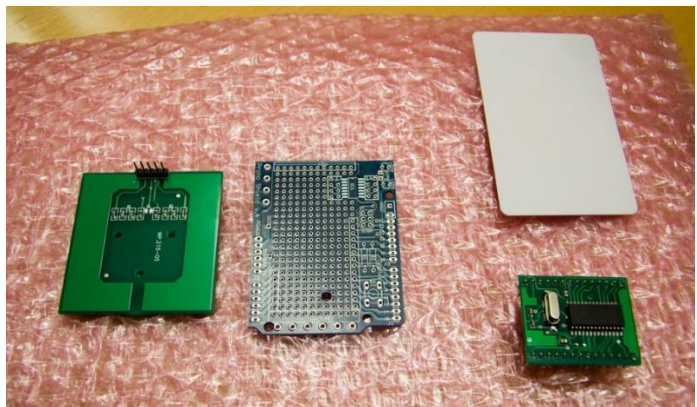
melanjutkan data tersebut ke komputer/microkontroller untuk diolah.

4. Jika data yang terkirim tersebut sesuai dengan yang diinginkan, maka akan dilakukan sesuai sesuai dengan keinginan[3].



Gambar 1.1

Radio Frequency Identification (RFID)



Gambar 1.2

RFID Reader dan Arduino Prototyping Shield

Kelebihan *Radio Frequency Identification (RFID)* dibandingkan alat-alat yang sejenis yaitu :

- Data yang dapat ditampung lebih banyak daripada alat bantu lainnya (kurang lebih 2000 byte)
- Ukuran sangat kecil (untuk jenis pasif *Radio Frequency Identification (RFID)*) sehingga mudah ditanamkan dimana-mana
- Bentuk dan design yang flexibel sehingga sangat mudah untuk dipakai diberbagai tempat dan kegunaan karena *chip Radio Frequency Identification (RFID)* dapat dibuat dari tinta khusus.
- Pembacaan informasi sangat mudah, karena bentuk dan bidang tidak mempengaruhi pembacaan, seperti sering terjadi pada *barcode*, magnetik dll.
- Jarak pembacaan yang flexibel bergantung pada antena dan jenis *chip Radio Frequency Identification (RFID)* yang digunakan.

Seperti contoh *autopayment* pada jalan tol, penghitungan stok pada ban berjalan, *access gate*.

- Kecepatan dalam pembacaan data.

Kelemahan teknologi *Radio Frequency Identification (RFID)* :

- Akan terjadi kekacauan informasi jika terdapat lebih daripada 1 *chip Radio Frequency Identification (RFID)* melalui 1 alat pembaca secara bersamaan, karena akan terjadinya tabrakan informasi yang diterima oleh pembaca (kendala ini dapat terselesaikan oleh kemampuan akan kecepatan penerimaan data sehingga *chip Radio Frequency Identification (RFID)* yang masuk belakangan akan dianggap sebagai data yang berikutnya).
- Jika terdapat *freq overlap* (dua freq dari pembaca berada dalam satu area) dapat memberikan informasi data yang salah pada komputer/pengolah data sehingga tingkat akurasi akan berkurang (permasalahan ini

dipecahkan dengan cara pengimplementasian alat deteksi tabrakan frekuensi atau menata peletakan area pembacaan sehingga dapat menghindari tabrakan).

- Gangguan akan terjadi jika terdapat frekuensi lain yang dipancarkan oleh peralatan lainnya yang bukan diperuntukkan untuk *Radio Frequency Identification (RFID)*, sehingga *chip* akan merespon frekuensi tersebut (frekuensi Wifi, *handphone*, radio pemancar, dll).
- Privasi seseorang akan secara otomatis menjadi berkurang, karena siapa saja dapat membaca informasi dari diri seseorang dari jarak jauh selama orang tersebut memiliki alat pembaca, sebagai contoh seseorang dapat membaca jumlah uang yang dimiliki orang lain didalam dompetnya[3].

Jika diimplementasikan di toko retail di Indonesia bagus sekali karena itu dapat menghemat biaya untuk membayar upah staf pekerja, bayangkan saja, tinggal barang dimasukkan di troli pas *customer* membawa troli di kasir, langsung juga

tidak menjadi panjang. Dan bisa diimplementasikan tetapi harus tersedia *cash* yang besar juga, karena kan harus *training* pegawai, tertera jumlah yang harus dibayar, menghemat waktu, antrian dan ini teknologi baru, jadi jika ada kerusakan atau *maintenance*, biaya yang harus dikeluarkan tidaklah murah. Jadi kalau menurut saya siiihh,, seperti retail yang besar, contoh di indonesia yaitu : *hypermart*, *carrefour*, *giant*, *gramedia* (karena toko buku *gramedia* toko buku terbesar di indonesia dan mempunyai cabang yang cukup banyak, serta untuk urusan TI, mereka tidak pelit).

Biasanya rentang pembacaan diindikasikan dengan besarnya antena. Antena yang lebih besar mengindikasikan rentang pembacaan yang lebih *Tag* tersebut terpasang atau tertanam dalam objek yang akan diidentifikasi. *Tag* dapat scan dengan *reader* bergerak maupun stasioner menggunakan gelombang radio[3]. Sebuah *reader* menggunakan antenanya sendiri untuk berkomunikasi dengan *tag*. Ketika *reader* memancarkan gelombang radio[4][5], seluruh *tag* yang dirancang pada frekuensi tersebut serta berada pada rentang bacanya akan memberikan respon. Sebuah *reader* juga dapat berkomunikasi dengan *tag* tanpa *line-of-sight* langsung,

tergantung kepada frekuensi radio dan tipe *tag* (aktif, pasif, atau semipasif) yang digunakan. *Reader* dapat memproses banyak item sekaligus. Menurut bentuknya, *reader* dapat berupa *reader* bergerak seperti peralatan genggam, atau stasioner seperti peralatan *point-of-sale* di supermarket[6].

Reader dibedakan berdasarkan kapasitas penyimpanannya, kemampuan pemrosesannya, serta frekuensi yang dapat dibacanya[7] *Radio Frequency Identification (RFID) reader* terdiri dari beberapa jenis antara lain rc522 dan rdm6300, kedua *reader* tersebut memiliki keutungan dan kelemahanya masing-masing, rc522 memiliki frekuesnsi 13,52 MHz memiliki kelemahan hanya mampu membaca *tag* dengan jarak 1 cm sedangkan rdm6300 memiliki frekuensi 125 MHz yang mampu membaca *tag* dengan jarak 5 cm[8][9][10]. Gudang merupakan salah satu bagian penting dari sebuah pabrik atau perusahaan yang berfungsi sebagai tempat penyimpanan, baik barang hasil produksi maupun bahan baku yang akan diproduksi perusahaan tersebut. Jika dilihat dari segi fungsi dapat diketahui bahwa tingkat mobilitas barang dalam gudang sangat tinggi setiap harinya, hampir terdapat ratusan bahkan ribuan barang produksi maupun bahan baku masuk atau

keluar gudang[11], proses pencatatan barang di dalam gudang masih menggunakan metode tradisional artinya pencatatan di dalam gudang dilakukan secara manual.

Proses pencatatan dilakukan dengan menulis spesifikasi terhadap barang yang masuk dan keluar, sedangkan pada pengecekan barang dengan melihat fisik barang dan menghitung jumlah barang satu per satu pada setiap barang yang masuk dan keluar. Metode pencatatan ini berisiko ketika manusia tidak teliti (*human error*) dan membuat gudang penuh karna terdapat produk-produk lama yang tidak digunakan sehingga terjadi penumpukan barang[2]. Masalah yang terjadi pada saat ini adalah barang yang berada di dalam Gudang belum bisa melakukan proses *Outbound*, Dalam aktivitas *Inbound* dan *Outbound Logistics* terdapat lima area dalam aktivitasnya yakni *order processing*, *inventory*, *transportation*, *warehousing*, *materials handling*, *packaging*, serta *facility network design*[15]. Karena belum ada *system* yang di rancang untuk proses *Outbound* barang tersebut. Oleh karna itu penulis merancang *system Outbound* tersebut.

Solusi untuk mengatasi masalah tersebut adalah membangun *system* baru, *system* baru tersebut berfungsi untuk mengatur proses *outbound* barang yang berada di Gudang,

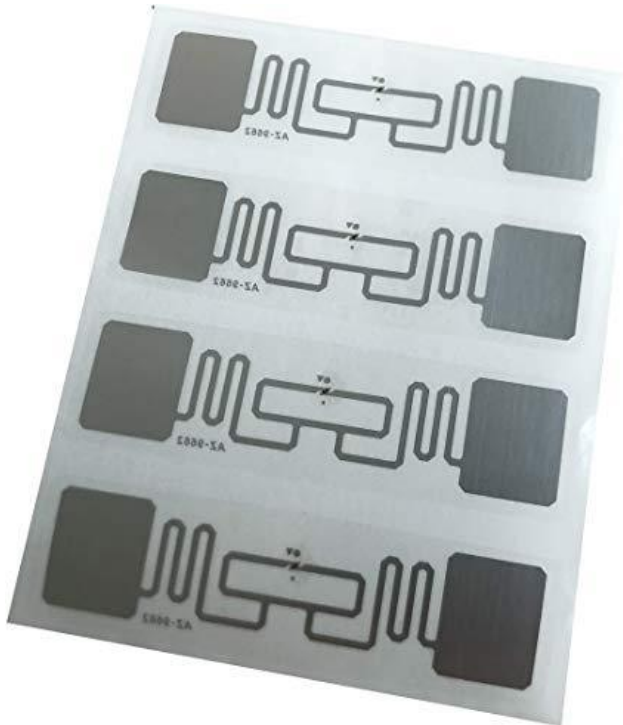
ketika barang tersebut berada di Gudang maka barang tersebut akan di tambahkan dengan data tujuan barang dan status barang akan berubah menjadi “proses” dan ketika barang tersebut di scan *outbound* maka status barang akan berubah menjadi “kirim”.

Metode pengembangan yang digunakan adalah metode pengembangan *prototype*[12] yaitu metode yang menggunakan pendekatan untuk membangun suatu program dengan cepat dan bertahap sehingga segera dapat dievaluasi oleh pemakai, dengan tahapan yang digunakan yaitu pengumpulan kebutuhan dan perbaikan, perancangan cepat, membentuk *prototype*, evaluasi pelanggan terhadap *prototype*, Perbaikan *prototype* dan produk rekayasa[13]. Yang dilakukan pada penelitian ini adalah mengembangkan alat yang sudah ada dengan menambahkan proses *outbound* yang bertujuan untuk mengeluarkan barang dari Gudang, dimana barang tersebut memiliki tujuan pengirimannya masing-masing, pengembangan ini menggunakan metode *prototype*.

1.1 Label Radio Frequency Identification (RFID)

Label *Radio Frequency Identification (RFID)* terdiri atas *mikrochip* silikon dan antena. Label yang pasif tidak

membutuhkan sumber tenaga, sedangkan label yang aktif membutuhkan sumber tenaga untuk dapat berfungsi.



Gambar 1.3

Label *Radio Frequency Identification (RFID)*

Teknologi *Radio Frequency Identification (RFID)* menjadi jawaban atas berbagai kelemahan yang dimiliki teknologi *barcode* yaitu selain karena hanya bisa diidentifikasi dengan cara mendekatkan *barcode* tersebut ke sebuah *reader*, juga karena mempunyai kapasitas

penyimpanan data yang sangat terbatas dan tidak bisa deprogram ulang sehingga menyulitkan untuk menyimpan dan memperbaharui data dalam jumlah besar untuk sebuah item. Salah satu solusi menarik yang kemudian muncul adalah menyimpan data tersebut pada suatu silikon *chip*, teknologi inilah yang dikenal dengan *Radio Frequency Identification (RFID)*. Kontak antara *Radio Frequency Identification (RFID) tag* dengan *reader* tidak dilakukan secara kontak langsung atau mekanik melainkan dengan pengiriman gelombang *electromagnet*. Berbeda dengan *smart card* yang biasa dipakai di kartu telepon atau kartu bank yang juga menggunakan silikon *chip*, kode-kode *Radio Frequency Identification (RFID) tag* bisa dibaca pada jarak yang cukup jauh Suatu sistem *Radio Frequency Identification (RFID)* secara utuh terdiri atas 3 komponen yaitu :

1. *Tag Radio Frequency Identification (RFID)*, dapat berupa stiker, kertas atau plastik dengan beragam ukuran. Didalam setiap tag ini terdapat *chip* yang mampu menyimpan sejumlah informasi tertentu.

2. *Terminal Reader Radio Frequency Identification (RFID)*, terdiri atas *Radio Frequency Identification (RFID) -reader* dan antena yang akan mempengaruhi jarak optimal identifikasi. *Terminal Radio Frequency Identification (RFID)* akan membaca atau mengubah informasi yang tersimpan didalam tag melalui frekuensi radio. *Terminal Radio Frequency Identification (RFID)* terhubung langsung dengan sistem *Host Komputer*.
3. *Host Komputer*, sistem komputer yang mengatur alur informasi dari *item-item* yang terdeteksi dalam lingkup sistem *Radio Frequency Identification (RFID)* dan mengatur komunikasi antara *tag* dan *reader*. *Host* bisa berupa komputer stand-alonemaupun terhubung ke jaringan LAN / Internet untuk komunikasi dengan server.

Label *Radio Frequency Identification (RFID)* atau yang biasa disebut *Radio Frequency Identification (RFID) tag* sendiri, pada dasarnya merupakan suatu *microchip* berantena, yang disertakan pada suatu unit barang. Dengan

piranti ini, perusahaan bisa mengidentifikasi dan melacak keberadaan suatu produk. Seperti halnya *barcode*, yang memiliki *Universal Product Code* atau singkat dengan (UPC), sebuah *tag Radio Frequency Identification (RFID)* memiliki *Electronic Product Code* atau disingkat dengan (EPC) berisi identitas produk tersebut, mulai dari nomor seri, tanggal produksi, lokasi manufaktur, bahkan tanggal kadaluarsa. EPC adalah identifikasi produk generasi baru, mirip dengan UPC atau *barcode*. Seperti halnya *barcode*, EPC terdiri dari angka-angka yang menunjukkan kode produsen, produk, versi dan nomor seri.

Namun perbedaan antara *Radio Frequency Identification (RFID)* dan *Barcode* sangat besar dalam beberapa hal seperti:

Keterangan	<i>Barcode</i>	RFID Pasif
Kondisi Baca	<i>Line Of Sight</i> (LOS)	Non LOS
Posisi Baca	Vertikal/Horizontal dengan posisi tertentu	Bebas, Segala Posisi
Kecepatan Baca	<i>Relative</i> (2,5, detik)	<100 mili detik per item

Jarak Baca Maksimum	+7cm(pendek)	Ada Beberapa kategori Pendek (+- 30 cm) Menengah(+ 3 Mtr) Jauh(+ 10 mter)
Kemampuan	Baca Saja	Baca Dan Tulis
Kapasitas <i>Memory</i>	Kecil	Hingga 64kb/ Lebih
Proses Pembacaan	Per <i>Item</i> (Satu Per Satu)	Multi <i>Item</i> (100 <i>Unit</i>) Per Proses
Kondisi Buruk (Debu, Air)	Merusak Atau Tidak Terbaca	Tidak Terpengaruh
Kemudahan Duplikasi	Mudah	Hampir Mustahil

Tabel 1.1

EPC (*Electronic Product Code*)

Label atau *tag Radio Frequency Identification (RFID)* memiliki banyak macam berdasarkan bahan pembuatannya, ketahanan suhu, kelenturan pada bidang

aset yang akan di tempel, jarak baca, frekwensi dan lain lain, sehingga setiap aset yang akan di tempeli oleh *tag* atau label tersebut harus sesuai dengan bahan dari aset, contoh untuk aset yang terbuat dari logam misal pipa besi dll maka harus digunakan label rfid khusus yang memang didesign untuk pemasangan pada besi. Untuk dapat membaca EPC pada suatu label *Radio Frequency Identification (RFID)* maka di perlukan sebuah *reader Radio Frequency Identification (RFID)* sehingga data pada *chip* label *Radio Frequency Identification (RFID)* dapat tampil pada monitor *Reader* ataupun PC. Setiap EPC memiliki 24 karakter *hexa* sehingga dapat dijadikan kode unik pada setiap perusahaan dengan memberikan arti dari masing masing karakter.

Namun, EPC memiliki digit ekstra untuk mengidentifikasi item yang unik. Ukuran bit EPC yang mencapai 96-bit memungkinkannya secara unik mengidentifikasi lebihdari 268 juta produsen, masing-masing memiliki lebih dari satu juta jenisproduk, sementara sisanya masih mencukupi untuk melabel seluruh produkindividualnya. Informasi EPC inilah yang tersimpan di dalam *chip Radio Frequency Identification (RFID)*.

Tipe *Radio Frequency Identification (RFID) tag*

Radio Frequency Identification (RFID) tag dapat bersifat aktif atau pasif.

➤ *Radio Frequency Identification (RFID) Pasif*

Radio Frequency Identification (RFID) tag yang pasif tidak memiliki *power supply* sendiri. Dengan hanya berbekal induksi listrik yang ada pada antenna yang disebabkan oleh adanya frekuensi radio *scanning* yang masuk, sudah cukup untuk memberi kekuatan yang cukup bagi *Radio Frequency Identification (RFID) tag* untuk mengirimkan respon balik. Sehubungan dengan power dan biaya, maka respon dari suatu *Radio Frequency Identification (RFID)* yang pasif biasanya sederhana, hanya nomor ID saja. Dengan tidak adanya power supply pada *Radio Frequency Identification (RFID) tag* yang pasif maka akan menyebabkan semakin kecilnya ukuran dari *Radio Frequency Identification (RFID) tag* yang mungkin dibuat. Beberapa *Radio Frequency Identification (RFID)* komersial yang saat ini sudah beredar di pasaran ada yang bisa diletakkan di bawah kulit. Pada tahun 2005 tercatat bahwa *Radio Frequency*

Identification (RFID) tag terkecil berukuran 0.4 mm x 0.4 mm dan lebih tipis daripada selembar kertas. Dengan ukuran sekian maka secara praktis benda tersebut tidak akan terlihat oleh mata. *Radio Frequency Identification (RFID) tag* yang pasif ini memiliki jarak jangkauan yang berbeda mulai dari 10 mm sampai dengan 6 meter. *Radio Frequency Identification (RFID) tag* yang pasif harganya bisa lebih murah untuk diproduksi dan tidak bergantung pada baterai.

➤ *Radio Frequency Identification (RFID) aktif*

Radio Frequency Identification (RFID) tag yang aktif, di sisi lain harus memiliki *power supply* sendiri dan memiliki jarak jangkauan yang lebih jauh. Memori yang dimilikinya juga lebih besar sehingga bisa menampung berbagai macam informasi didalamnya. Jarak jangkauan dari *Radio Frequency Identification (RFID) tag* yang aktif ini bisa sampai sekitar 100 meter dan dengan umur baterai yang bisa mencapai beberapa tahun lamanya. Perbedaan sifat antara *Radio Frequency Identification (RFID) aktif* dan pasif dapat dilihat pada tabel dibawah ini

Radio Frequency Identification (RFID) tag juga dapat dibedakan berdasarkan tipe memori yang dimilikinya :

1. *Read / Write* (Baca/Tulis)

Memori baca/tulis secara tidak langsung sama seperti namanya, memorinya dapat dibaca dan ditulis secara berulang-ulang. Data yang dimilikinya bersifat dinamis.

2. *Read only* (Hanya baca)

Tipe ini memiliki memori yang hanya diprogram pada saat *tag* inidibuat dan setelah itu datanya tidak bisa diubah sama sekali. Data bersifat statis.

Frekuensi Radio dan Jangkauan

Ada empat macam *Radio Frequency Identification (RFID) tag* yang sering digunakan bila dikategorikan berdasarkan frekuensi radio, yaitu:

- ❖ *Low frequency tag* (antara 125 ke 134 kHz)
- ❖ *High frequency tag* (13.56 MHz)
- ❖ *UHF tag* (868 sampai 956 MHz)
- ❖ *Microwave tag* (2.45 GHz)

Jarak antara antena pembaca *Radio Frequency Identification (RFID)* dengan tag secara langsung dipengaruhi oleh frekuensi kerja yang digunakannya. Frekuensi *Radio Frequency Identification (RFID)* yang berbeda akan menghasilkan jangkauan yang berbeda pula. Frekuensi *Radio Frequency Identification (RFID)* yang digunakan pada Tugas Akhir ini adalah 13.56 MHz dengan menggunakan *Radio Frequency Identification (RFID) reader/writer* ACR 120 dengan tag Mifare 1 *kbyte* memiliki jarak operasi 10 cm. *Mifare 1 Kbytes (MF1 IC S50) Radio Frequency Identification (RFID) tag* yang digunakan pada KTM STT Telkom adalah *Mifare 1 Kbytes* dengan spesifikasi sebagai berikut *Mifare, RF Interface (ISO/IEC 14443 A)*

1. Pertukaran data secara *contactless* dan tidak dibutuhkan baterai untuk pertukaran data dan *supply energy*
2. Jarak operasi hingga 10 cm
3. Frekuensi operasi 13,56 MHz
4. Kecepatan transfer data 106 kbps

➤ **EEPROM**

1. EEPROM 1 *Kbytes*, 16 sektor dengan 4 blok tiap

sektor dengan masing-masing 16 *byte* (satu blok terdiri dari 16 *bytes*)

2. Lama penyimpanan 10 tahun
3. Kemampuan tulis 100.000 kali
4. Transport *key* melindungi akses ke EEPROM saat pengiriman *chip*

➤ *Security*

1. *Mutual three pass authentication* (ISO/IEC DIS 9798-2)
2. Enkripsi data pada kanal RF
3. Serial Number yang unik pada setiap *device*

1.2 Faktor *Radio Frequency Identification (RFID)*

Dalam pemakaian di bidang bisnis dan industri, *Radio Frequency Identification (RFID)* mungkin belum setenar dan sefamiliar dari pada *barcode*, hal ini disebabkan beberapa faktor, misal:

1. Faktor Harga, *Radio Frequency Identification (RFID)* tentunya jauh lebih mahal dibandingkan dengan *barcode* dikarenakan beberapa keunggulan dan fungsi nya yang bisa dikembangkan lebih luas lagi tidak sekedar untuk identifikasi unik dari suatu

aset atau barang.

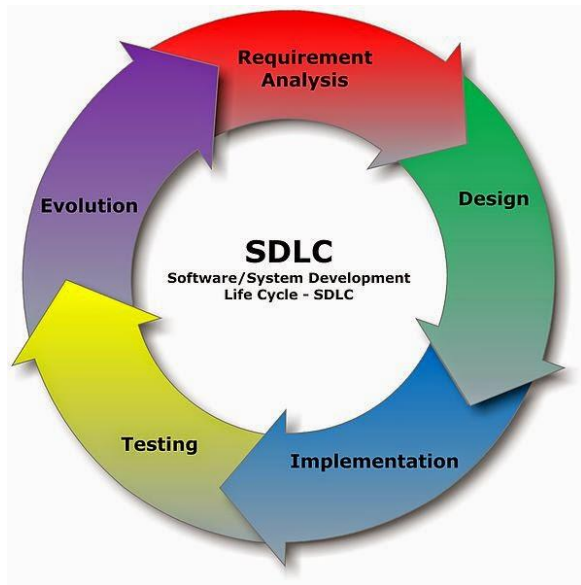
2. Faktor Bahan, Bahan untuk *barcode* biasanya hanya berupa label kertas dan plastik, sementara *Radio Frequency Identification (RFID) tag* dapat terbuat dari beberapa bahan seperti kertas, plastik, metal, *rubber/ silicon* dll sebagai bahan pembungkus chip rfid tersebut
3. Faktor Perangkat, semisal *reader* dan printer *Radio Frequency Identification (RFID)* masih sangat tinggi harganya kisaran *USD 3000* sampai dengan *USD 5000* berbeda dengan *reader barcode* yang hanya *100 – 500 USD*.
4. Aplikasi Pendukung, Untuk membuat aplikasi *Radio Frequency Identification (RFID)* pun tidak semudah membuat aplikasi dengan *barcode* dari sisi biaya pengembangan, dan bahasa aplikasi yang dibutuhkan tentunya *effort programmer* lebih tinggi dalam mengembangkan aplikasi rfid sehingga biaya pengembangan pun jauh lebih mahal.

Namun faktor faktor tersebut sebanding dengan kemampuan dari *Radio Frequency Identification (RFID)* tersebut yang memang lebih baik dari

barcode mungkin hingga beberapa tahun kedepan sebelum di temukan teknologi baru untuk menggantikan perangkat *Radio Frequency Identification (RFID)* ini[17].

1.3 Prototype

Prototipe adalah model kerja dasar dari pengembangan sebuah program (*software*) atau perangkat lunak. Prototipe dalam Bahasa Inggris “prototype” disebut juga dengan purwarupa. Prototipe biasanya dibuat sebagai model untuk tujuan demonstrasi atau sebagai bagian dari proses pengembangan atau pembuatan sebuah *software*.



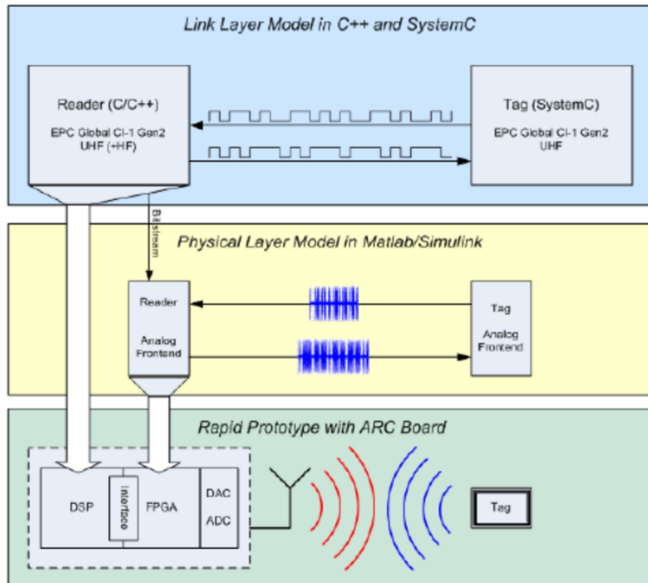
Gambar 1.4

Manfaat *Prototype*

Kata prototipe berasal dari Bahasa Latin, yaitu kata “proto” yang berarti asli, dan “*typus*” yang berarti bentuk atau model. Dalam konteks non-teknis, Prototipe adalah contoh khusus sebagai wakil dari kategori tertentu[18]

Dalam bidang desain, Prototipe atau purwarupa atau disebut juga dengan arketipe adalah bentuk awal sebagai contoh atau standar ukuran dari sebuah entitas. Sebuah Prototipe dibuat sebelum dikembangkan atau justru dibuat khusus untuk pengembangan sebelum dibuat dalam skala sebenarnya atau sebelum diproduksi secara

massal.



Gambar 1.5
RFID Prototyping System

Prototype adalah sebuah *Javascript Framework* yang dibuat untuk lebih memudahkan proses dalam membangun aplikasi berbasis *web*. Metode prototyping sebagai suatu paradigma baru dalam pengembangan sistem informasi, tidak hanya sekedar suatu evolusi dari metode pengembangan sistem informasi yang sudah ada, tetapi sekaligus merupakan revolusi dalam pengembangan sistem informasi manajemen

Ada 2 Jenis *Prototype* :

Jenis I : Suatu Sistem yang akan menjadi sistem operasional

Jenis II : Suatu model yang dapat dibuang yang berfungsi sebagai cetak biru bagi sistem operasional.

1.3.1 Karakteristik metode

Karakteristik metode *prototyping* meliputi langkah-langkah :

1. Pemilahan fungsi
2. Penyusunan Sistem Informasi
3. Evaluasi
4. Penggunaan Selanjutnya

1.4.2 Jenis-Jenis *Prototyping*

Jenis-jenis *prototyping* meliputi :

1. *Feasibility prototyping*
2. *Requirement prototyping*
3. *Desain Prototyping*
4. *Implementation prototyping*

1.4.3 Teknik-Teknik *Prototyping*

Teknik-teknik *prototyping* meliputi :

1. Perancangan Model
2. Perancangan Dialog

3. Simulasi

SISTEM YANG BERMANFAAT DARI PROTOTIPE (*SYSTEMS THAT BENEFIT FROM PROTOTYPING*)

Sejak kebutuhan (baca Spesifikasi Fungsi) pada umumnya berhubungan dengan pandangan user terhadap sistem, hanya dengan prototipe tampilan bagi *user* sudah cukup untuk memeriksa yang dibutuhkan. Menu-menu, bentuk tampilan input, tampilan keluaran, atau laporan yang dicetak, pertanyaan-pertanyaan, pesan-pesan merupakan calon yang ideal untuk prototipe.

Di lain pihak, perhitungan yang rumit, kumpulan update data dan *realtime*, dan sistem yang bersifat *scientific* sangat sulit untuk dijadikan model.

Sistem yang paling sesuai untuk prototipe adalah satu dari banyak hal yang bergantung pada sistem *input/output* dari *user*. Sistem dengan transaksi *on-line* dikendalikan melalui menu, layar, formulir, laporan, daftar dan perintah.

1.4.4 Keuntungan *Prototype*

Keuntungan dari prototipe :

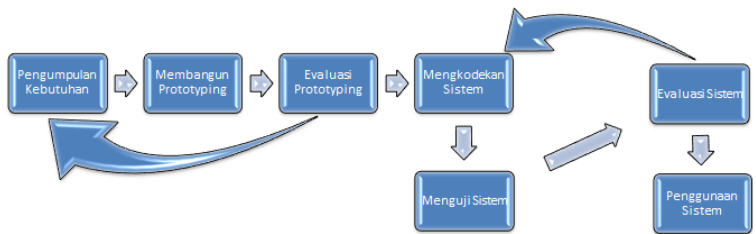
1. Menghasilkan syarat yang lebih baik dari produksi yang dihasilkan oleh metode ‘spesifikasi tulisan’.
2. *User* dapat mempertimbangkan sedikit perubahan selama masih bentuk prototipe.
3. Memberikan hasil yang lebih akurat dari pada perkiraan sebelumnya, karena fungsi yang diinginkan dan kerumitannya sudah dapat diketahui dengan baik.
4. *User* merasa puas. Pertama, *user* dapat mengenal melalui komputer. Dengan melakukan prototipe (dengan analisis yang sudah ada), *user* belajar mengenai komputer dan aplikasi yang akan dibuatkan untuknya. Kedua, *user* terlibat langsung dari awal dan memotivasi semangat untuk mendukung analisis selama proyek berlangsung[19].

1.4.5 Prose *Prototype*

Disini kita akan menjelaskan beberapa proses dalam *prototype* yaitu:

1. Proses perancangan dan analisi terlalu singkat.

2. Biasanya kurang fleksibel dalam menghadapi perubahan.
3. Pengembang ini kadang-kadang membuat kompromi implementasi dengan menggunakan sistem operasi yang tidak relevan dan algoritma yang tidak efisien.



Gamabr 1.6

Proses Prototype

BAGIAN 2 SEJARAH RFID

2.1 SEJARAH

Radio Frequency Identification (RFID) pertama kali diperkenalkan pertamakali sebagai alat *spionase* Pemerintah Rusia oleh Leon Theremin sekitar tahun 1945.



Gambar 2.1

Leon Theremin

Namun sebenarnya alat yang dipakai Theremin ini sebenarnya masih bersifat pasif sebagai alat pendengar dan bukan berujud suatu *identification tag*. Teknologi yang digunakan oleh *Radio Frequency Identification (RFID)* sendiri sebenarnya sudah ada sejak tahun 1920-an. Suatu teknologi yang lebih dekat dengan *Radio Frequency*

Identification (RFID), yang dinamakan *IFF transponder*, beroperasi pada tahun 1939 dan digunakan oleh Inggris pada Perang Dunia II untuk mengenali pesawat udara musuh atau teman. Implementasi *Radio Frequency Identification (RFID)* saat ini semakin menarik perhatian banyak karena digunakan oleh *supermarket* atau *retailer*.

2.2 TIPE DAN FREKUENSI *Radio Frequency Identification (RFID)*

Ada 3 tipe *tag Radio Frequency Identification (RFID)*, aktif, semi-pasif dan pasif. *RFID tag* dapat berupa pasif, aktif atau pasif dibantu baterai. *Radio Frequency Identification (RFID)* pasif tidak menggunakan baterai, sedangkan yang aktif memiliki baterai on-board yang selalu memancar atau menjadi suar sinyal. Sebuah baterai pasif dibantu (BAP) memiliki baterai kecil di papan yang diaktifkan ketika di hadapan sebuah pembaca *Radio Frequency Identification (RFID)*.

Suatu *Radio Frequency Identification (RFID) tag* adalah sebuah benda kecil, misalnya berupa stiker adesif yang dapat ditempelkan pada suatu barang atau produk. *Radio Frequency Identification (RFID) tag* berisi antena

yang memungkinkan peralatan itu menerima dan merespon terhadap suatu *query* yang dipancarkan oleh suatu RFID *transceiver*. Kebanyakan *Radio Frequency Identification (RFID) tag* mengandung setidaknya dua bagian: satu adalah sebuah sirkuit terpadu untuk menyimpan dan pengolahan informasi, modulasi dan demodulasi sebuah frekuensi sinyal radio (RF), dan fungsi khusus lainnya, yang lain adalah antena untuk menerima dan mengirimkan sinyal.

Pembaca (*reader*) *Radio Frequency Identification (RFID)* diklasifikasikan menjadi dua jenis: *Radio Frequency Identification (RFID)* tetap dan ponsel *Radio Frequency Identification (RFID)*. Jika pembaca membaca *tag* di posisi stasioner, hal itu disebut *Radio Frequency Identification (RFID)* tetap. Pembaca tetap adalah menetapkan zona interogasi tertentu dan menciptakan "gelembung" energi RF yang dapat dikontrol ketat. Hal ini memungkinkan area membaca sangat definitif pada saat *tag* masuk dan keluar dari zona interogasi. Di sisi lain, jika pembaca mobile ketika pembaca membaca *tag*, hal itu disebut ponsel *Radio Frequency Identification (RFID)*.

Tag ini murah untuk diproduksi dan cukup kecil untuk disisipkan pada item apapun. Aktif dan semi-pasif *tag*

bekerja pada baterai internal. *Tag* aktif menggunakan baterai untuk mengirim sinyal kepada pembaca, sedangkan *tag* semi-aktif tergantung pada alat pembaca dalam batas jangkauannya. *Tag* aktif dan semi-pasif, mengandung lebih banyak hardware dan karena itu lebih mahal. *Tag* ini digunakan untuk barang-barang mahal dan mampu menangkap data untuk jarak yang lebih jauh. *Tag* pasif bergantung sepenuhnya pada pembaca untuk sinyal. Sinyal untuk *tag* ini bisa mencapai jarak 20 kaki. Mereka lebih murah untuk memproduksi dan digunakan untuk *item* yang lebih murah. Sebuah botol shampo akan memiliki tag pasif, yang sekali pakai dengan botol shampo.

Ada tiga jenis penyimpanan data dalam *tag Radio Frequency Identification (RFID)*. Ini adalah baca-tulis, hanya membaca dan *WORM* (menulis pernah membaca berkali-kali). Data Sebuah *tag read-writes* dapat ditambahkan ke atau ditimpa. Baca *tag* hanya memiliki data yang hanya dapat dibaca, tidak ditambahkan atau ditimpa. *Tag WORM* dapat memiliki data tambahan tetapi tidak dapat ditimpa. *Tag Radio Frequency Identification (RFID) (transponder)* akan mengenali diri sendiri ketika mendeteksi sinyal dari perangkat yang hanya dapat dibaca saja (*Read only*) dibaca

dan ditulis (*Read/Write*) sekali tulis dan banyak baca (*write once read many*) juga tidak memerlukan kontak langsung maupun jalur cahaya untuk dapat beroperasi *Radio Frequency Identification (RFID)* dapat berfungsi pada berbagai variasi kondisi lingkungan dan menyediakan tingkat integritas data yang tinggi.

Radio Frequency Identification (RFID) tag yang pasif tidak memiliki *power supply* sendiri. Dengan hanya berbekal induksi listrik yang ada pada antena yang disebabkan oleh adanya frekuensi radio *scanning* yang masuk, sudah cukup untuk memberi kekuatan yang cukup bagi *Radio Frequency Identification (RFID)* tag untuk mengirimkan respon balik. Sehubungan dengan *power* dan biaya, maka respon dari suatu *Radio Frequency Identification (RFID)* yang pasif biasanya sederhana, hanya nomor *ID* saja. Dengan tidak adanya *power supply* pada *Radio Frequency Identification (RFID)* yang pasif maka akan menyebabkan semakin kecilnya ukuran dari *Radio Frequency Identification (RFID)* tag yang mungkin dibuat.

Ada empat macam *Radio Frequency Identification (RFID)* tag yang sering digunakan bila dikategorikan berdasarkan frekuensi radio, yaitu:

- *low frequency tag* (antara 125 ke 134 kHz)
- *high frequency tag* (13.56 MHz)

UHF *tag* (868 sampai 956 MHz), UHF *tag* tidak bisa digunakan secara global, karena tidak ada peraturan global yang mengatur penggunaannya.

Microwave tag (2.45 GHz)

2.3 SISTEM DAN CARA KERJA *Radio Frequency Identification (RFID)*

Suatu sistem *Radio Frequency Identification (RFID)* dapat terdiri dari beberapa komponen, seperti *tag*, *tag reader*, *tag programming station*, *circulation reader*, *sorting equipment* dan *tongkat inventory tag*. Keamanan dapat dicapai dengan dua cara. Pintu *security* dapat melakukan *query* untuk menentukan status keamanan atau *Radio Frequency Identification (RFID)* *tag*-nya berisi *bit security* yang bisa menjadi *on* atau *off* pada saat didekatkan ke *reader station*. Kegunaan dari sistem *Radio Frequency Identification (RFID)* ini adalah untuk mengirimkan data dari piranti *portable*, yang dinamakan *tag*, dan kemudian dibaca oleh *Radio Frequency Identification (RFID)* *reader* dan kemudian diproses oleh aplikasi komputer yang membutuhkannya. Data

yang dipancarkan dan dikirimkan tadi bisa berisi beragam informasi, seperti ID, informasi lokasi atau informasi lainnya seperti harga, warna, tanggal pembelian dan lain sebagainya.

Penggunaan *Radio Frequency Identification (RFID)* untuk maksud *tracking* pertama kali digunakan sekitar tahun 1980-an. *Radio Frequency Identification (RFID)* dengan cepat mendapat perhatian karena kemampuannya dalam *men-tracking* atau melacak *object* yang bergerak. Seiring dengan perkembangan teknologi, maka teknologi *Radio Frequency Identification (RFID)* sendiripun juga berkembang sehingga nantinya penggunaan *Radio Frequency Identification (RFID)* bisa digunakan untuk kehidupan sehari-hari.

Dalam suatu sistem *Radio Frequency Identification (RFID)* sederhana, suatu *object* dilengkapi dengan tag yang kecil dan murah. Tag tersebut berisi *transponder* dengan suatu *chip* memori *digital* yang di dalamnya berisi sebuah kode produk yang sifatnya unik. Sebaliknya, *interrogator*, suatu antena yang berisi *transceiver* dan *decoder*, memancarkan *sinyal* yang bisa mengaktifkan RFID tag sehingga dia dapat membaca dan menulis data ke dalamnya. Ketika suatu *Radio Frequency Identification (RFID) tag* melewati suatu *zone elektromagnetis*, maka dia akan

mendeteksi sinyal aktivasi yang dipancarkan oleh si *reader*. *Reader* akan men-*decode* data yang ada pada tag dan kemudian data tadi akan diproses oleh komputer.

Radio Frequency Identification (RFID) tag yang aktif, di sisi lain harus memiliki *power supply* sendiri dan memiliki jarak jangkauan yang lebih jauh. Memori yang dimilikinya juga lebih besar sehingga bisa menampung berbagai macam informasi di dalamnya. Sampai tulisan ini dipublikasikan, ukuran terkecil dari *Radio Frequency Identification (RFID) tag* yang aktif ini ada yang sebesar koin. Jarak jangkauan dari *Radio Frequency Identification (RFID) tag* yang aktif ini bisa sampai sekitar 10 meter dan dengan umur baterai yang bisa mencapai beberapa tahun lamanya. *RFID tag* yang pasif harganya bisa lebih murah untuk diproduksi dan tidak bergantung pada baterai. *Radio Frequency Identification (RFID) tag* yang banyak beredar sekarang adalah *Radio Frequency Identification (RFID) tag* yang sifatnya pasif.

2.4 KEUNGGULAN

Radio Frequency Identification (RFID) memiliki dua keunggulan yang membedakan dengan *barcode optic* yaitu :

- a. identifikasi unik sebuah *tag Radio Frequency Identification (RFID)* mampu merekam lebih banyak data transaksi secara unik dari jutaan objek yang identik seperti : *serial number, expired date* dan lain-lain. Sehingga informasi dari sebuah *item* yang menggunakan *Radio Frequency Identification (RFID) tag* dapat dengan mudah diketahui. Hal ini berbeda dibanding *barcode* yang hanya dapat mengidentifikasi tipe obyek tempat ia dicetak.
- b. segi otomasi, *Radio Frequency Identification (RFID)* menggunakan frekuensi radio untuk mengirimkan informasi atau data antara *Radio Frequency Identification (RFID) tag* dengan *Radio Frequency Identification (RFID) readernya*, sehingga tidak diperlukan kontak fisik diantara keduanya untuk dapat berkomunikasi. *Tag Radio Frequency Identification (RFID)* dapat dibaca tanpa kontak *line-of-sight* dan tanpa penempatan yang presisi, *Reader Radio Frequency Identification (RFID)* dapat melakukan *Scan* terhadap *tag-tag* sebanyak ratusan perdetik. Hal ini berbeda dengan *Barcode optic* yang pada saat melakukan *Scanning* memerlukan kontak *line-of-sight*

dengan *reader*, dan tentu saja peletakan fisik yang tepat dari objek yang *discan*. Kecuali pada lingkungan yang benar-benar terkontrol, scanning terhadap *barcode* memerlukan campur tangan manusia, sebaliknya *tag-tag Radio Frequency Identification (RFID)* Sebagai suksesor dari *barcode*, *Radio Frequency Identification (RFID)* dapat melakukan *control* otomatis untuk banyak hal.

Radio Frequency Identification (RFID) juga mudah untuk disembunyikan atau dimasukkan dalam *item* benda lainnya. Sebagai contoh, pada tahun 2009 para peneliti di Universitas Bristol berhasil merekatkan *Radio Frequency Identification (RFID) transponder mikro* untuk mempelajari kehidupan semut dan mempelajari perilaku mereka. *Hitachi* memegang rekor untuk *chip Radio Frequency Identification (RFID)* terkecil, di 0.05mm x 0.05mm. Ini adalah ukuran ke 1/64 ukuran pemegang rekor sebelumnya, *mu-chip*. Industri ini diaktifkan dengan menggunakan proses *silikon-on-insulator (SOI)*. Bentuknya yang sangat kecil seperti debu berukuran *chip* dapat menyimpan 38 digit. Hal ini kecenderungan semakin miniatur *Radio Frequency Identification (RFID)* kemungkinan akan berlanjut seiring

kemajuan teknologi.

2.5 MANFAAT DAN KEGUNAANNYA

Beberapa *Radio Frequency Identification (RFID)* komersial yang saat ini sudah beredar di pasaran ada yang bisa diletakkan di bawah kulit. Pada tahun 2005 tercatat bahwa *Radio Frequency Identification (RFID) tag* terkecil berukuran 0.4 mm x 0.4 mm dan lebih tipis daripada selembar kertas. Dengan ukuran sekian maka secara praktis benda tersebut tidak akan terlihat oleh mata. *Radio Frequency Identification (RFID) tag* yang pasif ini memiliki jarak jangkauan yang berbeda mulai dari 10 mm sampai dengan 6 meter.

Pada bisnis *Ritel*, *Radio Frequency Identification (RFID)* dipergunakan untuk melakukan *tracking* dan melakukan pencatatan terhadap seluruh inventori. Sistem *Radio Frequency Identification (RFID)* memungkinkan komunikasi antara produk-produk yang telah di-tag dengan *chip Radio Frequency Identification (RFID)* dengan *Radio Frequency Identification (RFID) reader* dan dengan *Server* lokal. Cara kerja sistem secara keseluruhan menyerupai dengan penggunaan *barcode* label dan *barcode scanner*,

tetapi jauh lebih mudah, praktis dan memuaskan, karena petugas tidak perlu melakukan *scanning* satu-per satu *item*, karena pada saat pelanggan melewati scanner (*RFID reader*) seluruh *item* akan langsung terdeteksi atau dihitung secara bersamaan.

Dari segi sistem, untuk implementasi *Radio Frequency Identification (RFID)* dibutuhkan sebuah server lokal, *Radio Frequency Identification (RFID) reader* dan *pre-encoded labels (tags)*. Server ini akan menyimpan seluruh data yang dibutuhkan dan aplikasi yang dapat membaca data dari *tag* melalui *Radio Frequency Identification (RFID) Reader*. *RFID Server* ini juga terintegrasi dengan *Inventory Management System* atau *POS System*. Baik *tag* maupun *Radio Frequency Identification (RFID) Reader* dilengkapi dengan antena sehingga dapat menerima dan memancarkan gelombang elektromagnetik yang memungkinkan untuk membaca multiple *tagged item* pada suatu saat. Sensitifitas antena dapat di -set untuk setiap register yang dilayaninya, setiap *reader* dapat meng-handel transmisi hingga 4 register pada saat bersamaan.

Radio Frequency Identification (RFID) dapat menjadi *barcode* generasi berikutnya yang dapat

dipergunakan untuk otomatisasi inventory control karena akan memberikan banyak kemudahan dan mengurangi biaya untuk distribusi barang dari pabrik atau vendor ke gudang. *Radio Frequency Identification (RFID)* tidak memerlukan kontak langsung dan sebuah *Radio Frequency Identification (RFID) Reader* dapat membaca semua *tag Radio Frequency Identification (RFID)* yang berada pada daerah jangkauannya. Dengan cara ini maka waktu untuk *inventory control* dapat dihemat. Contohnya : Pada saat sebuah *box* yang berisi ratusan *item* dikirim oleh vendor diterima oleh bagian gudang, maka personal gudang akan memeriksa isi *box* tersebut dengan menggunakan barcode scanner dan melakukan *scanning item* satu persatu. Tetapi dengan menggunakan *Radio Frequency Identification (RFID)*, seluruh item yang dikirim telah di-tag dengan *Radio Frequency Identification (RFID) chip*, yang dapat dibaca oleh *Radio Frequency Identification (RFID) Reader* seluruhnya pada saat bersamaan. Hal yang sama juga dapat dilakukan pada saat *Item Transfer* antar lokasi / gudang, *stock opname* dan lain sebagainya.

Radio Frequency Identification (RFID) juga dipergunakan untuk mempermudah dan mempercepat

transaksi pada sebuah Retail Store, antara lain untuk :

Smart Shelf : *Smart Shelf* yang berbasis *Radio Frequency Identification (RFID)* dapat mendeteksi keberadaan setiap *item* pada sebuah rak. Ketika sebuah *item* diambil dari rak, maka sistem dapat mendeteksi *item* yang diambil oleh pelanggan, memberikan tanda dan mencatat *item* yang diambil, sehingga dapat dilakukan *real-time shelf inventory*. Selain itu, perilaku pelanggan dapat dicatat dalam database dan dipergunakan untuk strategi marketing.

Pada saat pelanggan selesai berbelanja dan akan membayar di kasir (*check out*), maka *Radio Frequency Identification (RFID) Reader* secara otomatis mendeteksi seluruh *item (merchandise)* yang akan dibeli oleh pelanggan, hal ini biasanya dilakukan dengan melakukan scanning satu per satu *item* oleh kasir. *Radio Frequency Identification (RFID) Reader* membaca *Radio Frequency Identification (RFID) Chips* yang melekat pada setiap *item* melalui frekuensi radio, kemudian secara virtual melakukan scanning terhadap seluruh *item*. Kemudian *Radio Frequency Identification (RFID) Reader* akan mengkomunikasikan dengan Server untuk men-*generate* penjualan pada register secara otomatis.

Sales Return dapat dengan mudah dilakukan, karena sistem secara otomatis memeriksa barang yang dikembalikan, pelanggan dapat membawa atau mengembalikan *Radio Frequency Identification (RFID)* -tag pada *item* tanpa struk (*store receipt*), tag ini kemudian akan me-refer ke *database* untuk mengetahui waktu pembelian, harga beli saat itu (*original price*), bahkan informasi kartu kredit, dan lain-lain. Informasi detail tentang *Sales Return* ini juga akan membantu Store untuk mengupdate status stok dari item yang dikembalikan.

Manfaat *Radio Frequency Identification (RFID)* untuk pelaporan (*reporting*) akan lebih berkualitas dan lebih cepat, contohnya : Berbagai laporan tentang inventory dapat diketahui secara *real-time* dari Server Pusat, baik secara *OnLine* ataupun menggunakan metode sinkronisasi data, Retailer dapat mengakses data pada seluruh lokasi untuk mendapatkan laporan up-to-date mengenai stok barang. Dengan menggunakan *Radio Frequency Identification (RFID)*, Retailer bisa mengurangi permasalahan ‘kekurangan stok’, yang sering mengakibatkan ‘*lost sales*’, selain juga bisa mengurangi kepercayaan dan kepuasan pelanggan. Hal ini dimungkinkan karena status stok dapat dengan mudah di-

track untuk mendapatkan data yang akurat tentang suatu produk tertentu pada suatu saat, yang kemudian dihubungkan dengan *supply-chain*. Teknologi ini juga memungkinkan *Retailer* untuk menganalisa tingkat utilisasi pada suatu lokasi (*Store*) dan juga melakukan analisa produk per lokasi, sehingga *Retailer* bisa menyediakan produk yang bersifat custom kepada pelanggan pada lokasi tertentu.

Radio Frequency Identification (RFID) dapat dipergunakan untuk mengurangi tingkat kehilangan barang pada suatu *store*, karena *Radio Frequency Identification (RFID) tags* menempel pada setiap *item* dan setiap *item* yang dibawa oleh pelanggan dapat di-track apakah sudah dibayar atau belum. *Radio Frequency Identification (RFID)* juga dapat ditempatkan pada kartu pelanggan dan pada saat kartu tersebut di-scan pada saat pembayaran (*check out*) di konter, monitor POS dapat menawarkan produk-produk tambahan yang belum dibeli, berdasarkan data histori yang tersimpan di database. Wiraniaga dapat menggunakan *Radio Frequency Identification (RFID)* untuk membantu pelanggan mendapatkan barang sesuai kebutuhannya, misalnya : ukuran, warna, lokasi *item* di rak atau di gudang dan lain-lain, berdasarkan informasi yang disimpan pada *Radio Frequency*

Identification (RFID) tags menggunakan scanner.

Pemakaian *Radio Frequency Identification (RFID)* di perpustakaan misalnya pintu *security* ruang perpustakaan mampu mendeteksi buku-buku yang sudah dipinjam atau belum. Ketika seorang *user* mengembalikan buku, *security bit* yang ada pada *Radio Frequency Identification (RFID) tag* buku tersebut akan di-reset dan recordnya di ILS secara otomatis akan di-update. Pada beberapa solusi yang berbasis *Radio Frequency Identification (RFID)* maka slip pengembaliannya bisa di-generate secara otomatis pula. *Radio Frequency Identification (RFID)* juga mempermudah orang untuk menyortir barang.

Pada tahun 2010 tiga faktor kunci yang mendorong peningkatan signifikan dalam penggunaan *Radio Frequency Identification (RFID)*: penurunan biaya peralatan dan *tag*, meningkatkan kinerja untuk kehandalan 99,9% dan standar internasional yang stabil mengenai UHF *Radio Frequency Identification (RFID)* pasif. Dari perkembangannya pesat dan beberapa kelebihan RFID ini maka sistem-sistem *Radio Frequency Identification (RFID)* menawarkan peningkatan efisiensi dalam pengendalian inventaris, *logistic* dan manajemen rantai *supply*. Dimasa mendatang diperkirakan

jika *Radio Frequency Identification (RFID)* dapat semakin murah dan efektif maka *Radio Frequency Identification (RFID)* akan menjadi suksesor bagi penggantian seluruh *scanner barcode (barcode optic)*, mulai dari penjualan hingga integrasi dengan vendornya.

BAGIAN 3 LANDASAN TEORI

3.1 Tinjauan Studi

A. Tinjauan studi jurnal internasional

No	ol	Tahun	Nama Jurnal	Peneliti	Judul Penelitian
	7	2019	TELKO MNIK A	yafrial Fachri Pane, Rolly Maulana Awangga, Bayu Rahmad Azhari, Gilang Romadhanu Tartila	<i>FID-based conveyor belt for improve warehouse operations</i>
Kesimpulan					
Membuat sebuah alat yang dapat memisahkan 2 jenis barang yang ada di Gudang, seperti pecah belah dan bukan pecah belah, sebelum servo memisahkan jenis barang, barang yang memiliki tag <i>Radio Frequency Identification (RFID)</i> terlebih dahulu di scan oleh <i>reader</i> , kemudian selanjutnya akan di pisahkan oleh servo dan barang akan masuk ke gudang.					
2		2017	Appl. Technol. Secur.	M. Moh, L. Ho, Z. Walker, and	<i>A prototype on rfid and sensor</i>

			Priv	T. S. Moh	<i>networks for elder health care</i>
KESIMPULAN					
<p>Hasil Penelitian : Dalam penelitian ini mereka memiliki dua motes yaitu simulator pembaca RFID dan perangkat lunak simulator dengan PC yang sama, kedua motes tersebut dipisahkan berjarak 2 kaki dan dibekali baterai, kedua motes tersebut berkomunikasi melalui nirkabel. Tes awal mencapai sekitar 10 pesan <i>tag</i> dari 19 <i>byte</i> panjangnya. Untuk mencapai tingkat transfer yang lebih baik mereka memodifikasi motes tersebut, hasilnya mencapai sekitar 25 pesan <i>tag</i>, kebanyakan <i>Radio Frequency Identification (RFID)</i> komersial dapat menangani 50 hingga 100 <i>tag</i> per detik. Dengan demikian sistem ini dapat menangani sekitar 41 tag perdetik.</p>					

3	1	2007	IEEE	C. Floerkemeier, C. Roduner, and M. Lampe,	<i>RFID Application Development With the Accada Middleware Platform</i>	C. Floerke meier, C. Roduner, and M. Lampe,
KESIMPULAN						
<p>Hasil Penelitian : Disini penulis menggunakan <i>Radio Frequency Identification (RFID) Reader</i> yang di tempatkan di atas pintu masuk truk dan keluar truk dan dapat membaca <i>tag</i> secara cepat dan tepat tanpa harus membongkar isi muatan di dalam truk.</p>						

4	3	2008	IEEE	X. Pang, X. Yao, C. P. Liang, and W. Hong	<i>Design and realization of a highly integrated UHF RFID reader</i>
---	---	------	------	---	--

					<i>module</i>
KESIMPULAN					
Hasil penelitian : penulis ini menggunakan <i>tag</i> yg berfrekuensi 902-928 MHz jarak <i>reader</i> ke <i>tag</i> mencapai 6 meter.					
5		2001	IEEE	M. Murad, A. Rehman, A. A. Shah, S. Ullah, M. Fahad, and K. M. Yahya	<i>RFAIDE - An Radio Frequency Identification (RFID) based navigation and object recognition assistant for visually impaired people</i>
KESIMPULAN					
Hasil penelitian : Disini peneliti mengembangkan <i>Radio Frequency Identification (RFID)</i> untuk tuna netra, yang di tanamkan di ujung tongkat yang dapat membaca					

	<p><i>tag</i> hingga 100 m, serta <i>tag</i> nya berada di berbagai tempat dan benda sehingga tuna netra ketika berjalan sudah mengetahui posisi benda dan jala yang ia lalui.</p>
--	--

B. Tinjauan Studi Jurnal Nasional

Tabel 3.1 Tinjauan Studi Jurnal Nasional

No	Jl	Tahun	Nama Jurnal	Peneliti	Judul Penelitian
		2012	Conf. Nas. ICT-M Politek. Telkom	. Fahrudin	encatatan dan Pemantauan Kehadiran Perkuliahan di Lingkungan Politeknik Telkom Berbasis RFID dan Aplikasi <i>Web</i>
Kesimpulan					
<p>Hasil penelitian : Penelitian ini menggunakan RFID untuk sistem absensi nya, baik bagi dosen maupun mahasiswa yang dimana sistem ini dapat menekan tingkat DO bagi mahasiswa yang sering tidak absen dan dapat menekan dosen agar datang <i>on-time</i>[19].</p>					

2	8	2009	Tek. Elektro FTII	I. Winarsih and R. Mahendra	Sistem Parkir Otomatis Menggunakan Rfid Berdasarkan Mikrokontroler At 89S51
	Kesimpulan				
	Hasil penelitian : Penelitian ini menggunakan serial RS-232 dan <i>Hyperterminal</i> untuk melihat ID kartu yang dimiliki oleh masing <i>tag</i> RFID yang berhasil dibaca oleh <i>RFID Reader</i> [20].				
3	6	2004	J. Tek. Ind	Z. J. H. Tarigan	Integrasi Teknologi RFID Dengan Teknologi ERP Untuk Otomatisasi Data

Kesimpulan

Hasil penelitian : Penelitian ini menggunakan RFID dimana pada *tag* RFID mereka memasukkan tanggal proses, waktu masuk ke gudang, kode produk, lokasi peletakan dan nama penanggung jawab. Sehingga perusahaan tersebut memiliki keuntungan antara lain pengendalian inventori yang lebih baik, waktu pengiriman yg lebih cepat dan penggunaan tenaga kerja yg efektif[21].

Perbedaan : Penulis menggunakan tag yg berfrekuensi 12,5KHz, 13,56KHz dan *microwave*

4	11	2009	J. Tek. Ind	Iwan Vanany and Awaluddin Bin Mohamed Shaharoun	Pengadopsian Teknologi Rfid Di Rumah Sakit Indonesia, Manfaat Dan Hambatannya
---	----	------	----------------	---	---

	Kesimpulan				
	<p>Hasil penelitian : Penelitian ini menjelaskan keutungan penerapan RFID di rumah sakit antara lain memudahkan staf medis dalam mengidentifikasi pasien dan kemudahan dalam penelurusan keberadaan peralatan medis yang diperlukan sehingga pasien tidak terlalu lama melakukan proses administrasi dan menunggu operasi karna peralatan medis belum di siapkan[22].</p>				
5	4	2009	J. Inform. Mulaw arman	D. Cahyadi	Desain Sistem Absensi PNS Berbasis Teknologi RFID
	Kesimpulan				
	<p>Hasil penelitian : peneliti ini menggunakan RFID untuk sistem Absensi PNS yg bertujuan untuk mencegah titip tanda tangan, tanda tangan melebihi batas waktu, boros kertas dan tinta[23].</p>				

3.2 Gudang

Gudang merupakan salah satu bagian penting dari sebuah pabrik atau perusahaan yang berfungsi sebagai tempat penyimpanan, baik barang hasil produksi maupun bahan baku yang akan diproduksi perusahaan tersebut. Jika dilihat dari segi fungsi dapat diketahui bahwa tingkat mobilitas barang dalam gudang sangat tinggi setiap harinya, hampir terdapat ratusan bahkan ribuan barang produksi maupun bahan baku masuk atau keluar gudang[13].

Salah satu tempat yang paling penting dalam kegiatan *logistic* yakni gudang. Pada artikel ini kami akan sampaikan 10 Tipe dan jenis gudang berdasarkan karakteristik penyimpanan. Secara umum ada 5 jenis tipe gudang, yaitu gudang gudang pribadi, gudang publik, gudang otomatis, gudang dengan pengaturan iklim/suhu. Dan gudang pusat distribusi, berikut ini penjelasan dari gudang-gudang tersebut.



Gambar 3.1
Tipe Gudang

1. Gudang Pribadi/Swasta

Gudang pribadi ini dioperasikan oleh para pemasok dan *reseller* ini biasanya digunakan untuk kegiatan distribusi mereka sendiri. Contohnya: sebuah jaringan ritel mengoperasikan dan menyediakan gudang dan mereka menerima dan mendistribusikan produknya untuk tokonya sendiri.

2. Gudang Publik/Umum

Gudang publik, sebuah gudang yang disewakan, biasanya ini disewakan hanya untuk distribusi jangka pendek. Dan biasanya

ini digunakan perusahaan yang sudah memiliki gudang, dan mencari ruang penyimpanan tambahan, ini karena keterbatasan kapasitas gudang.

3. Gudang Otomatis

Gudang ini dioperasikan secara otomatis, atau dilakukan dengan teknologi robotika. Tahapan otomatis sampai pada pemakaian *conveyor belt*, ini biasanya digunakan untuk mengangkut barang sehingga meminimalkan kebutuhan SDM.

4. Gudang dengan Pengaturan Iklim/Suhu (*Climate Controlled Warehouse*)

Gudang tipe ini dikhususkan untuk menyimpan barang yang produk nya membutuhkan suhu dingin seperti *freezer*. Gudang ini untuk produk yang membutuhkan kelembaban atau udara tertentu, untuk menjaga kualitas produknya, misalnya produk beku.

5. Gudang Pusat Distribusi (*Distribution Centre*)

Beberapa gudang ada yang menyimpan produknya dengan jangka waktu yang sangat cepat. Gudang ini hanya berfungsi sebagai titik penerima berbagai pemasok dan segera akan dikirimkan ke berbagai pelanggan.

10 Gudang berdasarkan karakteristik penyimpanan

Itulah tadi beberapa tipe dan jenis gudang secara umum, lalu berdasarkan karakteristik penyimpanannya gudang dikategorikan dalam 10 tipe.

1. Gudang Penyimpanan Bahan Baku

Gudang ini digunakan untuk menyimpan bahan baku yang digunakan untuk proses produksi. Dan sudah dipastikan gudang ini pasti berdekatan dengan tempat proses produksi. Digudang ini biasanya untuk menyimpan karet, biji besi, agregat untuk bahan baku material beton.



Gambar 3.2

Gudang Penyimpanan Bahan Baku

2. Gudang Tempat Penyimpanan Barang Setengah Jadi

Proses manufaktur, ada beberapa bahan yang melalui beberapa proses produksi, ini menjadi produk setengah jadi. Dan masih melalui proses panjang, tentu saja ada beberapa proses dalam manufaktur, yang memerlukan bahan setengah jadi.



Gambar 3.3

Gudang Tempat Penyimpanan Barang Setengah Jadi

3. Gudang Penyimpanan Bahan Hasil Produksi

Gudang ini digunakan untuk menyimpan barang hasil produksi, atau barang yang siap untuk didistribusikan kepada konsumen. Dan gudang ini digunakan untuk tempat penyimpanan yang difungsikan sebagai buffer atau safety stock dari permintaan

pasar.



Gambar 3.4

Gudang Penyimpanan Hasil Produksi

4. Gudang Sebagai Pusat Konsolidasi dan Transit

Gudang ini digunakan untuk menerima barang dari berbagai asal, lalu disitulah terjadi proses penggabungan untuk diteruskan pada konsumen. Atau dikirimkan untuk dilanjutkan proses produksi lainnya.



Gambar 3.5

Gudang sebagai Pusat Konsolidasi dan Transit

5. Gudang Sebagai Pusat *Transshipment*

Gudang ini digunakan untuk menerima barang dalam jumlah yang cukup besar, ataupun dari berbagai *supplier*. Setelah itu di dalam gudang ini dilakukan proses pembagian barang dalam jumlah yang lebih kecil. selanjutnya dilakukan proses pengiriman barang ke beberapa lokasi tujuan.



Gambar 3.6

Pusat Transshipment

6. Gudang yang Berfungsi Sebagai *Cross Docking*

Salah satu gudang yang efisien dan bisa merespon dengan cepat dalam menangani perpindahan barang. Gudang ini digunakan untuk menyimpan beberapa produk dengan waktu yang sangat singkat, misalnya hari itu diterima dan hari itu juga dikirimkan. Biasanya digunakan untuk loading barang dari truk ke truk, tetapi memang implementasinya tidak mudah. Dalam proses ini ada beberapa persyaratan agar, proses *cross docking* bisa

berjalan dengan lancar, misalnya setiap barang harus sudah diberikan label dan barang dalam kondisi siap kirim.



Gambar 3.7

Cross Docking

7. Gudang Sebagai Pusat Sortir

Gudang ini biasanya digunakan oleh perusahaan yang terjun dalam bentuk jasa, seperti pengiriman surat, dan parsel. Atau persewaan palet yang akan didistribusikan ke tempat-tempat lainnya. Dalam proses ini awalnya semua barang dikumpulkan di gudang tersebut untuk disortir. Setelah itu barang tersebut akan di proses sortir berdasarkan kode pos ataupun zona. Setelah itu barang dikumpulkan akan dikonsolidasikan dan dilakukan pengiriman berdasarkan area tersebut. Beberapa gudang pusat

sortir ini sudah menggunakan otomasi untuk mempermudah proses sortirnya.



Gambar 3.8

Pusat Sortir

8. Gudang *Fulfillment*

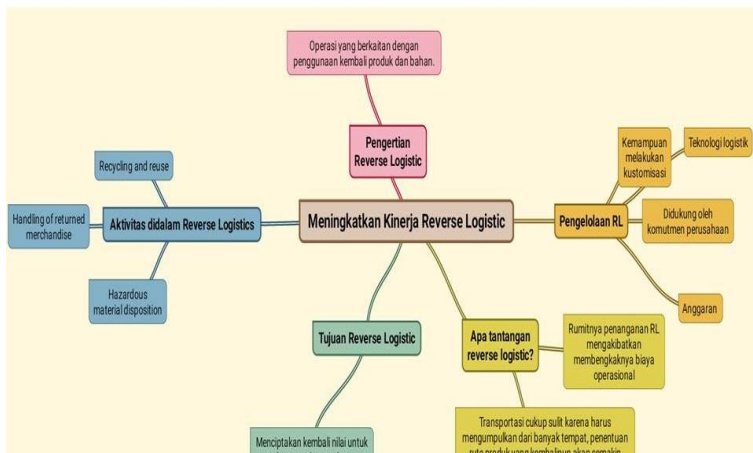
Gudang ini dibuat untuk mengelola pengiriman barang dengan jumlah atau volume yang sangat besar. Misalnya gudang yang dikelola oleh perusahaan *e-commerce*.



Gambar 3.9
Gudang *Fulfillment*

9. Gudang yang Difungsikan untuk Proses *Reverse Logistics*

Manfaat gudang ini digunakan untuk menyimpan barang yang akan di retur atau *defective*. Proses yang ada di dalam gudang ini proses pengecekan kembali barang retur, *defective*, proses perbaikan atau mengambil beberapa barang yang rusak. Untuk digunakan lagi, atau dimusnahkan barang-barang tersebut.



Gamabr 3.10

Kinerja *Reverse Logistics*

10. Gudang untuk Kepentingan Publik

Tidak hanya gudang yang dikomersilkan, namun ada juga gudang yang dikelola oleh Negara. Misalnya gudang bulog, gudang ini digunakan untuk menyimpan beras, milik tentara atau barang bantuan bencana. Barang yang disimpan dalam gudang tersebut misalnya, seragam, perlengkapan kantor, komputer. Yang intinya digunakan untuk menyimpan barang kepentingan public ini dikelola oleh Negara atau perusahaan pihak ketiga



Gamabr 3.11
Gudang bulog

3.2.1 Fungsi Dasar Manajemen Gudang

1. *Receiving*

Menerima barang.

2. *Inspection and quality control*

Melakukan pemeriksaan kebenaran barang yang diterima, kesesuaian dengan pesanan dan pemeriksaan spesifikasi barang (pemeriksaan Kualitas barang).

3. *Repackaging*

Pengepakan ulang.

4. *Put away*

Menempatkan barang yang baru datang di rak atau lokasi penempatan barang.

5. *Storage*

Menyimpan dan menjaga barang hingga diperlukan. Fungsi Dasar Manajemen Gudang

6. *Order picking*

Mengambil barang dari rak penempatan sesuai dengan permintaan yang diterima.

7. *Postponement*

Mengepak barang di dalam box untuk kemudahan dalam material handlingnya.

8. *Packing and shipping*

Melakukan pengecekan kebenaran dan kelengkapan pesanan. Melakukan pengepakan barang dan membuat dokumen *packing list*. Menyiapkan *container* untuk shipping Menyiapkan dokumen yang diperlukan untuk timbang dan muat kedalam *truck container*.

9. *Cross-docking*

Melakukan pengiriman barang sebelum barang dibongkar di

gudang. Barang yang diterima langsung dikirimkan pada pelanggan.

10. *Replenishing*

Pemesanan kembali jika *stock* sudah sampai batas minimum yang ditetapkan

3.2.2 Manfaat Gudang

1. Terjaganya kualitas dan kuantitas logistik dan peralatan.
2. Tertatanya logistik dan peralatan.
3. Peningkatan pelayanan pendistribusian.
4. Tersedianya data dan informasi yang lebih akurat dan aktual.
5. Kemudahan akses dalam pengendalian dan pengawasan.
6. Tertib administrasi.

3.3 SISTEM

Sistem adalah suatu jaringan kerja dari prosedur yang saling berhubungan, berkumpul bersama untuk melakukan suatu kegiatan atau untuk menyelesaikan suatu sasaran yang tertentu[14]. Sistem akses ruangan yang diimplementasikan memiliki spesifikasi sistem sebagai berikut:

1. *Input* data identifikasi menggunakan *Radio Frequency Identification (RFID)* tag (kartu RFID) dengan format

EM4001. Kartu *Radio Frequency Identification (RFID)* memiliki kode identifikasi bersifat unik yang tersimpan pada chip semikonduktor dan akan dikirimkan secara *wireless* dan *contactless* melalui antena, di mana keduanya tertanam di dalam kartu *Radio Frequency Identification (RFID)*. Kartu *Radio Frequency Identification (RFID)* berformat EM4001 ini dilengkapi dengan identitas personal pada bagian luar kartu berupa nama, nomor registrasi (NIM/NRP), keterangan jabatan, nomor *Radio Frequency Identification (RFID)*, durasi berlakunya kartu serta *barcode* yang merupakan kode identifikasi sebelumnya. *Radio Frequency Identification (RFID) tag* jenis ini hanya bersifat dapat dibaca saja (*read only*) dengan frekuensi kerja 125 kHz dan kecepatan transfer data sampai 4 kbps. Data kode identifikasi yang tertanam pada chip dan informasi pada bagian depan kartu tersimpan juga dalam program database pada PC. Chip yang terdapat di dalam kartu memiliki total kapasitas sebesar 8 *bytes*. Jumlah kartu *Radio Frequency Identification (RFID)* yang digunakan pada sistem ini adalah 10 (sepuluh) buah dimana 9 (sembilan) buah identitas kartu *Radio Frequency Identification (RFID)* disimpan dalam database mikrokontroler dan PC, sedangkan 1 (satu) buah identitas kartu *Radio Frequency Identification (RFID)* tidak

dimasukkan ke dalam database sebagai identitas di luar sistem guna pengontrolan dan pengujian validasi sistem.



Gambar 3.12

Kartu *Radio Frequency Identification (RFID)* EM 4001

2. Pendeteksi *Radio Frequency Identification (RFID)* tag menggunakan *Radio Frequency Identification (RFID)* reader dengan chip ID-12. RFID reader pada sistem ini mempergunakan *Radio Frequency Identification (RFID)* starter kit buatan *Innovative Electronics* berbasis *Radio Frequency Identification (RFID)* reader tipe ID-12 yang telah dilengkapi dengan jalur komunikasi RS-

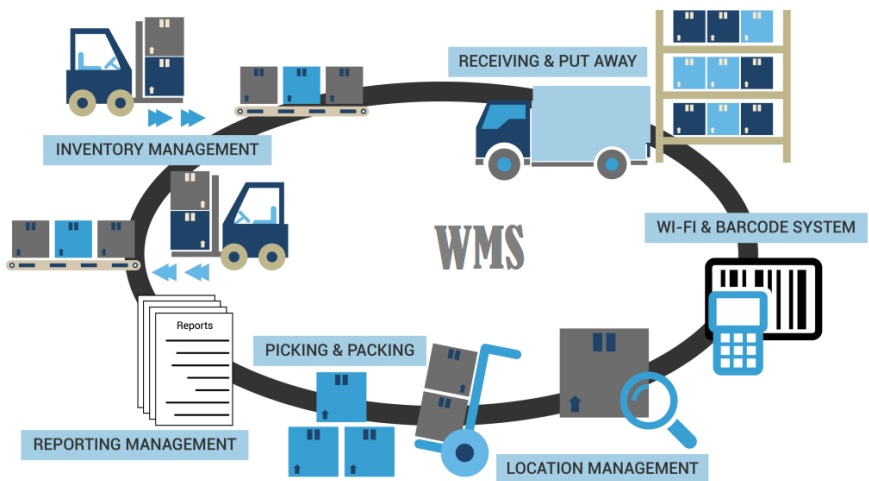
232, Frekuensi kerja dari *Radio Frequency Identification (RFID)* reader ini adalah 125 kHz dengan jarak baca kartu *Radio Frequency Identification (RFID)* maksimal sejauh 12 cm serta menggunakan catu daya dengan tegangan sebesar 9 – 12 VDC[5].



Gambar 3.13
RFID Starter Kit

3.4 Warehouse Management System

Warehouse Management dirancang bertujuan untuk mengontrol kegiatan pergudangan. Yang diharapkan dari pengontrolan ini adalah terjadinya pengurangan biaya-biaya yang ada di dalam gudang, pengambilan dan pemasukan barang ke gudang yang efektif dan efisien, serta kemudahan dan keakuratan informasi *stock* barang di gudang. Sistem informasi mengenai manajemen pergudangan ini sering disebut dengan *Warehouse Management System* (WMS).[15]



Gambar 3.14

Warehouse Management System

Warehouse Management System (WMS) atau Sistem Manajemen

Pergudangan merupakan kunci utama dalam *supply chain*, yang mana tujuan utamanya adalah mengontrol segala proses yang terjadi di dalamnya seperti *shipping* (pengiriman), *receiving* (penerimaan), *putaway* (penyimpanan), *move* (pergerakan), dan *picking* (pengambilan).

Tujuan dari *Warehouse Management System* (WMS) adalah untuk menyediakan satu set prosedur komputerisasi untuk menangani penerimaan dan pengiriman barang, mengelola fasilitas penyimpanan, mengelola stok barang untuk *picking*, *packing* dan *shipping*. [16]

Kelebihan di *Warehouse Management System* (WMS) merupakan penerapan WMS di suatu pergudangan Bisa mempercepat lead time proses yaitu dengan adanya proses yang dilakukan dengan cara komputerisasi atau otomatis yang sebelumnya wajib dengan cara manual dan dilakukan banyak orang. Dengan *Warehouse Management System* (WMS) kita mengetahui semua transaksi inventory dan jumlah stock dengan lebih cepat dan akurat dalam waktu kapan pun (*real time*). Dengan *Warehouse Management System* (WMS) kita Bisa mengatur lokasi penyimpanan barang dengan optimal. Jumlah dan tipe barang yang akan masuk ke gudang akan Bisa diatur penyimpanannya dengan tool yang ada dalam sistem.

Ada beberapa kelebihan atau keuntungan menggunakan *Warehouse Management System* (WMS), yaitu :

1. Mempercepat waktu proses dengan cara otomatis.
2. Untuk mengetahui transaksi inventori dan jumlah stok lebih cepat dan akurat kapan pun juga.
3. Mengatur lokasi penyimpanan barang secara teratur.

3.5 Inbound

inbound logistics merupakan pergerakan ke dalam perusahaan yang menunjukkan aliran material dari pemasok ke pabrik atau dinas operasi[15].



Gambar 3.15

Proses Inbound

Proses Dasar *Inbound*:

- Terima Informasi *Inbound*
- Cek *Inbound Receipt*, apakah informasi yang diberikan sudah sesuai dengan *standard order receipt*.
- Cek ketersediaan Gudang
- Alokasi barang akan ditaruh di mana?
- Cek sumber daya yang dibutuhkan
- Man Power
- *Tools* (*forklift, handpallet, Notebook, barcode scanner*, dan alat handling lainnya)
- Buat Jadwal *Inbound*.

Hari H :

- Mover datang
- Cek informasi dari Mover
- *Order Receipt*
- Surat Jalan
- *Data Mover*
- Cek Kesesuaian dengan informasi *inbound*

- Siapkan *Resources*
- Lokasi
- *Man Power*
- *Tools (forklift, handpallet, Notebook, barcode scanner, dan alat handling lainnya).*

Proses *Inbound* hari H

- Cek kondisi barang sebelum turun
 - Lakukan pengecekan visual dan ambil foto kondisi barang sebelum diturunkan.
- Turunkan barang
- Lakukan proses pendataan barang
 - *By systemo*
 - *By tally sheet*
- Jika ada problem di barang.
 - Ketidaksesuaian barang (jenis dan kuantiti)
 - Kerusakan barang
 - Catat, foto kondisi barang dan dilaporkan di Berita Acara

Akhir Proses

- Tanda tangan order receipt dan surat jalan mover untuk menyatakan barang sudah diterima.
- Buat berita acara kelengkapan/ketidaklengkapan barang
- Dokumentasi dokumen *inbound* (*copy* dan atau *scan*)Informasikan kepada customer bahwa barang sudah diterima dengan informasi kondisi barang. Informasi tersebut juga menyertakan dokumen pendukung, seperti:
 - *Copy order receipt*
 - *Copy* surat jalan
 - *Copy* berita acara
 - Foto kondisi barang pada saat sebelum *inbound* dan atau kondisi barang yang mengalami masalah.

3.6 *Oubound*

Outbond logistics merupakan pergerakan produk keluar pabrik atau dinas operasi menuju ke pelanggan atau konsumen[15]. *Outbound* adalah sebuah proses dimana seseorang mendapatkan pengetahuan, keterampilan dan nilainilainya langsung dari pengalaman memunculkan sikap sikap saling mendukung,

komitmen, rasa puas dan memikirkan masa yang akan datang yang sekarang tidak diperoleh melalui metode belajar yang lain. *Outbound* dalam pengertian lainnya adalah cara menggali diri sendiri, dalam suasana menyenangkan dan tempat penuh tantangan yang dapat menggali dan mengembangkan potensi, meninggalkan masa lalu, berada di masa sekarang dan siap menghadapi masa depan, menyelesaikan tantangan, tugas-tugas yang tidak umum, menantang batas pengamatan seseorang, membuat pemahaman terhadap diri sendiri tentang kemampuan yang dimiliki melebihi dari yang dikira[6]. Pengertian lain menyatakan bahwa *outbound* adalah sebuah petualangan yang berisi tantangan, bertemu dengan sesuatu yang tidak diketahui tetapi penting untuk dipelajari, belajar tentang diri sendiri, tentang orang lain dan semua tentang potensi diri sendiri[6]. *Outbound* adalah sebuah cara untuk menggali dan mengembangkan potensi anak dalam suasana yang menyenangkan.



Gambar 3.16
Proses *Outbound*

3.7 Lima Area Dalam *Inbound* dan *Outbound* Logistik

1. Order Processing

Order processing adalah aktivitas untuk pemenuhan pesanan. Proses *Order processing* atau fasilitasnya biasanya disebut dengan *distribution centers*. *Order processing* umumnya digunakan untuk mendeskripsikan proses atau aliran kerja yang berhubungan dengan pengambilan, pengemasan, dan pengiriman barang.

2. *Inventory*

Persyaratan *inventory* dari perusahaan secara langsung berhubungan dengan fasilitas jaringan dan level permintaan dari *customer*. Tujuan utama dari *inventory* yakni memenuhi keinginan konsumen dengan minimum *inventory*. Kelebihan *inventory* dapat menyebabkan inefisiensi logistik sehingga menyebabkan naiknya biaya gudang. *inventory* adalah semua *raw material*, *work in process*, dan *finish good* dalam sebuah *supply chain*. *Inventory* ini ada karena adanya perbedaan antara jumlah permintaan dan penawaran, yang digunakan untuk memenuhi permintaan di masa yang akan datang.

3. *Transportation*

Transportasi adalah area operasional dari logistik yang memindahkan dan meletakkan persediaan di berbagai poin lokasi. Transportasi merupakan hal yang penting dan membutuhkan biaya sehingga transportasi membutuhkan perhatian dari managerial.

4. *Warehousing, Materials Handling and Packaging*

Warehousing, Materials Handling and Packaging merupakan bagian terintegrasi dari area logistik. Heragu (2008) mendefinisikan bahwa pergudangan merupakan suatu aktivitas

yang membutuhkan waktu, tetapi tidak memberikan nilai tambah pada produk. Hal ini karena proses di pergudangan membutuhkan waktu dan tenaga sehingga menimbulkan biaya. Namun keberadaan gudang sangat penting karena membantu kelancaran proses produksi suatu pabrik. Gudang juga dapat membantu dalam memberikan *service* yang lebih baik untuk *customer* dan lebih responsif terhadap permintaan *customer*.

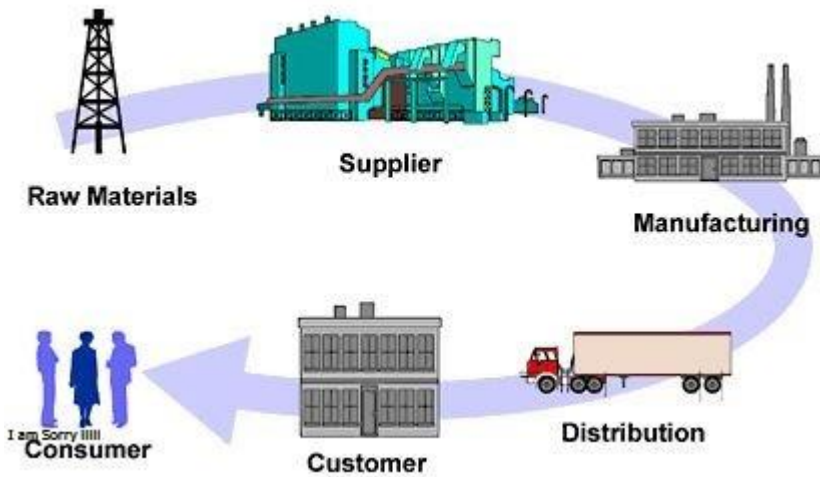
5. *Facility Network Design*

Facility Network System fokus pada penentuan jumlah, lokasi, dan kepemilikan dari semua fasilitas logistik yang diperlukan dalam aktivitas logistik. Itu merupakan suatu kebutuhan untuk menentukan *inventory* dan berapa banyak harus men-stok dalam setiap area konsumen. *Facility network* menciptakan sebuah struktur dari operasi logistik. Jaringan mengintegrasikan kapabilitas informasi dan transportasi. Pekerjaan seperti memproses permintaan konsumen, pergudangan, dan material *handling* semua dilakukan dalam *facility network*. Jadi dengan *facility network design* adalah sebuah struktur dari operasi logistik yang kompleks yang meliputi sejak dari penentuan jumlah stok di setiap area konsumen, penentuan *inventory*, penentuan lokasi, penentuan

kebutuhan akan peralatan logistik, material *handling*, pergudangan, hingga memproses permintaan konsumen. Dengan kata lain *facility network design* merupakan rangkaian dari empat area yang digambarkan dalam aktivitas *inbound* dan *outbound* logistik[15].

3.8 Logistik

Logistik adalah segala sesuatu yang berujud dan dapat digunakan untuk memenuhi kebutuhan dasar hidup manusia yang terdiri atas sandang, pangan dan papan atau turunannya. Termasuk dalam kategori logistik adalah barang yang habis pakai atau dikonsumsi, misalnya: sembako (sembilan bahan pokok), obat-obatan, pakaian dan kelengkapannya, air, tenda, jas tidur dan sebagainya[14]



Gambar 3.17

Sistem Logistik

3.8.1 Kualitas Layanan Logistik

Layanan merupakan strategi untuk meraih pangsa pasar dalam menghadapi persaingan. Jika layanan baik dan berkualitas maka konsumen akan mendapat kepuasan dan dihargai sehingga akan tetap merasa senang untuk menjadi pelanggan, demikian juga sebaliknya jika layanan buruk dan tidak berkualitas, konsumen akan merasa kecewa dan tidak puas atas layanan yang didapatnya. Layanan yang dikerjakan secara profesional akan memberikan keuntungan. Jika layanan diabaikan maka bisa

menimbulkan rasa tidak puas dipihak pelanggan dan ini jelas merugikan suatu organisasi atau perusahaan.

Sejak pertengahan 1980-an, kualitas layanan telah menjadi tema prioritas di kedua penelitian pemasaran dan logistik, berjalan paralel dengan kepentingan dalam kualitas, manajemen mutu dan kepuasan dalam perusahaan. keunggulan logistik telah diakui sebagai suatu bagian dimana perusahaan dapat menciptakan keunggulan kompetitif sebagian karena dampak terlihat pada layanan pelanggan[7].

Pendekatan kualitas layanan logistik, pada umumnya merupakan upaya untuk memahami kepuasan pelanggan dari perspektif perbedaan antara persepsi pelanggan dan layanan pelanggan yang sebenarnya pada berbagai atribut(dimensi-dimensi Kualitas Layanan[7].

3.8.2 Dimensi Kualitas Layanan Logistik

Menurut Mentzer,dimensi kualitas layanan distribusi sama halnya dengan kualitas layanan logistik (LSQ)[7] yang diaman ada 9 (sembilan) dimensi yang terdiri dari:

1.Personnel Concept Quality (Kualitas Personal atau Kontak Person)

Personnel contact quality mengacu pada orientasi pelanggan terhadap kontak person. Secara khusus, pelanggan peduli apakah personel dari kontak person memiliki pengetahuan yang luas, empati terhadap situasi, dan dapat membantu dalam menyelesaikan setiap masalah. Dalam layanan,

yang paling baik adalah persepsi kualitas yang terbentuk selama pelayanan. Persepsi kualitas layanan adalah terikat lebih ke proses layanan, yang melibatkan kontak person yang akhirnya mampu menghasilkan kepuasan. Dengan demikian *personnel contact quality* merupakan aspek penting sebagai penghubung antar karyawan dengan pelanggan.

2.Order Release Quantities (Ketersediaan Produk)

Ordering Release Quantities berkaitan dengan konsep ketersediaan produk. Perusahaan atau organisasi harus dapat memberikan tantangan kepada pelanggannya untuk memesan dalam jumlah berapapun. Pelanggan akan menjadi sangat puas ketika mereka dapat memperoleh jumlah yang mereka inginkan. Pentingnya ketersediaan produk kapan saja dapat dipenuhi akan menjadi salah satu komponen keunggulan dari logistik.

3.Information quality (Kualitas Informasi)

Information quality mengacu pada persepsi pelanggan terhadap informasi yang diberikan oleh pemasok mengenai produk bagi para pelanggan yang kemungkinan akan dipilih.

4. Order Procedure (Prosedur Pemesanan)

Order Procedure berkaitan dengan efektifitas dan efisiensi dari prosedur yang digunakan oleh pemasok.

5. Order accuracy (Akurasi Pesanan)

Order accuracy berkaitan dengan bagaimana pesanan dapat sesuai dengan keinginan pelanggan pada saat pesanan tersebut. Ini termasuk juga

memilih item yang tepat didalam pemesanan, jumlah yang sesuai dari item- itemnya, dan tidak ada penggantian produk yang telah dipesan.

6.*Order condition* (Kondisi Pesanan)

Order condition mengacu pada kurangnya kerusakan barang pada pesanan. Jika produk rusak, pelanggan tidak dapat menggunakannya dan harus mengikuti prosedur dalam pengaduannya, atau diklasifikasikan ke dalam tingkat kerusakannya.

7.*Order quality* (Kualitas Produk Pesanan)

Order quality berkaitan dengan bagaimana kualitas dari setiap produk yang ada. Kondisi ini termasuk juga bagaimana mereka yang menginformasikan tentang spesifikasi produk tersebut dan yang dibutuhkan oleh pelanggan. Sehingga tidak terjadi kemungkinan-kemungkinan yang mengakibatkan order condition tidak sesuai, *order accuracy* juga tidak sesuai bahkan sampai perusahaan yang memproduksinya juga tidak sesuai. Permasalahan ini sering terjadi apabila pelanggan menginginkan produk tertentu dari perusahaan tertentu, namun adakalanya yang datang produk yang sejenis dari perusahaan lain.

8.*Order discrepancy handling* (Penanganan Ketidakesesuaian Pesanan)

Order discrepancy handling berkaitan dengan bagaimana penanganan pengiriman produk yang dipesan dapat tiba dengan tidak ada kerusakan. Jika pelanggan pada saat menerima pesanan ditemukan tidak sesuai, kondisinya jelek, atau kualitasnya jelek, mereka dapat mengajukan

keberatan kepada pengirimnya. Kemampuan pihak pengirim yang dapat menangani setiap keluhan pengirimnya akan mampu memberikan kontribusi kepada persepsi pelanggan terhadap kualitas dan layanannya.

9. *Timeliness* (Ketepatan Waktu)

Timeliness berkaitan dengan waktu kedatangan pesanan pada tempat pelanggan ketika diharapkan. Lebih luas, *timeliness* juga mengacu pada lamanya waktu antara saat pemesanan dilakukan dan saat penerimaan pesanan. Waktu pengiriman ini dapat dipengaruhi oleh waktu yang dibutuhkan selama perjalanan, seperti halnya juga waktu pemesanan kembali ketika produk tidak tersedia.

3.9 Arduino

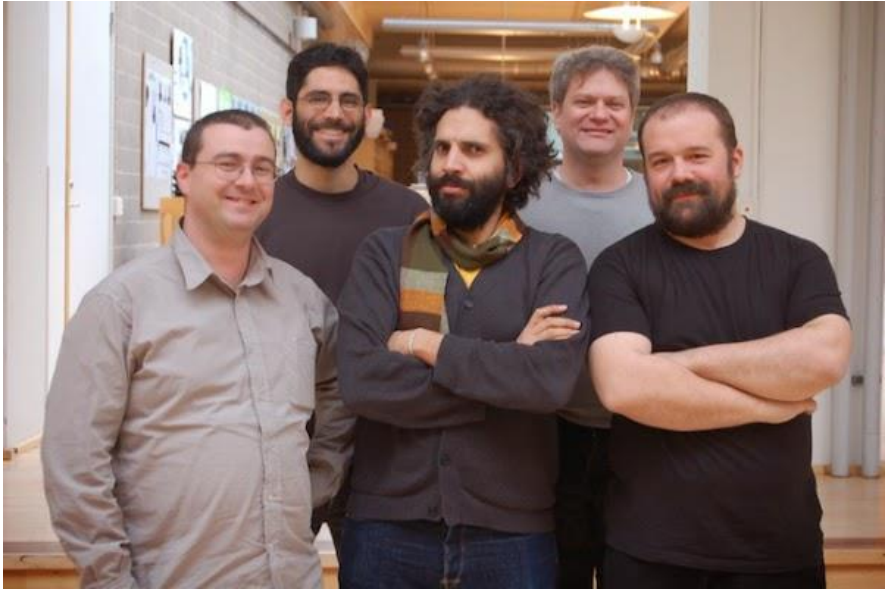


Gambar 3.19

Arduino

Arduino adalah terobosan baru dalam dunia mikrokontroller. Saat ini sudah banyak project elektronika dan robot yang berbasis Arduino[17]. Hal ini terjadi karena Arduino memiliki banyak sekali kemudahan dan mempunyai fleksibilitas yang tinggi baik dari segi *software* maupun *hardware*nya. Arduino adalah

Mikrokontroller single-board yang bersifat *open-source*, diturunkan dari *Wiring platform*, mempunyai fleksibilitas yang tinggi baik dari segi *software* maupun *hardware* untuk memudahkan Rancang bangun elektronik dalam berbagai bidang. Arduino menggunakan IC ATmega sebagai IC program dan softwarena memiliki bahasa pemrograman sendiri yang sering disebut bahasa processing. Bahasa ini sangat mirip dengan bahasa C, namun penulisannya mendekati bahasa manusia. Arduino menjadi *Platform mikrokontroller* paling populer di dunia saat ini. Kemudahan mempelajari dan mengaplikasikan arduino menjadikannya pilihan bagi pemula maupun mastah robotika dan elektronika[8]. Dan Arduino juga sudah banyak dipaka oleh perusahaan besar. Contohnya *Google* menggunakan Arduino untuk *Accessory Development Kit*, NASA memakai Arduino untuk *prototypin*, ada lagi *Large Hadron Colider* memakai Arduino dalam beberapa hal untuk pengumpulan data.



Gambar 3.19

Team Arduino

Arduino memiliki 14 pin *input/output* yang mana 6 pin dapat digunakan sebagai *output analog* (pin 3, 5, 6, 9, 10, dan 11), 6 pin *input analog* (pin 0-5), *crystal* osilator 16 MHz, koneksi USB, *jack power*, kepala ICSP, dan tombol *reset*. *Board* ini menggunakan daya yang terhubung ke komputer dengan kabel USB atau daya *external* dengan adaptor AC-DC atau baterai. Arduino mampu men-support mikrokontroller; dapat dikoneksikan dengan komputer menggunakan kabel USB. Arduino Uno adalah pilihan yang baik untuk pertama kali atau

bagi pemula yang ingin mengenal arduino. Disamping sifatnya yang reliabel juga harganya murah.

Dibawah ini adalah spesifikasi board Arduino Uno:

1. Mikrokontroler menggunakan Atmega328
2. Tegangan opsersinya adalah 5V
3. Tegangan *input* (disarankan) 7-12V
4. Batas tegangan *input* 6-20V
5. Pin *Digital* I/O ada 14 (dimana 6 pin *output* PWM)
6. Pin analog *input* ada 6 pin (pin 0- pin 5)
7. Arus DC per I/O pin adalah 40Ma
8. Arus DC untuk pin 3.3V adalah 50Ma
9. Flash memori yaitu 32 KB (Atmega328), dimana 0,5 KB digunakan oleh *bootloader*
10. SRAM-nya 2 KB (ATmega328)
11. EEPROM-nya 1 KB(Atmega328)
12. *Clock speed* 16 Mhz[18].

3.9.1 Kelebihan Arduino

Arduino memiliki berbagai kelebihan dibandingkan dengan mikrokontroller lain. Sesuai dengan 4 hal yang diupayakan pengembang arduino maka Arduino memiliki kelebihan yaitu :

1. Murah – 1 *Board* Arduino biasanya dijual relatif murah (antara 100 ribu hingga 400 ribu rupiah saja). Sekarang arduino juga banyak tersedia versi kloningan (tiruan) dengan harga yang lebih murah dari versi Originalnya yang merupakan pabrikan Italia.
2. Sederhana dan mudah pemrogramannya – Bahasa pemograman Arduino sangat Fleksibel karena hamper mendekati bahasa manusia. Tentunya sangat mudah memahami algoritma Program bagi pemula maupun tingkat lanjut.
3. *Software Open Source* – Perangkat lunak Arduino IDE berbasis *Open Source*, dan dapat dikembangkan pemrograman lebih lanjut. Bahasanya bisa dikembangkan lebih lanjut melalui pustaka-pustaka C++ yang berbasis pada Bahasa C untuk AVR.
4. *Hardware Open Source* – Perangkat keras Arduino berbasis *mikrokontroler* ATmega8, ATmega168, ATmega328 dan ATmega1280 dan ATmega 2650. Dengan demikian sangat mudah membuat dan menjual *board* Arduino. *Bootloader* Arduino juga tersedia langsung dari perangkat lunak Arduino IDE.

Dari segi *hardware* sendiri Arduino memiliki Keistimewaan Diantaranya :

1. Soket USB

Soket USB adalah soket kabel USB yang disambungkan kekomputer atau laptop. Yang berfungsi untuk mengirimkan program ke arduino dan juga sebagai *port* komunikasi serial. Jadi tidak diperlukan lagi *Downloader Eksternal* untuk mendownload Program ke IC Arduino.

2. *Input/output digital dan input analog*

Input /Output Digital dalam Arduino Fungsinya adalah membaca atau mengirim data berupa data Analog maupun Digital dari rangkaian terintegrasi dengan Arduino. Biasanya *Pin Input/ Output Analog* dan *Digital* berada pada barisan terpisah.

3. Catu Daya

Catu daya pada Arduino bisa menggunakan *Socket* USB dan terdapat pula *Socket* DC 12V untuk tipe tertentu. Di dalam *Board* Arduino sudah tersedia IC *regulator* Untuk menstabilkan dan menyuplay tegangan ke modul Arduino.

4. Ukuran Fisik

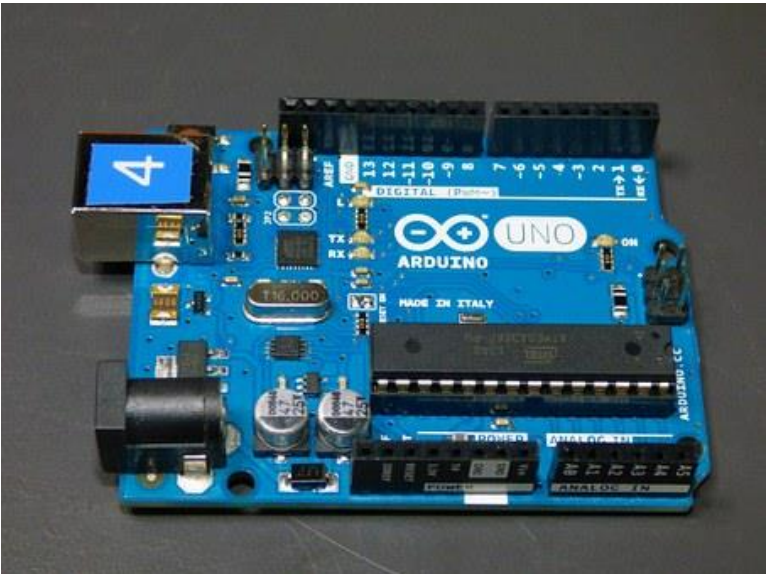
Ukuran Fisik untuk Satu Board arduino relatif kecil karena dibangun menggunakan komponen SMD (*surface mounted Device*) yaitu komponen yang sangat kecil , biasanya terdapat pada board hp dan

motherboard sehingga sangat ringkas dan Tahan terhadap berbagai situasi.

3.9.2 Jenis-jenis Arduino

1. Arduino Uno

Arduino adalah jenis yang paling banyak digunakan. Terutama untuk pemula sangat disarankan untuk menggunakan Arduino Uno karena banyak sekali referensi yang membahas Arduino Uno. Versi yang terakhir adalah Arduino Uno R3 (Revisi 3), menggunakan ATMEGA328 sebagai Microcontrollernya, memiliki 14 pin *I/O digital* dan 6 *pin input analog*. Untuk pemograman cukup menggunakan koneksi USB *type A to type B*.



Gambar 3.20

Arduino

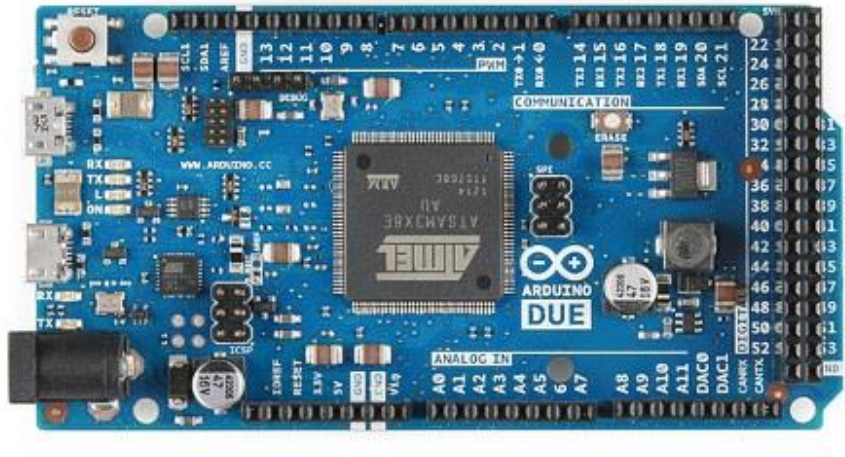
SPESIFIKASI	
Arduino Uno	
Microcontroller	ATmega328P
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limit)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
PWM Digital I/O Pins	6
Analog Input Pins	6

DC Current per I/O Pin	20 mA
DC Current for 3.3V Pin	50 mA
Flash Memory 32 KB	(ATmega328P)
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
Clock Speed	16 MHz
LED_BUILTIN	13
Length	68.6 mm
Width	53.4 mm
Weight	25 g

Tabel 3.1

Spesifikasi Arduino Uno

2. Arduino Due



Gambar 3.21

Arduino Due

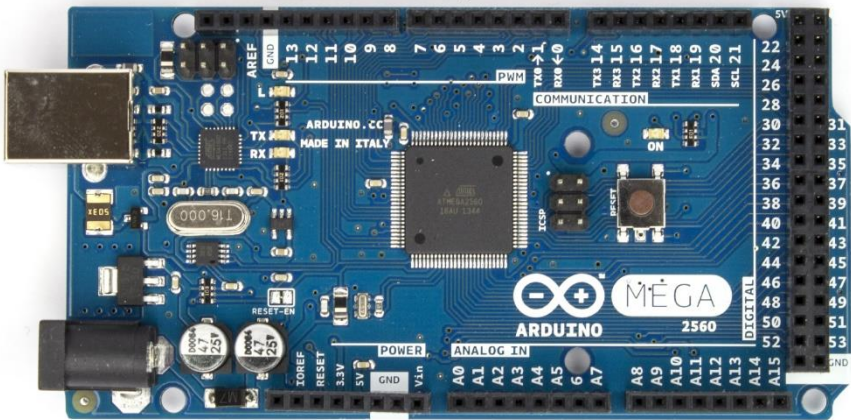
Berbeda dengan Arduino Uno, Arduino *Due* tidak menggunakan ATMEGA, melainkan dengan *chip* yang lebih tinggi ARM *Cortex* CPU. Memiliki 54 I/O pin digital dan 12 pin *input analog*. Untuk pemogramannya menggunakan *Micro* USB, terdapat pada beberapa *SmartPhone*

SPESIFIKASI	
Arduino Uno	
<i>Microcontroller</i>	ATmega328P
<i>Operating Voltage</i>	5V
<i>Input Voltage (recommended)</i>	7-12V
<i>Input Voltage (limit)</i>	6-20V
<i>Digital I/O Pins</i>	14 (<i>of which</i> 6 provide PWM output)
<i>PWM Digital I/O Pins</i>	6
<i>Analog Input Pins</i>	6
<i>DC Current per I/O Pin</i>	20 mA
<i>DC Current for 3.3V Pin</i>	50 mA
<i>Flash Memory</i> 32 KB	(ATmega328P)
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
<i>Clock Speed</i>	16 MHz
<i>LED_BUILTIN</i>	13
<i>Length</i>	68.6 mm
<i>Width</i>	53.4 mm
<i>Weight</i>	25 g

Tabel 3.2

Spesifikasi Arduino *Due*

3. Arduino *Mega*



Gambar 3.22

Arduino *Mega*

Arduino *Mega* 2560 adalah *Board* pengembangan *mikrokontroller* yang berbasis Arduino dengan menggunakan *chip* ATmega2560. *Board* ini memiliki *pin* I/O yang cukup banyak, sejumlah 54 buah *digital* I/O *pin* (15 *pin* diantaranya adalah PWM), 16 *pin* *analog input*, 4 *pin* UART (*serial port hardware*). Arduino *Mega* 2560 dilengkapi dengan sebuah *oscillator*

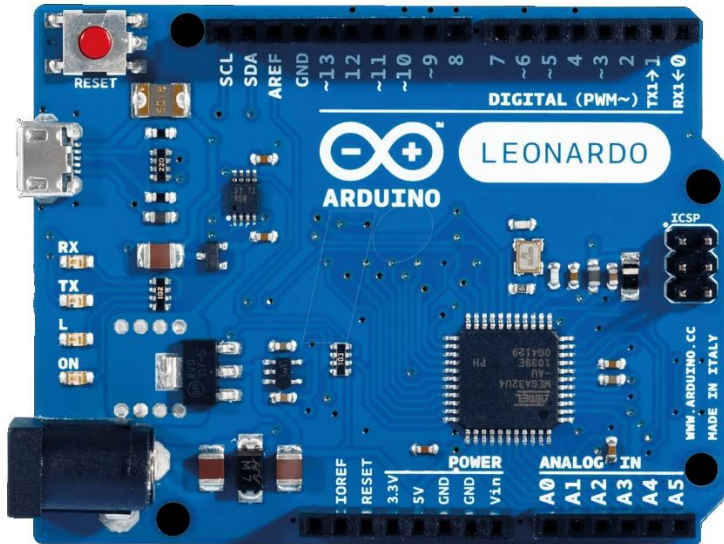
16 Mhz, sebuah *port* USB, *power jack* DC, ICSP *header*, dan tombol *reset*.
Tentunya versi Arduino mega lebih memberikan peluang dalam Rancang bangun sistem yang lebih besar.

SPESIFIKASI	
Arduino Mega	
<i>Microcontroller</i>	ATmega2560
<i>Operating Voltage</i>	5V
<i>Input Voltage (recommended)</i>	7-12V
<i>Input Voltage (limit)</i>	6-20V
<i>Digital I/O Pins</i>	54 (of which 15 <i>provide</i> PWM <i>output</i>)
<i>Analog Input Pins</i>	16
DC <i>Current per</i> I/O Pin	20 mA
DC <i>Current for</i> 3.3V Pin	50 mA
<i>Flash Memory</i>	256 KB <i>of which</i> 8 KB <i>used by</i> <i>bootloader</i>
SRAM	8 KB
EEPROM	4 KB
<i>Clock Speed</i>	16 MHz
<i>LED_BUILTIN</i>	13
<i>Length</i>	101.52 mm
<i>Width</i>	53.3 mm
<i>Weight</i>	37 g

Tabel 3.3

Spesifikasi Arduino *Mega*

4. Arduino Leonardo



Gambar 3.23

Arduino Leonardo

Arduino Leonardo adalah mikrokontroler berbasis ATmega32u4. Arduino Leonardo memiliki 20 *digital pin input/output* (yang mana 7 pin dapat digunakan sebagai output PWM dan 12 pin sebagai *input analog*), 16 MHz kristal osilator, koneksi *micro USB*, jack power suplai tegangan, *header ICSP*, dan tombol *reset*. Ini semua yang diperlukan untuk mendukung *mikrokontroler*. Cukup dengan menghubungkannya ke komputer melalui kabel USB atau *power* dihubungkan dengan adaptor

AC-DC atau baterai untuk mulai mengaktifkannya. Arduino Leonardo berbeda dari Arduino yang lainnya karena ATmega32u4 secara terintegrasi (*built-in*) telah memiliki komunikasi USB, sehingga tidak lagi membutuhkan prosesor sekunder (tanpa *chip* ATmega16U2 sebagai konverter USB-*to*-serial). Hal ini memungkinkan Arduino Leonardo yang terhubung ke komputer digunakan sebagai *mouse* dan *keyboard*, selain bisa digunakan sebagai virtual (CDC) serial/COM *port*.

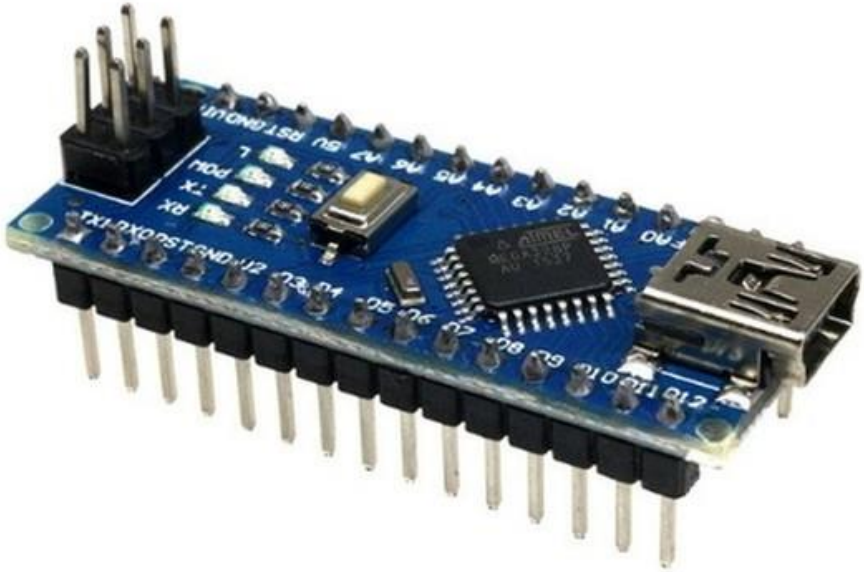
SPESIFIKASI	
Arduino Leonardo	
<i>Processor</i>	802.3 10/100 Mbit/s
<i>Microcontroller</i>	ATmega32u4
<i>Architecture</i>	AVR
<i>Operating Voltage</i>	5V
<i>Flash memory</i>	32 KB <i>of which</i> 4 KB <i>used by bootloader</i>
SRAM	2.5Kb
<i>Clock Speed</i>	16 MHz
<i>Analog I/O Pins</i>	12
EEPROM	1 KB
<i>DC Current per I/O Pins</i>	40 mA on I/O Pins; 1A on 3.3 V Pin <i>only when powered via external power supply</i>
<i>Input Voltage</i>	7-12 V
Digital I/O Pins	36-57 V
Reserved Pins	4 <i>used for</i> SD card <i>select</i> ; 10 <i>used for</i> W5500 <i>select</i>

<i>Digital I/O Pins</i>	20
<i>PWM Output</i>	7
<i>Power Consumption</i>	82 mA
<i>PCB Size</i>	53.34 x 68.58 mm
<i>Card Reader</i>	<i>Micro SD card, with active voltage translators</i>
<i>Weight</i>	28g

Tabel 3.4

Spesifikasi Arduino Leonardo

5. Arduino Nano



Gambar 3.24

Arduino Nano

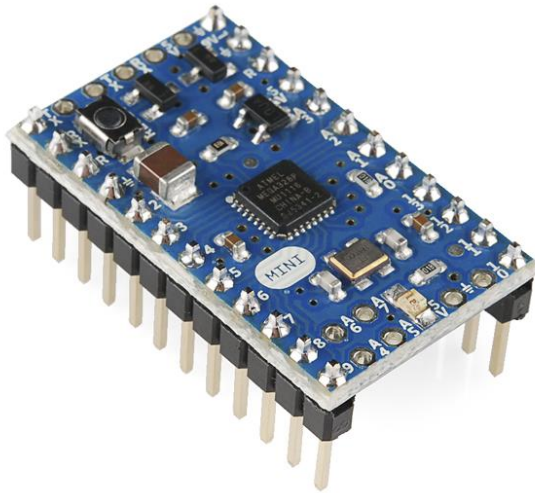
Arduino Nano adalah Versi mini dari Arduino uno. Karena bentuknya yang ringkas maka konsekuensinya adalah membuang beberapa komponen penting diantaranya *jack* DC dan *Socket* USB tipe B diganti dengan *Mikro* USB.

SPESIFIKASI	
Arduino Nano	
<i>Microcontroller</i>	ATmega328
<i>Architecture</i>	AVR
<i>Operating Voltage</i>	5 V
<i>Flash Memory</i>	32 KB of which 2 KB used by bootloader
<i>SRAM</i>	2 KB
<i>Clock Speed</i>	16 MHz
<i>Analog I/O Pins</i>	8
<i>EEPROM</i>	1 KB
<i>DC Current per I/O Pins</i>	40 mA (I/O Pins)
<i>Input Voltage</i>	7-12 V
<i>Digital I/O Pins</i>	22
<i>PWM Output</i>	6
<i>Power Consumption</i>	19 mA
<i>PCB Size</i>	18 x 45 mm
<i>Weight</i>	7 g
<i>Product Code</i>	A000005

Tabel 3.5

Spesifikasi Arduino Nano

6. Arduino Mini



Gambar 3.25

Arduino Mini

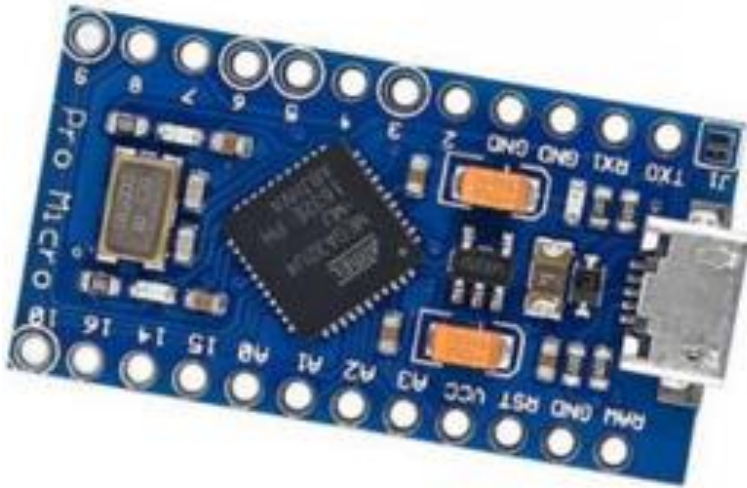
Versi Arduino mini merupakan versi yang lebih sederhana dari arduino nano. Pada *board* arduino mini tidak tersedia lagi *mikro* USB untuk *mendownload* program, artinya kita harus menggunakan *Downloader Eksternal*.

SPESIFIKASI	
Arduino Mini	
<i>Microcontroller</i>	ATmega328
<i>Operating Voltage</i>	5V
<i>Input Voltage</i>	7-9 V
<i>Digital I/O Pins</i>	14 (of which 6 provide PWM output)
<i>Analog Input Pins</i>	8 (of which 4 are broken out onto pins)
<i>DC Current per I/O Pin</i>	40 mA
<i>Flash Memory</i>	32 KB (of which 2 KB used by bootloader)
SRAM	2 KB
EEPROM	1 KB
<i>Clock Speed</i>	16 MHz
<i>Length</i>	30 mm
<i>Width</i>	18 mm

Tabel 3.6

Spesifikasi Arduino Mini

7. Arduino *Micro*



Gambar 3.26

Arduino Micro

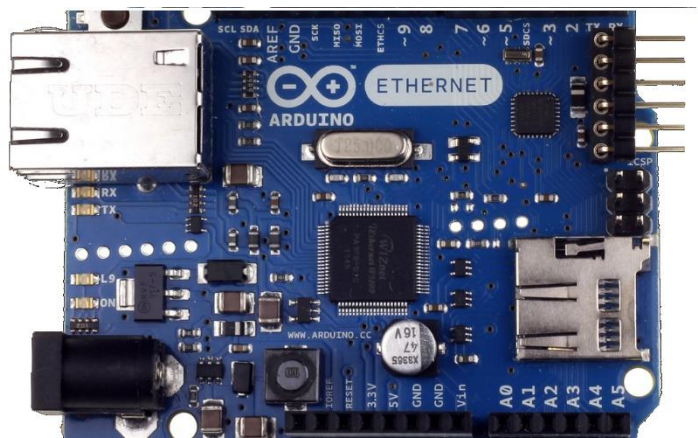
Arduino *mikro* memiliki fisik yang sama dengan Arduino nano namun lebih panjang. Perbedaannya dengan Arduino nano adalah jumlah pin yang lebih banyak, yaitu 20 pin *I/O digital* dan 12 Pin *analog*.

SPESIFIKASI	
Arduino Micro	
<i>Microcontroller</i>	ATmega32U4
<i>Operating Voltage</i>	5V
<i>Input Voltage (recommended)</i>	7-12V
<i>Input Voltage (limit)</i>	6-20V
<i>Digital I/O Pins</i>	20
<i>PWM Channels</i>	7
<i>Analog Input Channels</i>	12
<i>DC Current per I/O Pin</i>	20 mA
<i>DC Current for 3.3V Pin</i>	50 mA
<i>Flash Memory</i>	32 KB (ATmega32U4)
<i>SRAM</i>	2.5 KB (ATmega32U4)
<i>EEPROM</i>	1 KB (ATmega32U4)
<i>Clock Speed</i>	16 MHz
<i>LED_BUILTIN</i>	13
<i>Length</i>	48 mm
<i>Width</i>	18 mm

Tabel 3.7

Spesifikasi Arduino Micro

8. *Arduino Ethernet*



Gambar 3.27

Arduino Ethernet

Arduino Ethernet memungkinkan koneksi Jaringan LAN antara *Board* dengan komputer. Arduino Ethernet memiliki fungsi yang sama dengan Arduino yang lainnya.

SPESIFIKASI	
Arduino <i>Ethernet</i>	
Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage Plug (recommended)	7-12V
Input Voltage Plug	6-

<i>(limits)</i> 20V	
<i>Input Voltage PoE (limits)</i>	36-57V
<i>Digital I/O Pins)</i>	14 (of which 4 provide PWM output)
<i>Arduino Pins reserved:</i>	10 to 13 used for SPI
	4 used for SD card
	2 W5100 interrupt (when bridged)
<i>Analog Input Pins</i>	6
<i>DC Current per I/O Pin</i>	40 mA
<i>DC Current for 3.3V Pin</i>	50 mA
<i>Flash Memoryer</i>	32 KB (ATmega328) of which 0.5 KB used by bootload
<i>SRAM</i>	2 KB (ATmega328)
<i>EEPROM</i>	1 KB (ATmega328)
<i>Clock</i>	16 Mhz
<i>Length</i>	68.6 mm
<i>Width</i>	53.3 mm
<i>Weight</i>	28 g

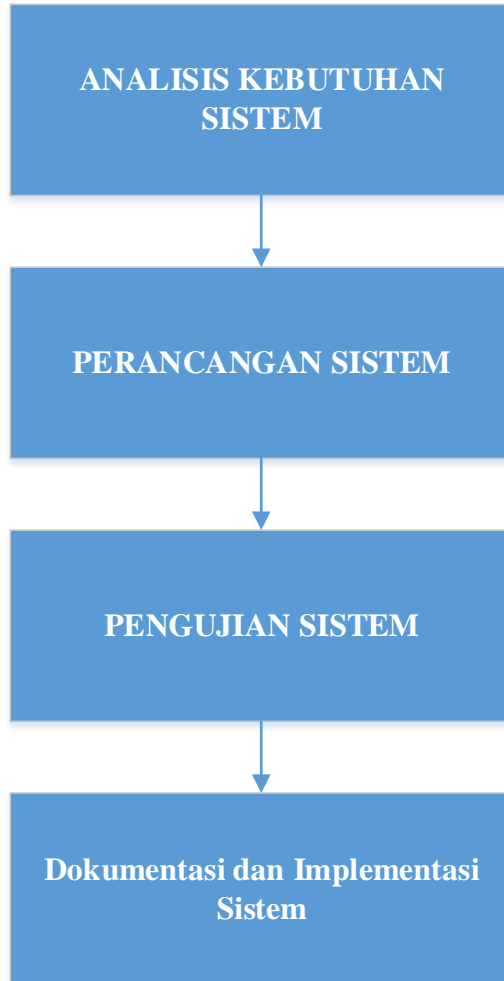
Tabel 3.8

Spesifikasi Arduino *Ethernet*

3.10 Metode Pengembangan Sistem

Metode pengembangan sistem yang digunakan adalah metode *Prototype*. Dalam pengembangan sistem pada model *prototype*

terdapat tahapan-tahapan sebagai berikut:



Gambar 3. 27

Metode Prototype.

1. Metode Analisa Kebutuhan

Metode ini melakukan analisa suatu sistem yang sudah ada, bagaimana sistem itu berjalan dan apakah kekurangan dari sistem tersebut. Pada sistem yang sekarang dalam penggunaannya masih manual, sehingga perlu adanya sistem yang dapat membantu pekerjaan di bidang industri.

2. Metode Perancangan Sistem

Dalam metode perancangan ini kita dapat mengetahui bagaimana sistem itu dibuat atau dirancang dan alat apa saja yang dibutuhkan. Melalui tahapan pembuatan *flowchart* dari sistem yang akan dibuat dan pembuatan desain aplikasi pengontrolan berupa perancangan perangkat lunak (*Software*) dan perangkat keras (*Hardware*).

3. Metode Pengujian Sistem

Pada metode pengujian ini yang dipakai adalah metode pengujian *black box*.

4. Dokumentasi dan Implementasi Sistem


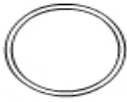


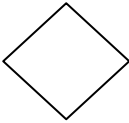
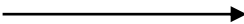
Tahap ini dilakukan untuk pengumpulan hasil kerja menjadi suatu dokumen untuk menjelaskan dari awal pembuatan sistem hingga menjadi sistem yang layak diimplementasikan[12].

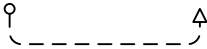

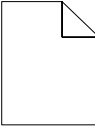
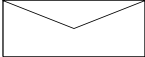
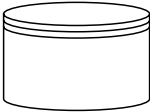

3.11 BPMN (*Business Process Model and Notation*)

BPMN (*Business Process Models And Nation*) merupakan representasi grafis untuk menentukan proses bisnis dalam model proses bisnis. BPMN merupakan kependekan dari *Business Process Model and Notation* merupakan presentasi grafis mengenai proses bisnis dalam model proses bisnisnya. BPMI yaitu *Business Process Management Initiative* inilah yang mengembangkan BPMN atau *Business Process Model and Notation*. BPMN ini kemudian dipegang oleh *Object Management Group*. *Business Process Model* ini menggunakan *flowchart* dalam melakukan penyajiannya. Tujuan utama dari DMPN adalah membuat standarisasi notasi dalam penyajian proses bisnis sehingga dapat dipahami oleh semua pemangku kepentingan. Implementasi praktis dari penerapan BPM adalah membuat *Business Process Diagram*. Baik *developer* aplikasi ERP maupun pengguna aplikasi ERP sangat disarankan membuat dokumentasi *Business Process Diagram* agar manajemen perusahaan dapat tertata rapi karena informasi SOP (*Standard Operating Procedure*) yang jelas[8].

Tujuan dari *Business Process Modelling Notation* (BPMN) adalah untuk mendukung manajemen proses bisnis, baik untuk pengguna teknis dan pengguna bisnis, dengan menyediakan notasi yang intuitif untuk pengguna bisnis, namun dapat mewakili semantik proses yang

kompleks. Berikut simbol-simbol *Business Process Modelling Notation* (BPMN) pada tabel 3.9

No	Simbol	Keterangan
1		<i>Start Event</i>
2		<i>Intermediate event</i>
3		<i>End Evend</i>
4		<i>Task/Activity</i>
5		<i>Gateway</i>
6		<i>Sequence Flow</i>

7		<i>Message Flow</i>
8		<i>Group</i>
9		<i>Data Object</i>
10		<i>Message</i>
11		<i>Data Store</i>
12		<i>Pool/Line</i>

Tabel 3.9

Simbol BPMN

3.11.1 Pentingnya *Business Process Modelling Notation* (BPMN)

- *Business Process Modelling Notation* (BPMN) adalah standar proses pemodelan diterima secara internasional.
- *Business Process Modelling Notation* (BPMN) adalah suatu metodologi pemodelan proses.
- *Business Process Modelling Notation* (BPMN) menciptakan jembatan standar yang mengurangi kesenjangan antara proses bisnis dan pelaksanaannya.
- *Business Process Modelling Notation* (BPMN) memungkinkan Anda untuk proses model dalam cara bersatu dan standar sehingga setiap orang dalam organisasi dapat saling memahami.

3.11.2 Notasi *Business Process Modelling Notation* (BPMN)

Dalam dasar kategori elemen, variasi tambahan dan informasi dapat ditambahkan untuk mendukung kebutuhan untuk kompleksitas tanpa mengubah tampilan dasar diagram. Lima (5) kategori dasar elemen *Business Process Modelling Notation* (BPMN) adalah:

- *Flow Objects*
- Data

- *Connecting Objects*
- *Swimlanes*
- *Artifacts*

1. *Flow Object* adalah elemen grafis utama untuk menentukan perilaku dalam Proses Bisnis.

Ada tiga (3) *Flow Object*:

- Events
- Activities
- Gateways

2. Data direpresentasikan dengan empat (4) elemen :

- *Data Objects*
- *Data Inputs*
- *Data Outputs*
- *Data Stores*

3. *Connecting Objects*.

Ada empat (4) cara menghubungkan Obyek Arus informasi satu sama lain atau lainnya. Ada empat (4) *Connecting Objects*:

- *Sequence Flows*
- *Message Flows*
- *Associations*
- *Data Associations*

4. Swimlines. Ada dua (2) cara pengelompokan unsur-unsur pemodelan utama melalui "*Swimlanes*:"

- *Pools*
- *Lanes*

5. *Artifacts* digunakan untuk memberikan informasi tambahan tentang Proses. Ada dua (2) Artefak standar, tapi pemodel atau alat pemodelan bebas untuk menambahkan sebanyak Artefak yang diperlukan.

- *Group*
- *Text Annotation*

3.11.3 Dasar (*Business Process Modelling Notation*) BPMN

1. *Event*

Event adalah sesuatu yang "terjadi" selama proses bisnis. *Event* ini mempengaruhi aliran Proses dan biasanya memiliki pemicu atau hasil *Event* ini dapat memulai (*Start*), Interupsi (*Interrupt*), atau mengakhiri

aliran.

Event terdapat beberapa bagian yaitu:

1. *Start event*

Arti: mengindikasikan proses dimulai.

Letak : di awal proses.

2. *Intermediate event*

- *Event* yang melekat pada batas (*boundary*) aktivitas menunjukkan bahwa aktivitas tersebut harus di-interupsi ketika Event ini dipicu.
- *Event-event* tersebut dapat melekat pada *Task* atau *Sub-process*
- *Event-event* tersebut digunakan untuk menangani *Error Handling, exception handling*, dan kompensasi.
Arti : simbol ini akan mempengaruhi proses.
Letak : di antara *start* dan *end event*.

3. *End Event*

- *End Events* menunjukkan dimana proses akan berakhir
- Ada yang berbeda "Hasil" yang menunjukkan keadaan tertentu akhir Proses

- *None Start Events* digunakan untuk menandai awal Sub-Proses atau ketika start tidak terdefinisi/ tertentu
- *Link End Event* akan diganti dalam versi BPMN berikutnya (mungkin dengan Signal)

2. Activity

Activities digunakan untuk mewakili berbagai makna dalam kehidupan sehari-hari. Aktivitas dianggap mencakup berbagai kegiatan yang dapat diselesaikan dalam waktu 5 menit, satu minggu atau lebih.

3. Gateway

Gateway adalah pemodelan elemen yang digunakan untuk mengontrol bagaimana *Sequence* Arus berinteraksi saat mereka berkumpul (*Converge*) dan menyimpang (*diverge*) dalam Proses . *Gateway* dapat mendefinisikan semua tindakan Arus Urutan Proses Bisnis. Sebuah *Gateway* kadang-kadang memainkan salah satu dari dua peran, dan kadang-kadang bermain baik pada waktu yang sama.

- Semua jenis *Gateways* bentuknya adalah belah ketupat #penanda internal didalamnya menunjukkan jenis perilaku yang berbeda-beda

- Semua *Gateways* dapat men-split dan menggabungkan aliran
- Jika aliran tidak perlu dikontrol, maka *Gateway* tidak diperlukan. Jadi, *gateway* merupakan tempat di mana kontrol diperlukan *Exclusive Gateways*
- *Gateways Eksklusif (Decision)* adalah lokasi dalam suatu proses bisnis di mana *Arus Sequence* bisa mengambil dua atau lebih jalur alternatif. Hal ini pada dasarnya adalah "pertigaan jalan" untuk proses.
- Hanya salah satu jalan keluar mungkin dapat diambil ketika Proses dilakukan
- Ada dua jenis mekanisme keputusan (*decision*) :
 - Data (misalnya, ekspresi kondisi)
 - Kejadian (*Event*) (misalnya, penerimaan pesan alternatif)
- *Gateway* tersebut juga digunakan untuk menggabungkan *Sequence Flow*
 - Perilaku penggabungan mungkin berubah dalam versi *Business Process Modelling Notation (BPMN)* berikutnya

4. *Sequence Flow*

Suatu model sequence diagram adalah suatu pandangan yang dinamis

menyangkut interaksi antar unsurunsur model pada *runtime*. *Sequence Diagram* biasanya digunakan sebagai model yang menjelaskan skenario kasus penggunaan. Dengan penciptaan suatu diagram urutan dengan *object* dan aktor dilibatkan di dalam kasus penggunaan, kamu dapat model urutan langkahlangkah pemakai dan sistem melakukan untuk melengkapi tugas yang diperlukan itu. Suatu Diagram urutan adalah sering dipasang secara langsung di bawah suatu kasus penggunaan yang ditunjuk. Hal ini menyimpan unsur bersama-sama, baik dalam model dan ketika dokumentasi diproduksi. untuk mengerjakan ini, klik kanan kasus penggunaan pada diagram dan memilih *add sequence*. *Sequence Flow* digunakan untuk menunjukkan urutan kegiatan yang akan dilakukan dalam Proses 2. Sumber dan target harus menjadi salah satu objek berikut: *Events*, *Activities*, dan *Gateways* 3 Sebuah *Sequence Flow* tidak dapat menyeberangi batas Sub-Proses atau batas *Pool*.

5. *Message Flow*

Message Flow Digunakan untuk menunjukan aliran Pesan antara dua pelaku yang telah dipersiapkan untuk mengirim dan menerima mereka. Dalam *Business Process Modelling Notation* (BPMN), dua *Pools* terpisah dalam Diagram Kolaborasi akan mewakili dua

peserta (misal: *partner entitas* atau *partner roles*)

6. *Pool/Line*

Pool Adalah representasi grafis dari pelaku/peserta kolaborasi. Hal ini juga bertindak sebagai "*swimlane*" dan wadah grafis untuk partisi satu set kegiatan dari *Pools* lain, biasanya dalam konteks situasi B2B. *Pool* A mungkin memiliki *internal* yang rinci, dalam bentuk proses yang akan dieksekusi. *Lane* adalah partisi sub-dalam Proses, terkadang dalam *Pool*, akan memperpanjang seluruh Proses baik secara vertikal ataupun horisontal. Jalur yang digunakan untuk mengatur dan mengkategorikan Kegiatan.

7. *Data Object*

Data Object memberikan informasi tentang kegiatan apa yang perlu diadakandan atau apa yang mereka hasilkan. *Data Object* dapat mewakili benda tunggal atau koleksi benda-benda. data *input* dan data *Output* memberikan informasi yang sama untuk Proses.

8. *Group*

Group Adalah pengelompokan unsur-unsur grafis yang berada dalam kategori yang sama. Jenis pengelompokan tidak mempengaruhi *Sequence Flow* dalam *Group*. Nama Kategori muncul pada diagram sebagai label kelompok. Kategori dapat digunakan untuk

dokumentasi atau analisis tujuan. Group adalah salah satu cara dimana kategori benda dapat secara visual ditampilkan pada Diagram

Kelebihan menggunakan notasi *Business Process Modelling Notation* (BPMN) adalah:

1. Proses dan cara kerja hampir sama dengan yang sudah dipelajari yaitu *Flowmap/flowchart*
2. terdapat simulasi pada pengujian proses bisnis
3. Validasi *error* apabila ada relasi yang tidak nyambung
4. Dapat menggambarkan kesuruhan proses dalam satu diagram sederhana sehingga representasi proses bisnis relatif lebih cepat dipahami.
5. Mampu memodelkan aliran pesan
6. Mampu memodelkan aliran proses secara sekuensial dari kejadian awal sampai hasil akhir.

Kekurangan dari menggunakan notasi *Business Process Modelling Notation* (BPMN) adalah

1. Simbol-simbol pada *Business Process Modelling Notation* (BPMN) terlalu *complicated* untuk diimplementasikan pada *real* transaksi di *industry*.
2. *Business Process Modelling Notation* (BPMN) tidak bisa menggambarkan hasil dari proses dan model resiko, sehingga *Key Performance Indicator* (KPI) tidak bisa digambarkan menggunakan notasi *Business Process Modelling Notation* (BPMN).
3. Tidak bisa menggambarkan conceptual modeling, business logic dan detail dari aktivitas

2.1

2.2


2.3

2.4

2.5 Perbedaan *Python 2.x* dan *Python 3.x*

Banyak perbedaan yang akan kita temui jika kita dahulu pernah menggunakan *python* versi 2.x cukup lama sehingga berpindah ke versi 3.x, berikut contoh perbedaan pada *python* versi 2.x dan 3.x yang sangat penting untuk diketahui:

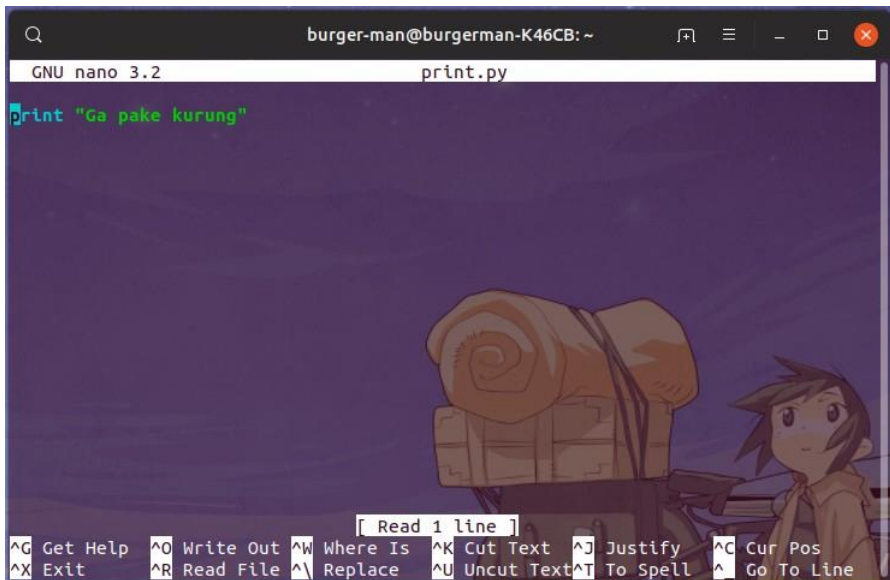
1. Perintah **print** Perbedaan perintah *print* pada dua versi ini adalah python 2.x tidak memakai kurung dan 3.x memakai kurung untuk perintah *print* bisa dilihat pada gambar 2.1 dan 2.2



```
Q burger-man@burgerman-K46CB: ~
(base) burger-man@burgerman-K46CB:~$ python2 print.py
Ga pake kurung
(base) burger-man@burgerman-K46CB:~$ python3 print.py
File "print.py", line 1
  print "Ga pake kurung"
    ^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print("Ga pake kurung")?
(base) burger-man@burgerman-K46CB:~$
```

The image shows a terminal window with a dark background and a cartoon illustration of a character on a motorcycle at the bottom. The terminal output demonstrates that Python 2 uses the `print` statement without parentheses, while Python 3 requires parentheses, resulting in a `SyntaxError` for the Python 3 command.

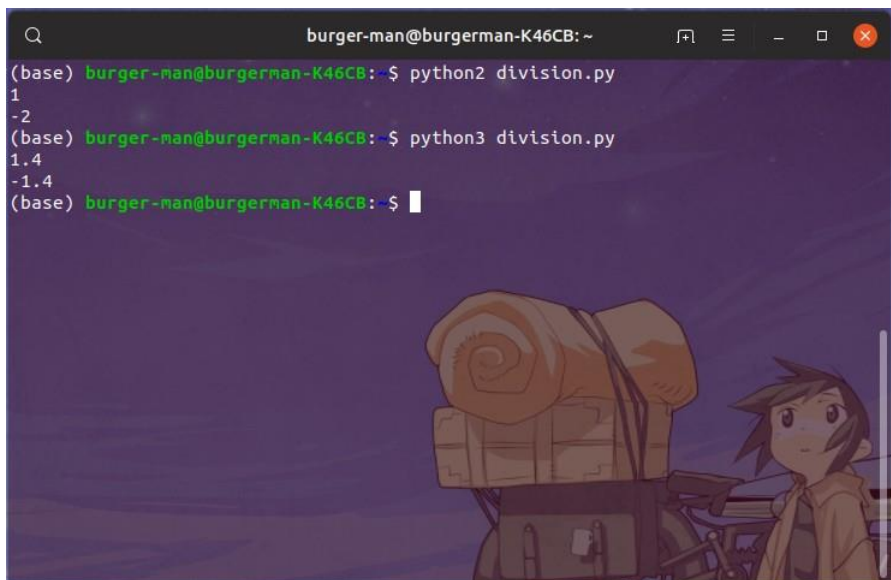
Gambar 2.1 Gambar hasil print



Gambar 2.2 Gambar perintah print

2. Perintah pembagian *integer* Hasil dari perintah pembagian cukup jelas berbeda

yang mana versi 2.x tidak secara mendetail untuk hasilnya sehingga angka yang dihasilkan bilangan *integer* sedangkan versi 3.x bertipe *float* perbedaannya bisa dilihat pada gambar 2.3 dan 2.4.

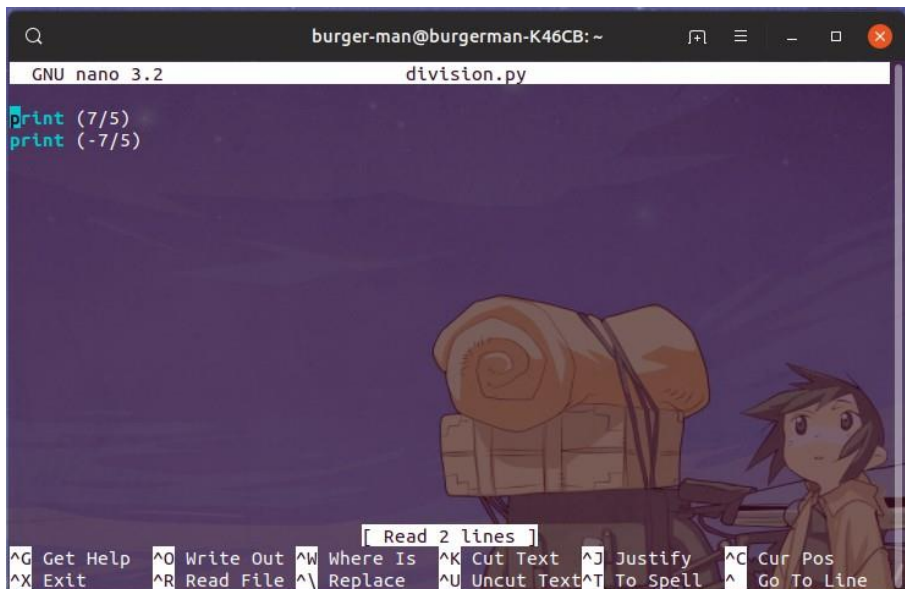


```

burger-man@burgerman-K46CB: ~
(base) burger-man@burgerman-K46CB:~$ python2 division.py
1
-2
(base) burger-man@burgerman-K46CB:~$ python3 division.py
1.4
-1.4
(base) burger-man@burgerman-K46CB:~$

```

Gambar 2.3 Gambar hasil pembagian



```

GNU nano 3.2      division.py

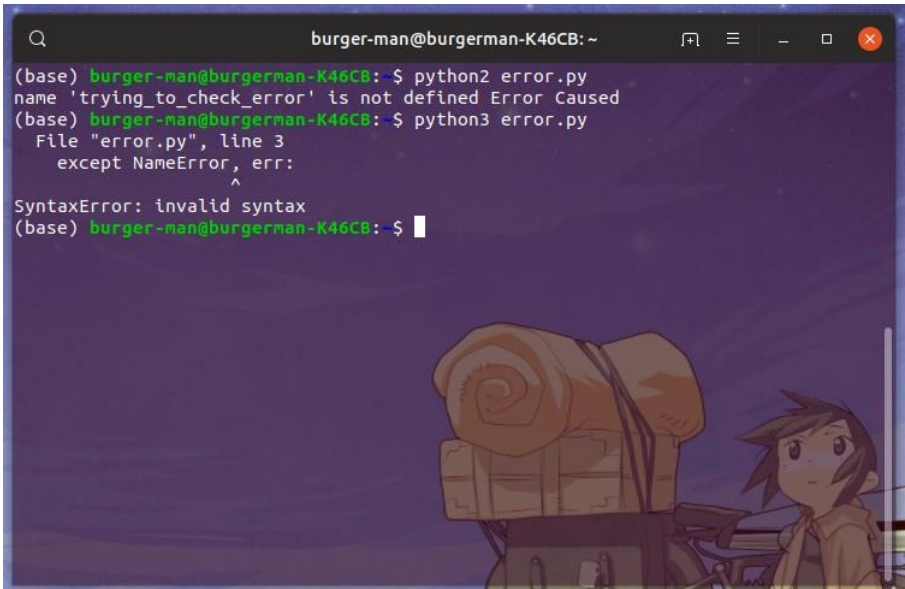
print (7/5)
print (-7/5)

```

Read 2 lines

[^]G Get Help [^]O Write Out [^]W Where Is [^]K Cut Text [^]J Justify [^]C Cur Pos
[^]X Exit [^]R Read File [^]\ Replace [^]U Uncut Text [^]T To Spell [^]_ Go To Line

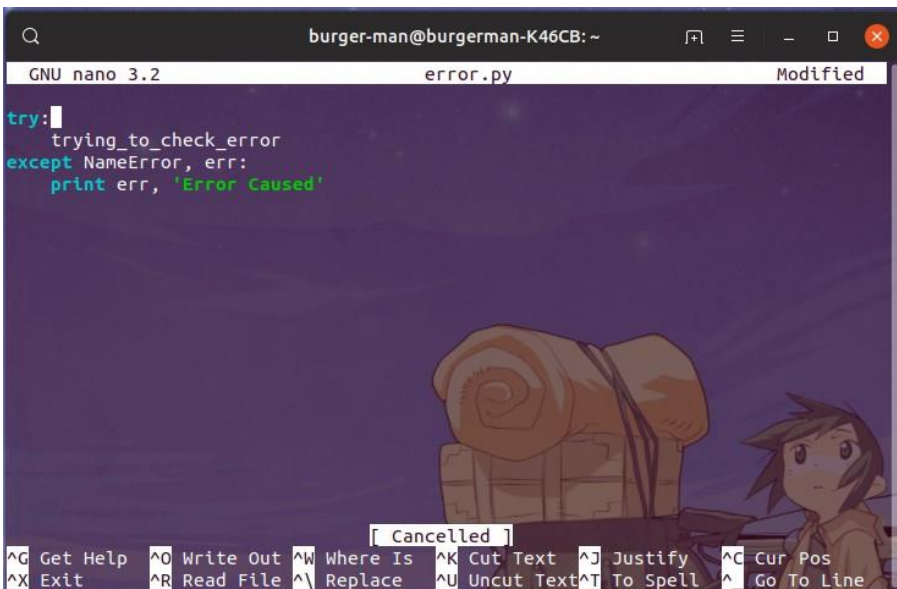
3. **Try and Except** Perbedaan pada *try and except* hanya berbeda di penggunaan , untuk versi 2.x dan **as** untuk versi 3.x.



```

burger-man@burgerman-K46CB: ~
(base) burger-man@burgerman-K46CB: $ python2 error.py
name 'trying_to_check_error' is not defined Error Caused
(base) burger-man@burgerman-K46CB: $ python3 error.py
  File "error.py", line 3
    except NameError, err:
            ^
SyntaxError: invalid syntax
(base) burger-man@burgerman-K46CB: $
  
```

Gambar 2.5 Gambar hasil error



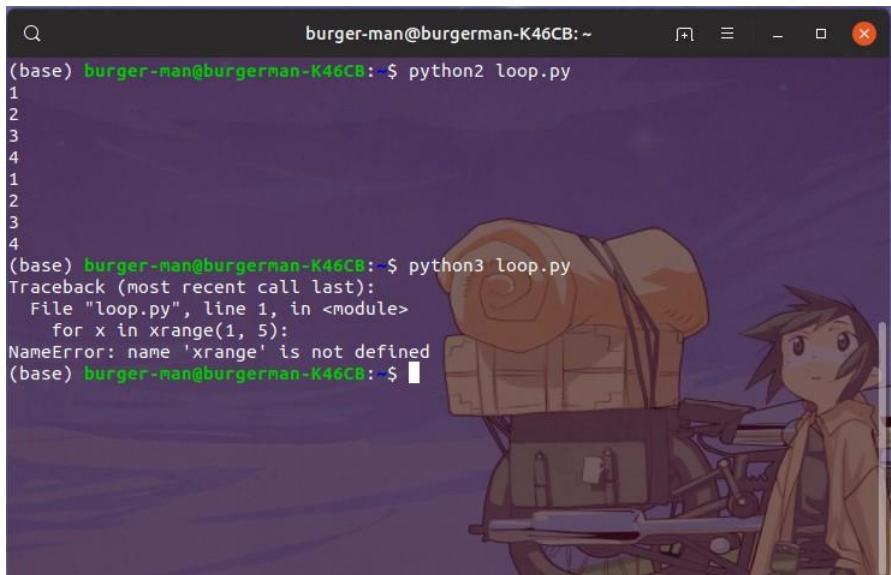
```

GNU nano 3.2                                error.py                                Modified
try:
    trying_to_check_error
except NameError, err:
    print err, 'Error Caused'
  
```

Cancelled

^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
 ^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

4. **Looping** Perbedaan pada looping hanya saja versi 3.x tidak bisa menggunakan sintaks *xrange* lagi.

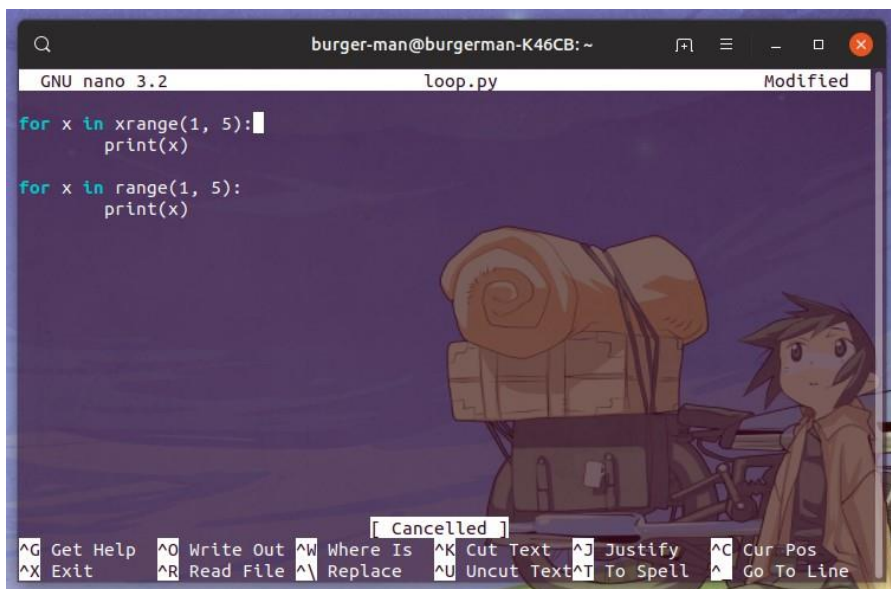


```

burger-man@burgerman-K46CB: ~
(base) burger-man@burgerman-K46CB:~$ python2 loop.py
1
2
3
4
1
2
3
4
(base) burger-man@burgerman-K46CB:~$ python3 loop.py
Traceback (most recent call last):
  File "loop.py", line 1, in <module>
    for x in xrange(1, 5):
NameError: name 'xrange' is not defined
(base) burger-man@burgerman-K46CB:~$

```

Gambar 2.7 Gambar hasil looping



```

GNU nano 3.2                                loop.py                                Modified
for x in xrange(1, 5):
    print(x)

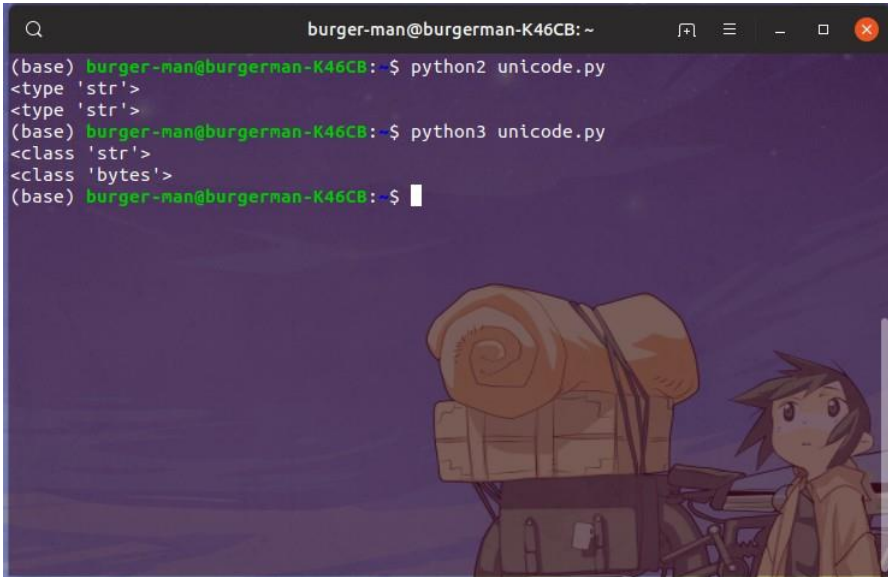
for x in range(1, 5):
    print(x)

```

Cancelled

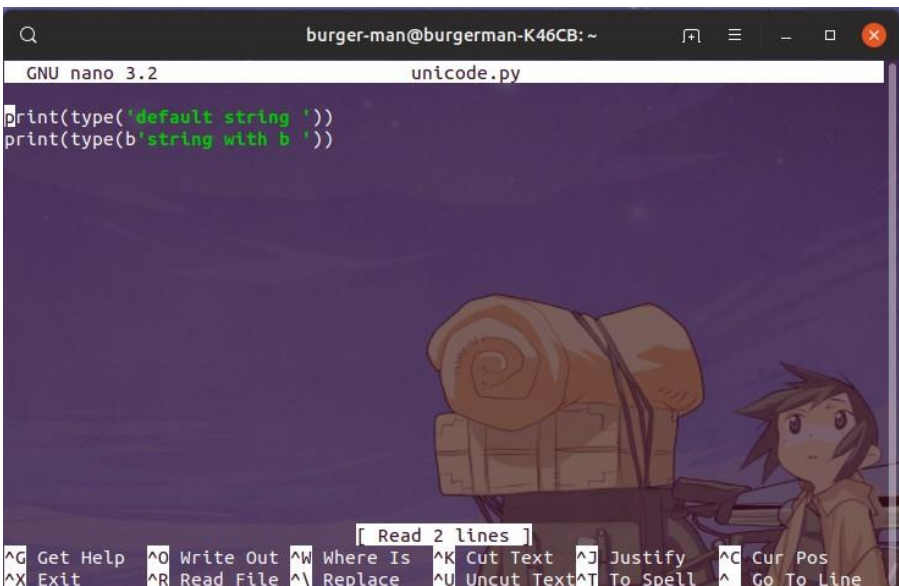
^G Get Help ^O Write Out ^W Where Is ^K Cut Text ^J Justify ^C Cur Pos
 ^X Exit ^R Read File ^\ Replace ^U Uncut Text ^T To Spell ^_ Go To Line

5. **Unicode** Unicode ini cukup penting karena kita mengetahui bagaimana setiap versi merespons setiap unicode yang diberikan.



```
burger-man@burgerman-K46CB: ~  
(base) burger-man@burgerman-K46CB: $ python2 unicode.py  
<type 'str'>  
<type 'str'>  
(base) burger-man@burgerman-K46CB: $ python3 unicode.py  
<class 'str'>  
<class 'bytes'>  
(base) burger-man@burgerman-K46CB: $
```

Gambar 2.9 Gambar hasil unicode (bytes)

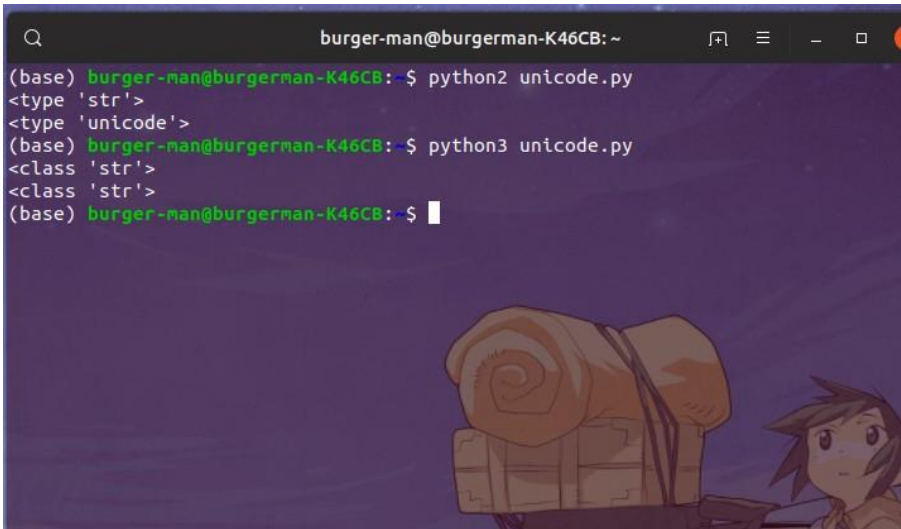


```
GNU nano 3.2 unicode.py  
print(type('default string '))  
print(type(b'string with b '))
```

Read 2 lines

^G Get Help	^O Write Out	^W Where Is	^K Cut Text	^J Justify	^C Cur Pos
^X Exit	^R Read File	^_ Replace	^U Uncut Text	^T To Spell	^_ Go To Line

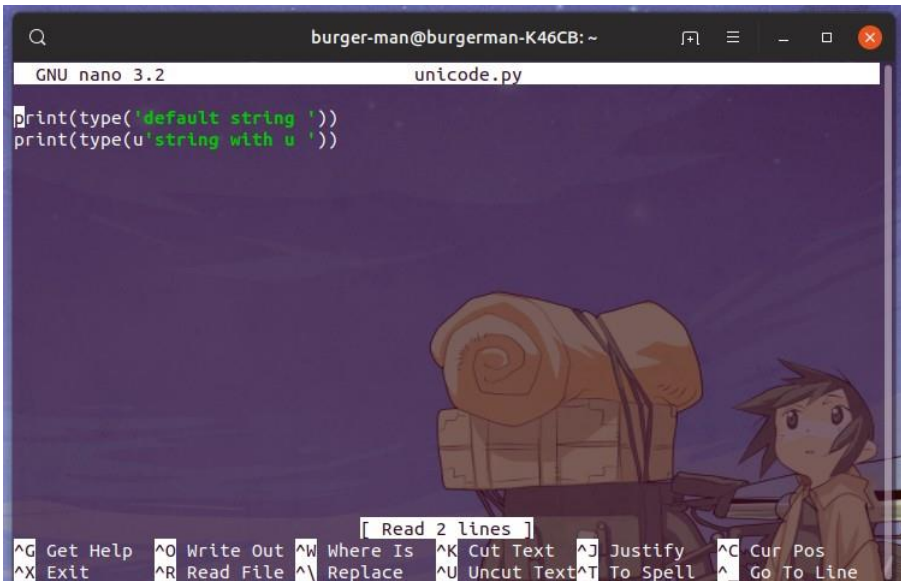
Pada gambar 2.9 terlihat jelas bahwa perintah *bytes* hanya direspon pada versi 3.x sedangkan versi 2.x merespon *string*



```

burger-man@burgerman-K46CB: ~
(base) burger-man@burgerman-K46CB: $ python2 unicode.py
<type 'str'>
<type 'unicode'>
(base) burger-man@burgerman-K46CB: $ python3 unicode.py
<class 'str'>
<class 'str'>
(base) burger-man@burgerman-K46CB: $
  
```

Gambar 2.11 Gambar hasil unicode



```

GNU nano 3.2 unicode.py
print(type('default string '))
print(type(u'string with u '))
  
```

Read 2 lines

^G Get Help	^O Write Out	^W Where Is	^K Cut Text	^J Justify	^C Cur Pos
^X Exit	^R Read File	^_ Replace	^U Uncut Text	^T To Spell	^_ Go To Line

2.6 Instalasi Python

Untuk instalasi kali ini akan bagi menjadi dua sistem operasi yaitu Windows (Windows 10), dan Linux (Ubuntu 19.04). Instalasi menggunakan *environment* Anaconda sebagai instalasi *python*. Anaconda merupakan *environment open-source* untuk bahasa pemrograman *Python*, dan *R* berfungsi untuk manajemen penggunaan *package* pada *python* dan *R*.

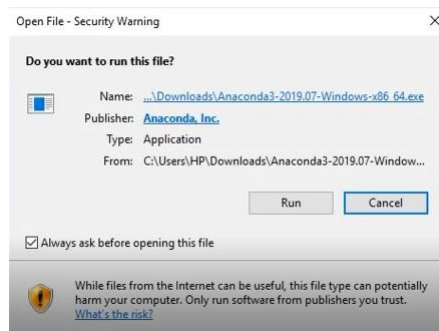
2.3.1 Windows (Windows 10)

Hal yang harus diperhatikan sebelum melakukan instalasi *Anaconda Python*

1. Perhatikan versi dari sistem operasi yang digunakan (versi 32bit atau 64bit)
2. Download file anaconda yang sesuai dengan versi sistem operasi (32bit atau 64bit)
3. *Download Anaconda Python* <https://www.anaconda.com/distribution/>

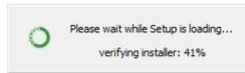
Berikut langkah-langkah instalasi anaconda.

1. Buka aplikasi *installer Anaconda* tersebut lalu akan muncul gambar *installer anaconda*.



Gambar 2.13 Run Setup Anaconda

2. Tunggu hingga *setup loading* selesai



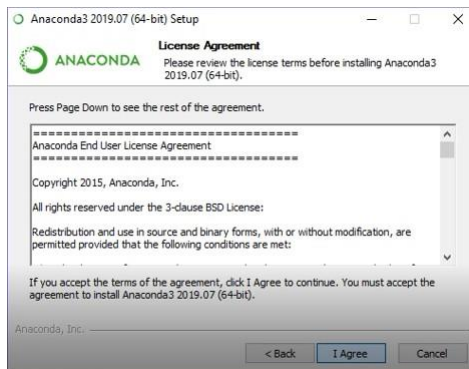
Gambar 2.14 Setup Loading

3. Jika *setup loading* telah selesai, maka klik *next*

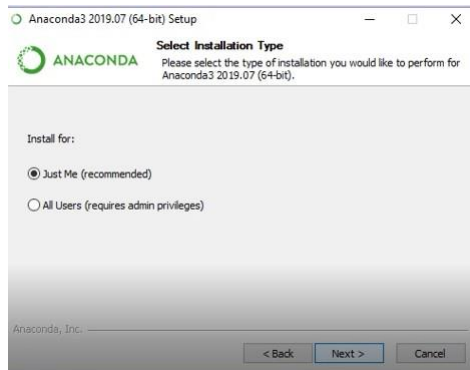


Gambar 2.15 Welcome to Anaconda Setup

4. Pada *License Agreement* klik *I Agree* gambar *License Agreement*.

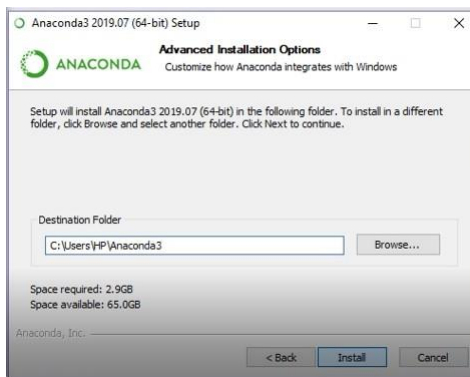


5. Kemudian pilih *Just Me(Recommended)* agar sesuai dengan komputer yang digunakan, kemudian klik *next* gambar *Just Me(recommended)*.



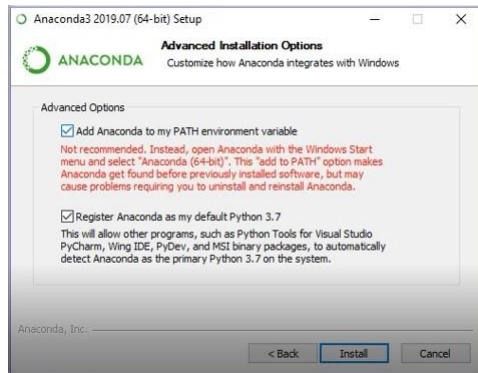
Gambar 2.17 *Just Me(recommended)*

6. Kemudian pilih lokasi tempat *install anaconda* gambar *Pilih lokasi*.



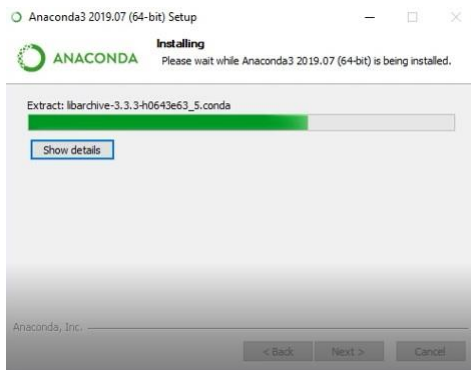
Gambar 2.18 *Pilih lokasi*

7. Kemudian centang *Add Anaconda to my Path environment variable*, agar saat *install selenium* langsung ke *path anaconda* tidak ke aplikasi yang lain. Klik *install* gambar *Centang Anaconda to my PATH*.



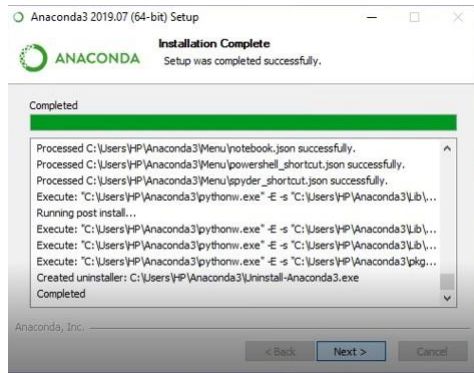
Gambar 2.19 Centang Anaconda to my PATH

8. Tunggu sampai proses *installasi* selesai gambar *Installation Complete*.



Gambar 2.20 Installation Complete

9. Apabila instalasi telah selesai klik *next*



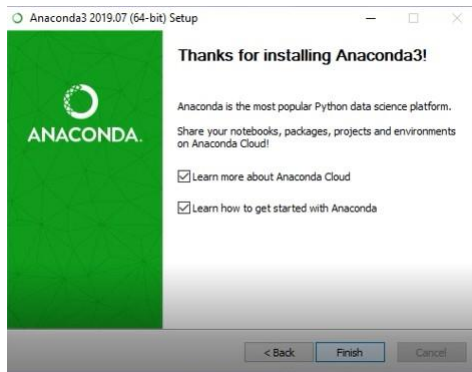
Gambar 2.21 *Installation Complete*

10. klik *next*



Gambar 2.22 *Anaconda+JetBrains*

11. Jika sudah klik *finish* gambar *Thanks fo install Anaconda*.



Gambar 2.23 Thanks for install Anaconda

2.7 Instalasi Pip

2.4.1 Windows (Windows 10)

1. buka anaconda prompt
2. ketikkan `conda install -c anaconda pip`

```
(base) C:\Users\trian>conda install -c anaconda pip
Collecting package metadata (current_repodata.json): done
Solving environment: done

==> WARNING: A newer version of conda exists. <==
current version: 4.7.10
latest version: 4.7.12

Please update conda by running

    $ conda update -n base -c defaults conda

## Package Plan ##

environment location: C:\Users\trian\Anaconda3

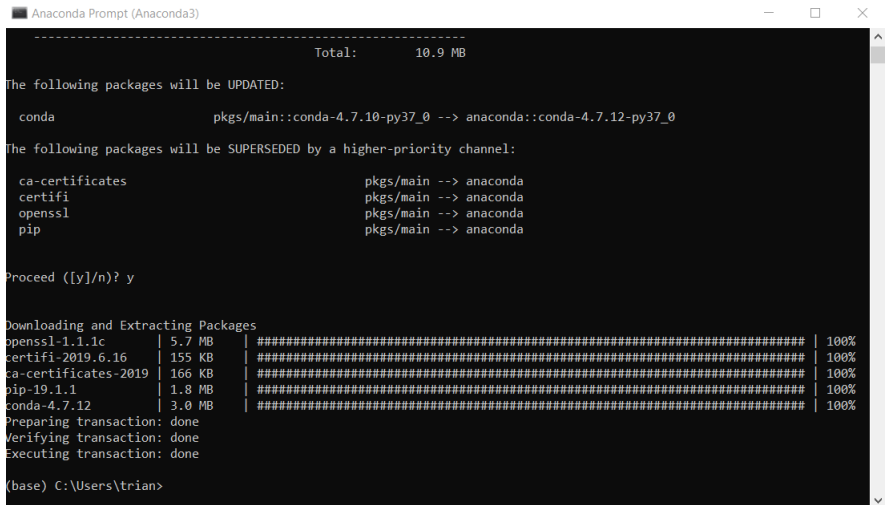
added / updated specs:
- pip

The following packages will be downloaded:
```

package	build	size	channel
ca-certificates-2019.5.15	0	166 KB	anaconda
certifi-2019.6.16	py37_0	155 KB	anaconda

Gambar 2.24 Install pip

3. ketik y, lalu enter. Tunggu hingga proses instalasi selesai.



```

Anaconda Prompt (Anaconda3)
-----
Total:      10.9 MB

The following packages will be UPDATED:

conda                pkgs/main::conda-4.7.10-py37_0 --> anaconda::conda-4.7.12-py37_0

The following packages will be SUPERSEDED by a higher-priority channel:

ca-certificates      pkgs/main --> anaconda
certifi              pkgs/main --> anaconda
openssl              pkgs/main --> anaconda
pip                  pkgs/main --> anaconda

Proceed ([y]/n)? y

Downloading and Extracting Packages
-----
openssl-1.1.1c      | 5.7 MB | ##### | 100%
certifi-2019.6.16   | 155 KB | ##### | 100%
ca-certificates-2019 | 166 KB | ##### | 100%
pip-19.1.1          | 1.8 MB | ##### | 100%
conda-4.7.12        | 3.0 MB | ##### | 100%

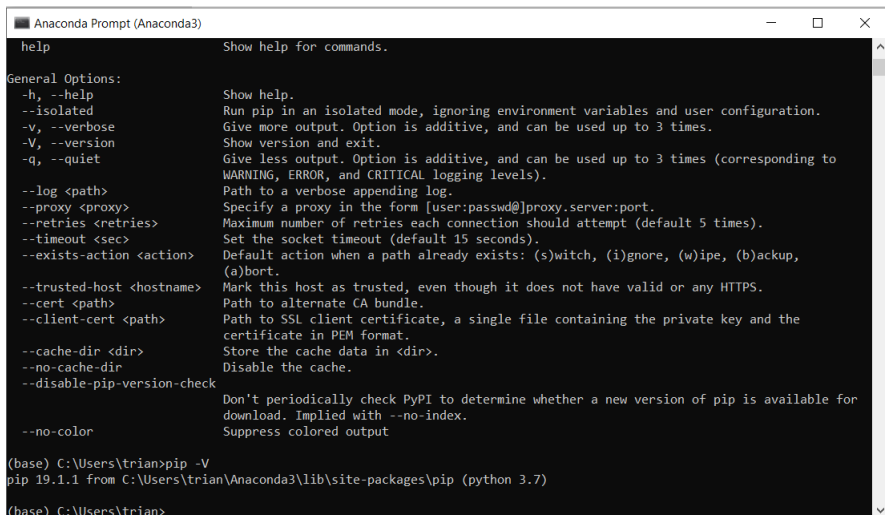
Preparing transaction: done
Verifying transaction: done
Executing transaction: done

(base) C:\Users\trian>

```

Gambar 2.25 *Install pip Selesai*

4. jika telah selesai, lakukan pengecekan versi pip dengan mengetikkan pip -V



```

Anaconda Prompt (Anaconda3)
-----
help      Show help for commands.

General Options:
-h, --help      Show help.
--isolated      Run pip in an isolated mode, ignoring environment variables and user configuration.
-v, --verbose   Give more output. Option is additive, and can be used up to 3 times.
-V, --version   Show version and exit.
-q, --quiet     Give less output. Option is additive, and can be used up to 3 times (corresponding to
                WARNING, ERROR, and CRITICAL logging levels).
--log <path>   Path to a verbose appending log.
--proxy <proxy> Specify a proxy in the form [user:passwd@]proxy.server:port.
--retries <retries> Maximum number of retries each connection should attempt (default 5 times).
--timeout <sec> Set the socket timeout (default 15 seconds).
--exists-action <action> Default action when a path already exists: (s)witch, (i)gnore, (w)ipe, (b)ackup,
                (a)bort.
--trusted-host <hostname> Mark this host as trusted, even though it does not have valid or any HTTPS.
--cert <path> Path to alternate CA bundle.
--client-cert <path> Path to SSL client certificate, a single file containing the private key and the
                certificate in PEM format.
--cache-dir <dir> Store the cache data in <dir>.
--no-cache-dir Disable the cache.
--disable-pip-version-check Don't periodically check PyPI to determine whether a new version of pip is available for
                download. Implied with --no-index.
--no-color      Suppress colored output

(base) C:\Users\trian>pip -V
pip 19.1.1 from C:\Users\trian\Anaconda3\lib\site-packages\pip (python 3.7)

(base) C:\Users\trian>

```

Gambar 2.26 *Melihat Versi pip*

2.4.2 Linux (Ubuntu 19.04)

1. pertama kita buka terminal kita lalu ketikkan perintah **sudo apt install python3-pip -y** untuk pip3 dan **sudo apt install python-pip -y** untuk pip contoh seperti gambar 2.27, lalu enter

```

burger-man@burgerman-K46CB: ~
(base) burger-man@burgerman-K46CB: $ sudo apt install python3-pip -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
python3-pip is already the newest version (18.1-5).
0 upgraded, 0 newly installed, 0 to remove and 0 not upgraded.
(base) burger-man@burgerman-K46CB: $

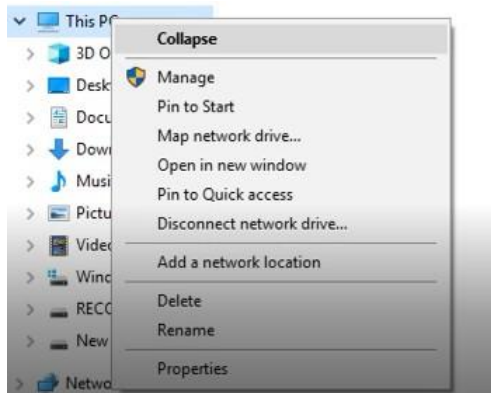
```

Gambar 2.27 Gambar instal pip

2.8 Setting Environment

2.5.1 Windows (Windows 10)

1. Buka file explorer
2. Klik kanan pada This pc, lalu pilih properties

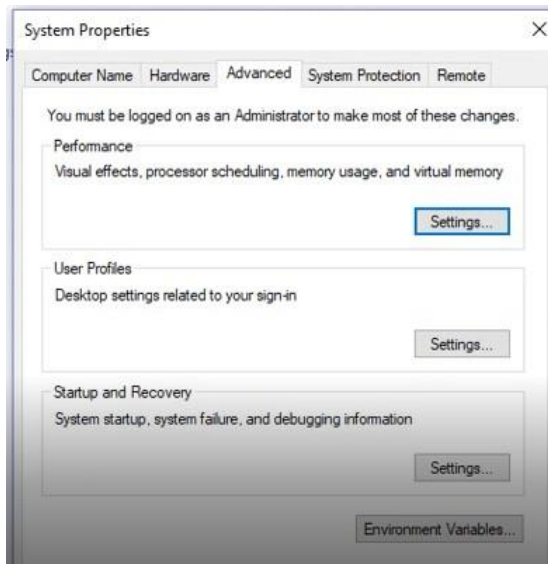


3. Pilih menu Advanced system settings



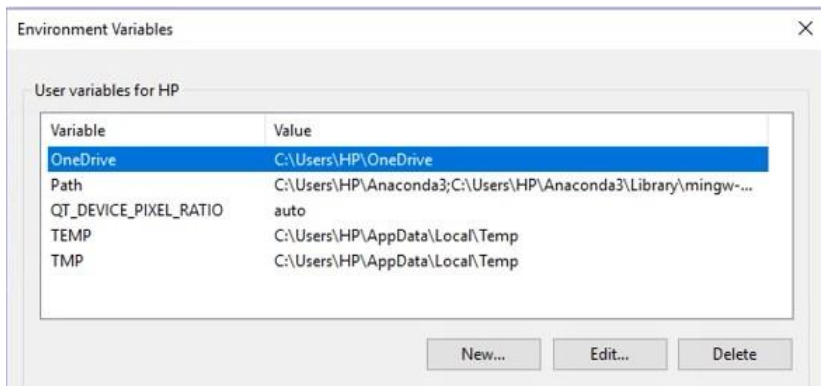
Gambar 2.29 *Advanced system settings*

4. Pilih Environment Variables



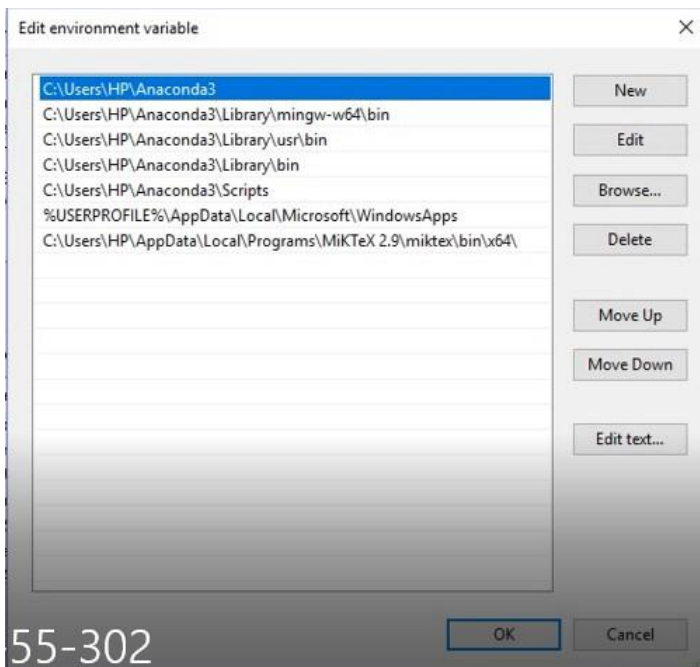
Gambar 2.30 *Environment Variables*

5. Pilih Path



Gambar 2.31 Path

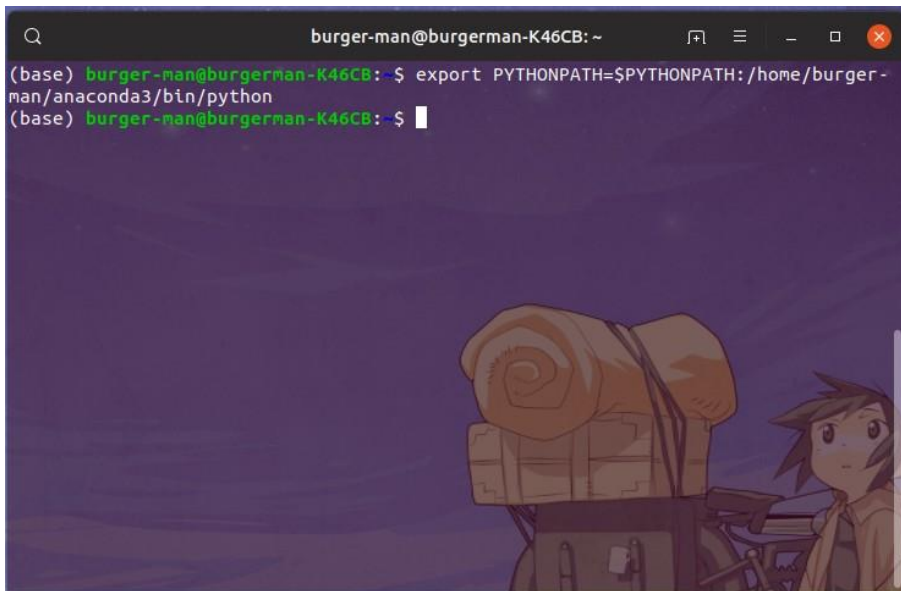
6. lalu pilih environment variable yang ingin ditambahkan, klik OK



Gambar 2.32 Edit Environment Variable

2.5.2 Linux (Ubuntu 19.04)

1. pertama kita buka terminal kita lalu ketikkan perintah `export PYTHONPATH=$PYTHON` contoh seperti gambar 2.33, lalu enter

A screenshot of a terminal window titled 'burger-man@burgerman-K46CB: ~'. The terminal shows a command prompt '(base) burger-man@burgerman-K46CB: \$' followed by the command 'export PYTHONPATH=\$PYTHONPATH:/home/burger-man/anaconda3/bin/python'. The command is executed, and the prompt returns to '(base) burger-man@burgerman-K46CB: \$'. The background of the terminal window features a cartoon illustration of a character with spiky hair and a backpack, standing next to a large, rolled-up object, possibly a rug or a large scroll, against a dark, textured background.

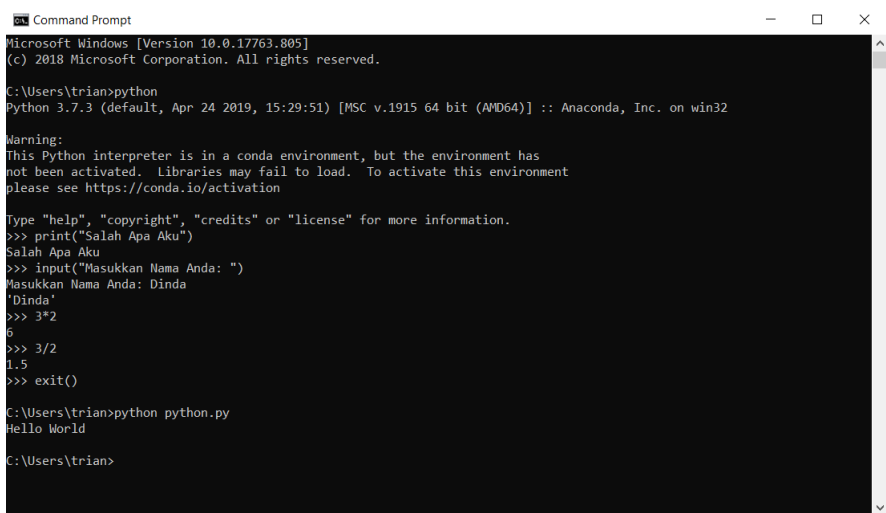
Gambar 2.33 Gambar setpath

2.9 Command Line Interface/Interpreter

2.6.1 Windows (Windows 10)

1. Buka command prompt lalu ketikkan python
2. Buatlah perintah print, input, perkalian, dan pembagian

3. Bisa juga menjalankan file .py yang telah dibuat di IDE dengan cara python namafile.py, lalu klik enter



```

Microsoft Windows [Version 10.0.17763.805]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\trian>python
Python 3.7.3 (default, Apr 24 2019, 15:29:51) [MSC v.1915 64 bit (AMD64)] :: Anaconda, Inc. on win32

Warning:
This Python interpreter is in a conda environment, but the environment has
not been activated. Libraries may fail to load. To activate this environment
please see https://conda.io/activation

Type "help", "copyright", "credits" or "license" for more information.
>>> print("Salah Apa Aku")
Salah Apa Aku
>>> input("Masukkan Nama Anda: ")
Masukkan Nama Anda: Dinda
'Dinda'
>>> 3*2
6
>>> 3/2
1.5
>>> exit()

C:\Users\trian>python python.py
Hello World

C:\Users\trian>

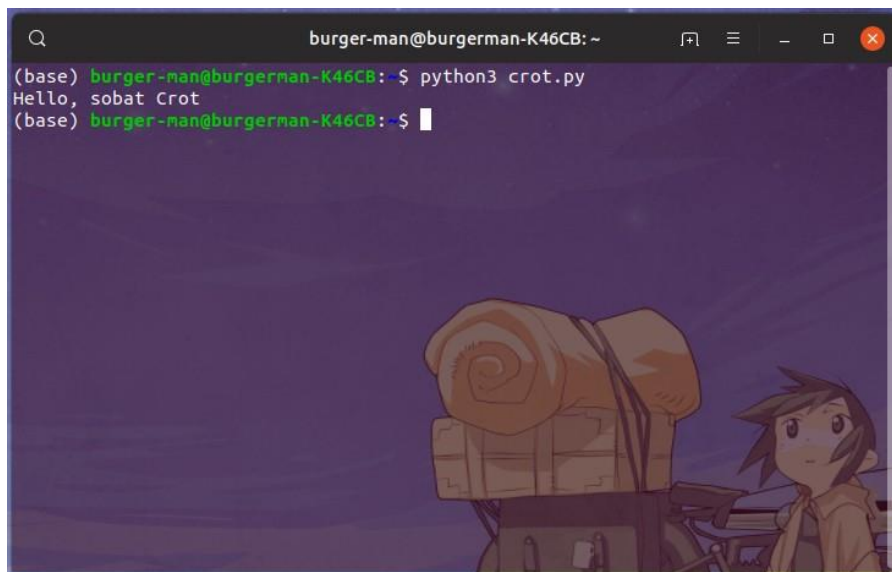
```

Gambar 2.34 CLI in Command Prompt

2.6.2 Linux (Ubuntu 19.04)

Untuk menjalankan perintah CLI cukup mudah yaitu sebagai berikut

1. Buka terminal lalu ketikkan **python namafile.py** seperti gambar 2.35, lalu enter



```

burger-man@burgerman-K46CB: ~
(base) burger-man@burgerman-K46CB:~$ python3 crot.py
Hello, sobat Crot
(base) burger-man@burgerman-K46CB:~$

```


BAB 3

JUDUL BAGIAN KEDUA

3.1 Variabel

Variabel adalah sebuah tempat untuk menampung value dimemori, dapat dimisalkan seperti sebuah ruangan atau wadah, variabel dibagi dua berdasarkan ruang lingkup yaitu variabel lokal dan global, untuk menentukan variabel global atau lokal itu tergantung dari tempat dideklarasikannya variabel pada program yang sedang dibuat. Variabel global yaitu variabel yang dapat diakses di semua lingkup dalam program yang sedang dibuat, dalam kata lain variabel global ini dapat dikenali oleh semua fungsi dan prosedur, sementara variabel lokal yaitu variabel yang dapat diakses hanya di lingkup khusus, dalam kata lain variabel lokal ini hanya bisa diakses pada fungsi/prosedur dimana variabel itu dideklarasikan.

Berikut merupakan standar-standar dalam penulisan variabel:

1. Nama variabel diawali dengan huruf atau garis bawah, contoh: nama, nama, namaKu, nama_variabel.

2. Karakter selanjutnya dapat berupa huruf, garis bawah atau angka, contoh: nama, nama1, p1.

Cerdas Menguasai Git, First Edition.

23

By Rolly M. Awangga Copyright © 2019 Kreatif Industri Nusantara.

3. Nama variabel tidak boleh diawali dengan angka
4. Karakter bersifat case-sensitive (huruf besar dan huruf kecil dibedakan), contoh: Nama dan NAMA keduanya memiliki arti yang berbeda dan merupakan variabel yang berbeda.
5. Nama variabel tidak boleh menggunakan kata kunci yang ada pada bahasa pemrograman python, contoh: if, else, while

3.2 Input dan Output

Input & output bertujuan agar pengguna dan program dapat berinteraksi, Perintah input() berguna untuk meminta inputan dari user, sehingga memungkinkan user untuk menginputkan data.

Perintah print() berguna untuk menampilkan output dari data yang diinputkan oleh user, sehingga data yang diinputkan user dapat ditampilkan ke layar.

Contoh dari penggunaan input dan output adalah sebagai berikut:

```

1 #Input yang ditujukan untuk user
2 nama=informaticsrsearchcenter
3
4 #output yang didapatkan user
5 print( Halo , nama, selamat datang )

```

3.3 Operasi Aritmatika

Python memiliki operasi aritmatika, antara lainnya seperti :

1. penjumlahan (+)
2. pengurangan (-)
3. perkalian (*)
4. pembagian (/)
5. sisa bagi/modulus (%)
6. pemangkatan (**)

Penggunaan dari simbol simbol ini sama hal nya dengan fungsi aritmatika pada umumnya.

3.4 Perulangan

Dalam membuat sebuah program, terkadang kita memerlukan satu baris atau satu

blok kode yang sama secara berulang, disini fungsi perulangan dipakai sehingga kita tidak perlu menulis baris atau blok kode yang sama secara terus menerus, dalam python perulangan dibagi menjadi 2, yaitu for dan while.

3.4.1 For

For merupakan perulangan yang akan mengulang kondisi true sampai batas yang telah ditentukan, biasanya digunakan untuk perulangan yang mana parameter pengulangannya menggunakan list atau range. Berikut ini merupakan contoh penggunaan sintaks perulangan for.

```
1 for i in range(0,10):
2     print(i)
```

3.4.2 While

While merupakan perulangan yang akan terjadi apabila kondisinya True, perulangan akan terus berjalan hingga diperoleh kondisi False. Berikut ini merupakan contoh penggunaan sintaks perulangan while.

```
1 #perulangan while
2 hitung=0
3 while(hitung < 9):
4     print ( hitungan ke :      , hitung)
5     hitung = hitung + 1
6
7 print(" Good bye ! ")
```

3.5 Kondisi

Pengambilan keputusan kadang diperlukan dalam sebuah program untuk menentukan tindakan apa yang akan dilakukan sesuai dengan kondisi yang terjadi, contoh kasus misalkan ada seorang anak bernama idam, seorang manusia yang membutuhkan makan, jika idam lapar maka idam akan makan. Maka dapat dijabarkan seperti dibawah ini :

Kondisi, jika :

Idam lapar

Maka :

Idam akan makan

Namun terkadang kondisi juga diberikan tambahan opsi sebuah kondisi tambahan, misalkan jika idam makan maka idam kenyang, namun jika tidak maka idam akan kelaparan. Penjabarannya dapat dilihat sebagai berikut :

Kondisi, jika :

Idam makan

Maka :

Idam akan kenyang
Jika tidak :
Idam akan kelaparan

Contoh diatas dapat ditulis dalam syntax python dengan menggunakan kondisi, pengkondisian dalam python dibagi menjadi 4, yaitu : IF, IF ELSE, ELIF, nested IF. Berikut merupakan pembahasannya.

3.5.0.1 IF IF adalah suatu struktur yang memiliki suatu perlakuan jika terjadi suatu kondisi. Akan tetapi, tidak terjadi sesuatu yang lain atau terjadi apa-apa ketika berada di dalam luar kondisi tersebut. IF hanya menjalankan satu kondisi dan menampilkan satu output. Contoh: kondisi dimana variabel a lebih besar dari variabel b, maka tampilkan hasil bahwa a lebih besar dari b.

```
1 #ifstatement
2 a = 330
3 b = 200
4 if a > a:
5     print("alebihbesardarib")
```

3.5.0.2 IF ELSE IF ELSE digunakan apabila kondisi yang terjadi bernilai salah, maka lakukan else. Contoh: kondisi dimana variabel a lebih besar dari variabel b, maka jika b lebih besar dari a, tampilkan hasil b lebih besar dari a, jika salah maka tampilkan a lebih besar dari pada b

```
1 #else
2 a = 200
3 b = 33
4 if b > a:
5     print("b is greater than a")
6 else:
7     print("aisgreaterthanb")
```

3.5.0.3 ELIF Kondisi ELIF merupakan suatu strktur logika majemuk yang memiliki banyak pilihan aksi terhadap berbagai kemungkinan kejadian yang terjadi. ELIF digunakan apabila kondisi pertama tidak benar maka lakukan kondisi lain (alternatif). Contoh: kondisi dimana variabel a sama dengan variabel b, maka jika b lebih besar dari a, tampilkan hasil b lebih besar dari a, namun jika a dan b bernilai sama, maka tampilkan a sama dengan b

```
1 #elif
2 a = 33
3 b = 33
4 if b > a:
5     print("b lebih besar dari a")
6 elif a == b:
7     print("a sama dengan b")
```

3.5.0.4 Nested IF Nested if merupakan if didalam if (if bersarang), terdapat dua if didalam satu kondisi. Contoh: variabel x sama dengan 41, kondisi pertama yaitu

jika x besar dari 10 maka tampilkan lebih besar dari 10, kondisi kedua yaitu jika x besar dari 20, maka tampilkan lebih besar dari 20, jika salah maka tampilkan tidak melebihi 20.

```

1 #nestedif
2 x = 41
3
4 if x > 10:
5     print("lebih besar dari 10,")
6     if x > 20:
7         print("lebih besar dari 20!")
8     else:
9         print("tidak melebihi 20.")

```

3.6 Error

1. NameError, terjadi apabila kode mengeksekusi nama yang tidak terdefiniskan.
Contoh:

```

1 nama = "Dinda Majesty"
2 print(Nama)

```

Maka akan menghasilkan output NameError: name Nama is not defined. error ini dapat diatasi dengan mengubah variabel yang di print sesuai dengan variabel yang didefinisikan, karena penulisan pada python bersifat case-sensitive

2. SyntaxError, terjadi apabila kode python mengalami kesalahan saat penulisan. Contoh: menuliskan variabel yang didahului angka (1nama = Dinda Majesty) maka akan muncul error SyntaxError: invalid syntax. error ini dapat diatasi dengan memperhatikan tata cara penulisan kode pada bahasa pemrograman python.
3. Logic error merupakan kesalahan yang terjadi karena kesalahan pembacaan data pada command perintah seperti data tidak terbaca atau tidak ada, dan tidak sesuai dengan aturannya. Contoh kesalahan tipe data yaitu

```

1 a= 4
2 b=6
3
4 print(a+b)

```

4. TypeError, terjadi apabila kode melakukan operasi atau fungsi terhadap tipe data yang tidak sesuai. Contoh: melakukan penjumlahan terhadap tipe data string dan integer. error ini dapat diatasi dengan mengubah tipe data string menjadi integer.

```

1 a = "10"
2 b = 5
3
4 print(a + b)

```

Maka akan menghasilkan output error TypeError: can only concatenate str (not int) to str

5. `IndentationError`, terjadi apabila kode perulangan atau pengkondisian tidak menjorok kedalam (tidak menggunakan indentasi), error ini dapat diatasi dengan menambahkan tab atau spasi. Contoh


```
1 a = 200
2 b = 330
3
4 if b > a:
5     print("lebih besardaria")
```

Maka akan menghasilkan output eror `IndentationError: expected an indented block`

3.7 Try Except

Try Except merupakan salah satu bentuk penanganan error di dalam bahasa pemrograman python, perintah try except ini memiliki fungsi untuk menangkap sebuah error dan tetap menjalankan program kita, sehingga program yang sedang dijalankan akan mengeksekusi program hingga akhir. Contohnya terdapat pada listing berikut

```
1 a="1"
2 b=2
3
4 try:
5     a+b
6 except:
7     print("Error, keduatipedata berbeda")
```

BAB 4

FUNGSI DAN KELAS

4.1 Teori

4.1.1 Fungsi

Fungsi adalah sebuah blok kode yang memiliki nama fungsi dan kode program di dalamnya jika dijalankan maka fungsi itu akan mengembalikan nilai. Fungsi dapat dipanggil berkali-kali sesuai dengan nama fungsi yang telah didefinisikan. Fungsi memiliki nilai kembalian (return). Contoh fungsi

```
1 def nambahinAngka ( angka1 , angka 2 ) :  
2     hasil = angka 1 + angka 2  
3     return hasil
```

Apabila kita dapat memberikan nilai ke angka1 dan angka2, dan apa bila sudah

diberi nilai dan program sudah dijalankan, maka program pun akan mengembalikan nilai berupa hasil dari penjumlahan angka 1 dan angka 2.

Cerdas Menguasai Git, First Edition.

29

By Rolly M. Awangga Copyright © 2019 Kreatif Industri Nusantara.

4.2 Package

Package merupakan sekumpulan modul yang dikemas oleh programmer dengan tujuan agar mempermudah dalam pembuatan kode program. Kita dapat membuat sebuah kode program atau fungsi didalamnya dan dapat secara mudah menggunakan kode program itu dengan cara memanggilnya pada kode program lainnya atau import package. Contoh nya adalah sebagai berikut

```

1 def my_biodata( nama, umur ):
2     bio = "nama saya " + nama + " umur saya " + umur
3     return bio
4
5 def my_study( kampus, prodi ):
6     study = "saya berkuliah di " + kampus + " program studi " + prodi
7     return study

```

Kode diatas merupakan isi dari le fungsi.py, sedangkan saya ingin menjalankan program fungsi.py pada main.py sehingga kode program pada le main.py akan dituliskan seperti berikut:

```

1 import fungsi
2 nama = "Dinda Majesty" umur
3 = "19 Tahun"
4 biodata = my_biodata(nama, umur)
5 print( biodata )
6
7 kampus = "Politeknik Pos Indonesia"
8 prodi = "D4-Teknik Informatika"
9 kuliah = my_study( kampus, prodi )
10 print( kuliah )

```

Kode program pada le main.py akan mengimport kode program yang ada pada le fungsi.py, sehingga dengan adanya fungsi dan package kita dapat dengan mudah melakukan pemanggilan fungsi yang telah kita deskripsikan sebelumnya, walaupun berada pada le python yang berbeda.

4.3 Class, Object, Atribute, and Method

Class atau Kelas merupakan sebuah blueprint/kerangka dari objek yang berisi fungsi dan dibuat untuk mendefenisikan objek dengan atribut yang sesuai dengan kelas yang telah dibuat yang nantinya akan diinisiasikan. Objek adalah sebuah wujud yang dapat kita lakukan perintah sesuai dengan methodnya, Sebuah kelas harus memiliki objek yang nantinya akan di kodekan sesuai dengan fungsi yang telah dibuat pada kelas, tanpa adanya objek sebuah kelas tidak akan bisa menjalankan fungsi-fungsi didalamnya. Atribut berisi variabel yang memiliki tipe data dan dapat kita berikan pada objek, atribut ada 2 yaitu kelas atribut dan instansi atribut, perbedaannya hanya di letak, kalau kelas atribut ada di bawah kelas, dan instansi atribut ada didalam

fungsi, atribut itu sebuah variabel yang dimiliki oleh parentnya seperti fungsi atau class. `.Method` merupakan kode program yang berisi tindakan atau perintah untuk menjalankan objek.

```

1 class Fungsi(object):
2
3 def Nama(self, namakamu):
4     self.kamu = namakamu

```

4.4 Pemanggilan Class

Pemanggilan library kelas dapat dilakukan dengan cara import dan membuat objek dari kelas tersebut. Contohnya, kita memiliki le python yang diberi nama ngitung dan didalamnya terdapat class Ngitung yang memiliki banyak fungsi didalamnya. Untuk melakukan pemanggilan class maka kita bisa mengetikkan kode seperti berikut.

```

1 import Fungsi

```

4.5 Pemakaian Package Fungsi Apabila File Didalam Folder

Pemakaian Package fungsi apabila le terdapat didalam sebuah folder maka kita bisa menggunakan from folder import le dan from le import fungsi. Contohnya, kita memiliki folder src yang didalamnya terdapat le fungsi.py dan didalam fungsi.py terdapat fungsi Berhitung, untuk mengimportkan fungsi maka kita dapat mengetikkan kode seperti berikut.

```

1 from src import fungsi
2 from fungsi import Berhitung

```

4.6 Pemakaian Package Kelas Apabila File didalam Folder

Pemakaian package kelas apabila le terdapat didalam sebuah folder maka kita bisa menggunakan from folder import le dan from le import kelas. Contohnya, kita memiliki folder src yang didalamnya terdapat le fungsi.py dan didalam fungsi.py terdapat kelas Ngitung, maka untuk melakukan import kelas kita dapat mengetikkan kode sebagai berikut.

```

1 from src import fungsi
2 Kelas = fungsi.Nama(namakamu)

```


DAFTAR PUSTAKA

1. R. Awangga, "Sampeu: Servicing web map tile service over web map service to increase computation performance," in *IOP Conference Series: Earth and Environmental Science*, vol. 145, no. 1. IOP Publishing, 2018, p. 012057.

Cerdas Menguasai Git, First Edition.

By Rolly M. Awangga Copyright © 2019 Kreatif Industri Nusantara.

33

