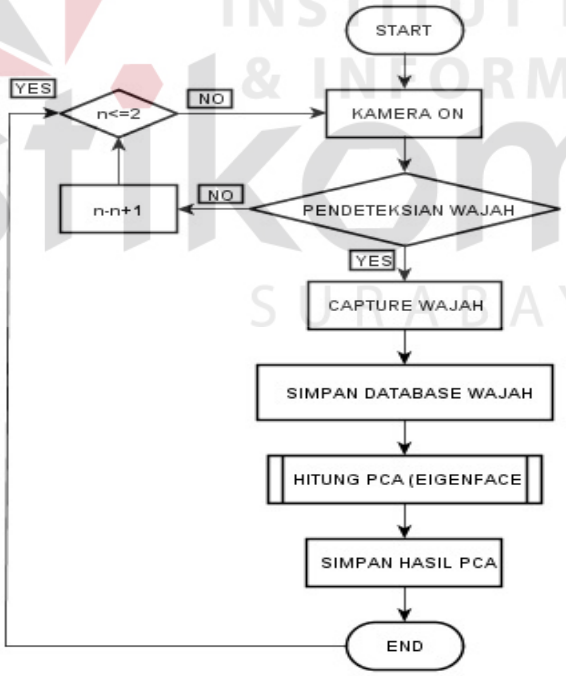
BAB II

LANDASAN TEORI

2.1 Metode *Eigenface*

Metode *eigenface* adalah sekumpulan *standardize face ingredient* yang diambil dari analisis statistik dari banyak gambar wajah. Supaya menghasilkan *eigenface*, sekumpulan citra digital dari wajah manusia diambil pada kondisi pencahayaan yang sama kemudian dinormalisasikan dan diproses pada resolusi yang sama (misal *m* x *n*), kemudian citra tadi diperlakukan sebagai vektor dimensi *m* x *n* dimana komponennya diambil dari nilai piksel citra. (Mulyawan, Ridho Dyakso, 2015). Berikut adalah *flowchart* dari pendeteksian wajah menggunakan metode *eigenface* :



Gambar 2.1 *Flowchart* sistem pendektesian wajah menggunakan metode *eigenface* (Bayu, Setya, 2009)

Dari *flowchart* pada gambar 2.1 dapat dijelaskan bahwa proses melakukan pengenalan / deteksi wajah menurut (Bayu, Setya, 2009) adalah sebagai berikut :

1. Aktifkan *webcam* untuk menampilkan gambar yang ditangkap *webcam* kedalam aplikasi.
2. Penangkapan citra wajah (*image capturing*) dapat dilakukan secara langsung (*real time*) menggunakan *webcam*, setelah terdeteksi adanya gambar wajah pada tampilan *window* dari *webcam*.
3. Apabila sistem tidak menemukan / mendeteksi wajah maka sistem akan melakukan pengulangan proses pendekteksian sebanyak 2x apabila tetap terdeksi maka proses akan dianggap selesai.
4. Melakukan pemrosesan awal yang meliputi, normalisasi ukuran citra, RGB ke *grayscale*, *resize* untuk membuang bagian daerah selain wajah sehingga hanya bagian wajah saja yang diproses dan normalisasi pencahayaan ketika mengambil citra *input*.
5. Simpan data wajah yang diambil.
6. Kemudian dilakukan proses PCA untuk mengutip bagian terpenting dengan metode *eigenface* sehingga didapatkan *eigenvector* dan *eigenvalue* dari gambar tersebut.
7. Data yang disimpan nantinya akan digunakan sebagai nilai pembanding pada proses penghitungan jarak untuk pengenalan wajah.

2.1.1 Konsep *Eigenface*

Merupakan suatu metode pengenalan wajah yang berdasarkan pada algoritma *Princpal Component Analysis* (*PCA*). Yaitu mempresentasikan citra dalam sebuah gabungan vector yang dijadikan satu matriks tunggal. Dari matriks tunggal tersebut akan diekstraksi suatu ciri utama yang akan membedakan antara citra wajah satu dengan lainnya. Citra yang digunakan adalah citra digital dengan format grayscale. Dengan membandingkan antara citra uji dengan citra referensi menggunakan konsep jarak euclidean, maka akan didapat kesimpulan apakah suatu citra wajah dikenali atau tidak dikenali. (Urifan, Isbadi, 2010)

2.1.2 Proses Sistem Pengenalan Wajah dengan *Eigenface*

1. Preprocessing

Tujuan dari *preprocessing* adalah agar data tersebut dapat diolah sehingga menghasilkan data yang sesuai dengan kebutuhan. Pada tahap *preprocessing* terdapat 4 langkah yaitu konversi citra RGB mejadi *grayscale* pada citra wajah. (Mulyawan, Ridho Dyakso, 2015)

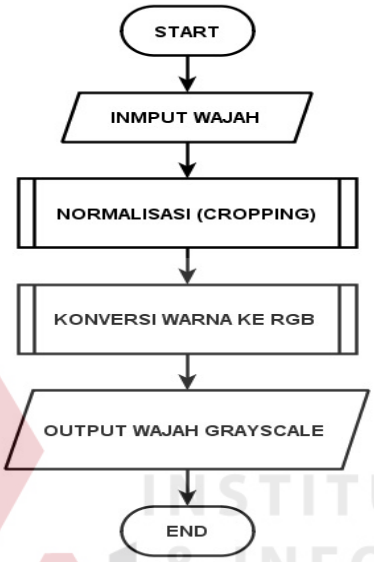
Grayscale sendiri adalah warna – warna dalam *pixel* yang berada dalam rentang gradasi warna hitam dan putih. Citra yang ditampilkan dari *grayscale* adalah citra yang terdiri atas warna abu – abu, berorientasi pada warna hitam pada gradasi warna dengan intensitas lemah dan warna putih pada gradasi dengan intensitas warna kuat.

Citra *grayscale* memiliki format 8 bit pada setiap piksel, yang memiliki intensitas tingkatann warna sebanyak 256 tingkatan. Untuk mengubah citra berwarna dalam suatu *image* R,G, dan B menjadi *image* grayscale, maka dapat

dikonversi dengan mengambil nilai rata – rata dari nilai R, G, dan B sehingga dapat dituliskan dalam rumus sebagai berikut :

$$X = (R+G+B) / 3 \quad \text{.....(2.1)}$$

Berikut adalah *flowchart* rancangan *grayscale* :



Gambar 2.2 *Flowchart* Rancangan *Grayscale*

Sesuai dengan *flowchart* pada gambar 2.2 bahwa rancangan *grayscale* dalam tahap pengenalan wajah adalah setelah program melakukan proses cropping gambar yaitu dimana hanya intens pada gambar wajah saja dan setelah itu melakukan proses pengkorvesian dari gambar berwarna menjadi *grayscale* dengan rumus (2.1).

2. Ekstrasi Fitur

Tujuan dari ektraksi fitur adalah mencari informasi-informasi yang merupakan ciri khusus dari setiap citra wajah. Pengambilan ciri pada *database training*

melalui tahap penghitungan nilai *eigenface* dan PCA, sedangkan pengambilan ciri pada *database test* hanya melalui tahap penghitungan PCA. Kemudian ciri ini akan digunakan sebagai penghitungan kesamaan jarak antar citra wajah *training* dengan citra wajah *test*. (Mulyawan, Ridho Dyakso, 2015). Pada tahap ektraksi fitur terdapat beberapa langkah yaitu :

1. Menghitung nilai *mean* citra wajah

$$m = \frac{Jumlah\ tiap\ baris}{Jumlah\ CItra} \quad \text{.....(2.2)}$$

2. Menghitung selisih antara nilai m dengan matriks Transpose

$$A = T - m \text{.....(2.3)}$$

3. Menghitung nilai matriks kovarian

$$L = A' \times A \text{.....(2.4)}$$

4. Menghitung nilai *eigenvalue* (λ) dan *eigenvector* (*v*) dari matriks kovarian (*C*)

$$C \times v_i = \lambda_i \times v_i \text{.....(2.5)}$$

5. Menghitung nilai *eigenface* (μ)

$$E = A \times V \text{.....(2.6)}$$

6. Menghitung nilai bobot

$$W = E' \times A \text{.....(2.7)}$$

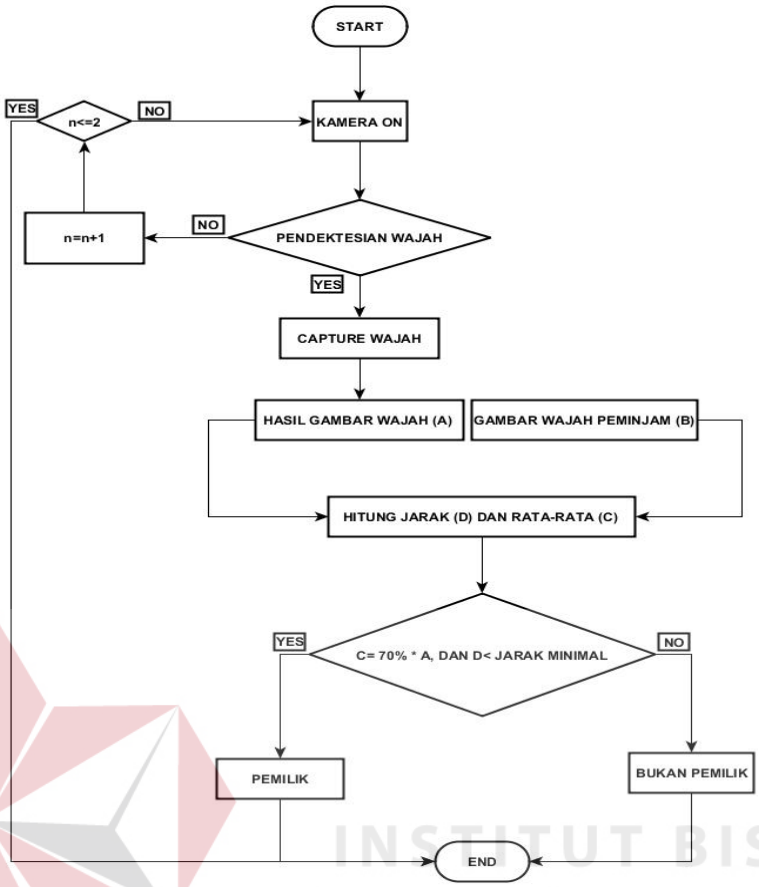
3. Nilai Bobot

Jarak terpendek antara nilai bobot *database training* dengan nilai bobot citra wajah *test*. Pencarian jarak terpendek ini menggunakan metode *manhattan distance*. (Mulyawan, Ridho Dyakso, 2015)

$M_d = |Wt - Wr|$ ……...…(2.8)

2.2 Template Matching

Template matching adalah teknik dalam pengolahan citra digital untuk menemukan bagian-bagian kecil dari suatu gambar yang cocok atau mempunyai korelasi yang kuat dengan gambar *template*. Pada dasarnya adalah menempatkan *template* pada suatu posisi didalam citra inputan dan untuk menentukan posisi tersebut dapat dilakukan dengan membandingkan membandingkan nilai intensitas dalam *template* dengan nilai-nilai intensitas yang sesuai pada citra inputan. Karena jarang terjadi bahwa nilai-nilai intensitas pada *template* dan citra akan sama persis, maka dibutuhkan pengukuran tingkat kemiripan antara *template* dengan gambar atau perbedaan antara nilai intensitas *template* dan nilai-nilai intensitas pada citra. (Gandhi, Arfive, 2012). Berikut pada gambar 2.3 adalah *flowchart* algoritma pencocokan wajah menggunakan metode *template matching* :



Gambar 2.3 *Flowchart* pencocokan wajah (Bayu, Setya, 2009)

Dari *flowchart* pada gambar 2.3 dapat dijelaskan bahwa proses pencocokan wajah dimulai pada saat proses pengambilan isi loker, setelah sistem mendeteksi wajah maka kamera akan meng-*capture* / mengambil gambar wajah pengguna dan mulai melakukan proses pencocokan wajah dengan mengambil contoh wajah pengguna yang melakukan peminjaman diawal. Setelah mengambil sample gambar wajah peminjam dan wajah pengguna yang mengambil isi loker sistem akan mulai melakukan perhitungan nilai jarak dan rata – rata dari kedua gambar tersebut. (Purba, Antonov Ramen, 2013)

Sebagai contoh :

Gambar 2.4 Contoh pencocokan gambar wajah.

Mula – mula diumpamakan gambar pada wajah pengambilan menpunya nilai 100 dan wajah pada peminjam bernilai 130, pada contoh kali ini sudah ditentukan bahwa nilai rata- rata kecocokan wajah minimal bernilai 70 dan jarak maksimal kemiripan adalah 20, maka akan dicari nilai jarak dan rata – rata sebagai berikut :

Jarak (D) = (Gambar B – Gambar A) = 30

Rata – rata = ((Gambar B + Gambar A) / 2) = 115

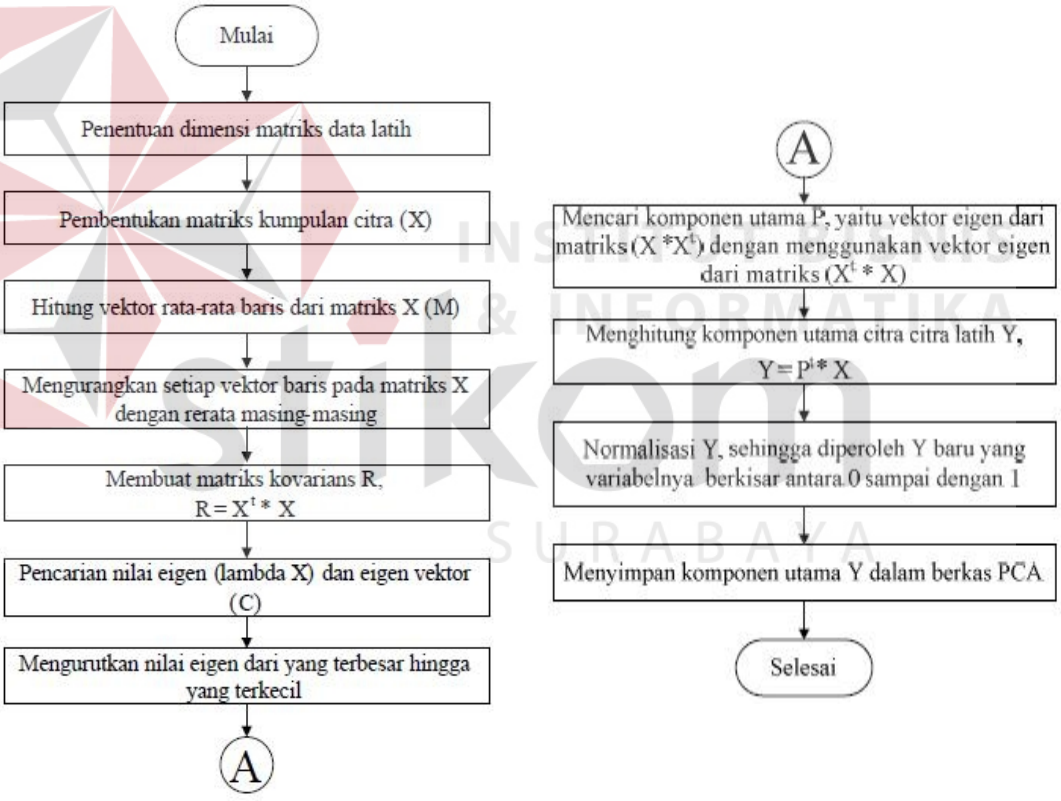
Setelah mengetauhi nilai dari jarak dan rata – rata maka sistem akan mulai mengkalkulasi apakah Gambar B cocok / mirip dengan Gambar A dengan menggunakan rumus :

C = 70% * Gambar A dan Gambar D < nilai jarak minimum kemiripan

Apabila nilai kecocokan (C) lebih dari 70 dan nilai jarak maksimal kurang dari nilai jarak (D) maka Gambar A akan dikenali sebagai Gambar B / cocok, jika tidak maka gambar akan dinyatakan tidak cocok.

2.3 *Principal Component Analysis* (PCA)

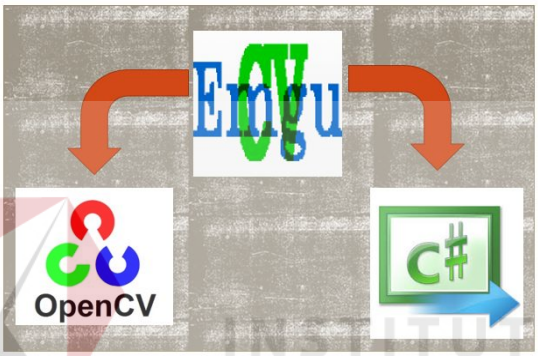
Metode *PCA* merepresentasikan citra dalam satu vektor ciri, misalkan *I* adalah matriks representasi dari citra wajah yang berukuran N1 x N2 Vektor ciri dari citra *I* diperoleh dengan cara merangkai baris-baris menjadi satu *vektor* baris. Baris kedua dirangkai setelah baris pertama, dilanjutkan baris ketiga dan seterusnya sampai hanya terdapat satu baris. Jadi dimensi dari *vektor* ciri tersebut adalah banyaknya *pixel* pada citra yaitu N1 dan N2. Tujuannya mereduksi dimensi dari *vektor* ciri tersebut. Berikut adalah *flowchart* logika dari cara kerja PCA :



Gambar 2.5 Contoh *flowchart* logika cara kerja PCA

2.4 Emgu CV

EmguCV adalah *wrapper* bagi OpenCV *library* untuk C#. OpenCV sendiri adalah *library* yang berisi berbagai metode untuk membantu pengembang mengolah citra dan *computer vision*. OpenCV ditujukan bagi pengembang bahasa C++. Bahasa mudahnya adalah, karena OpenCV ditujukan bagi bahasa C++, maka beberapa pengembang, mengembangkan EmguCV sebagai penerjemah, sehingga para pengembang C# pun dapat mempergunakan OpenCV *Library*.



Gamabr 2.6 Emgu Cv berfungsi sebagai penghubung antara Open Cv dengan c#

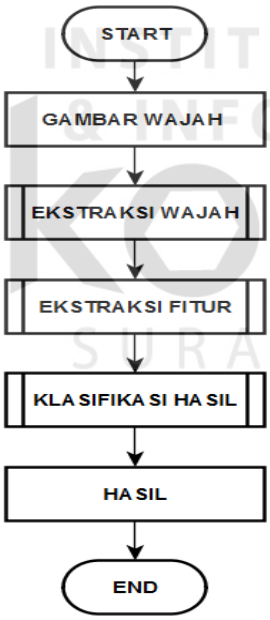
2.5 *Face Detection* menggunkan *Haar Cascade Classifier*

Face Detection adalah teknologi komputer yang digunakan dalam berbagai aplikasi yang mengidentifikasi wajah manusia dalam gambar digital. Deteksi wajah juga merupakan salah satu tahap praproses yang sangat penting di dalam sistem pengenalan wajah yang digunakan untuk sistem biometrik.

Deteksi wajah juga dapat digunakan untuk pencarian dan pengindeksan citra atau *video* yang di dalamnya terdapat wajah manusia dalam berbagai ukuran, posisi, dan latar belakang. Bagaimana sistem deteksi wajah ini memproses gambar dari obyek bergerak. Pemrosesan gambar ini bertujuan untuk mencari wajah dari gambar obyek bergerak yang telah di-*capture*, kemudian gambar tersebut diolah

dengan memisahkan gambar dengan latar belakangnya, sehingga hanya bagian yang dianggap kulit yang ditampilkan sedangkan bagian yang bukan kulit akan dihitamkan. Pemisahan gambar dengan latar belakang ini bertujuan untuk memudahkan proses pencarian wajah.

Deteksi objek wajah menggunakan *Haar Cascade classifier* yang merupakan sebuah fungsi pendeteksian objek wajah yang dimiliki oleh aplikasi OpenCV. *Face detector* akan melakukan proses pengujian tiap lokasi *image* dan mengklasifikasi sebagai wajah dan bukan wajah. Klasifikasi wajah ini memakai suatu ketetapan skala, misalnya 100×100 *pixel*. Jika wajah pada objek *image* lebih besar atau lebih kecil dari *pixel* tersebut, *classifier* terus menerus jalan beberapa kali, untuk mencari wajah pada gambar tersebut. Berikut adalah flowchart alur dari metode *haar cascade* :



Gambar 2.7 *Flowchart* alur kerja dari metode *haar cascade*

Sesuai dengan gambar 2.7 bahwa alur kerja metode *haar cascade* terdiri dari 3 hal yaitu :

1. Ekrtraksi Wajah

   Pada bagian ektraksi wajah ini sistem melakukan proses pendeteksi / pengkalsifikasi (pengambilan gambar) apakah gambar tersebut merupakan ciri dari wajah.

2. Ekstraksi Fitur

   Pada bagian ektraksi fitur ini sistem melakukan mengolahan dari hasil ekraksi wajah dengan mengola gambar dengan melakukan proses pemisahan latar belakang dengan dengan objek.

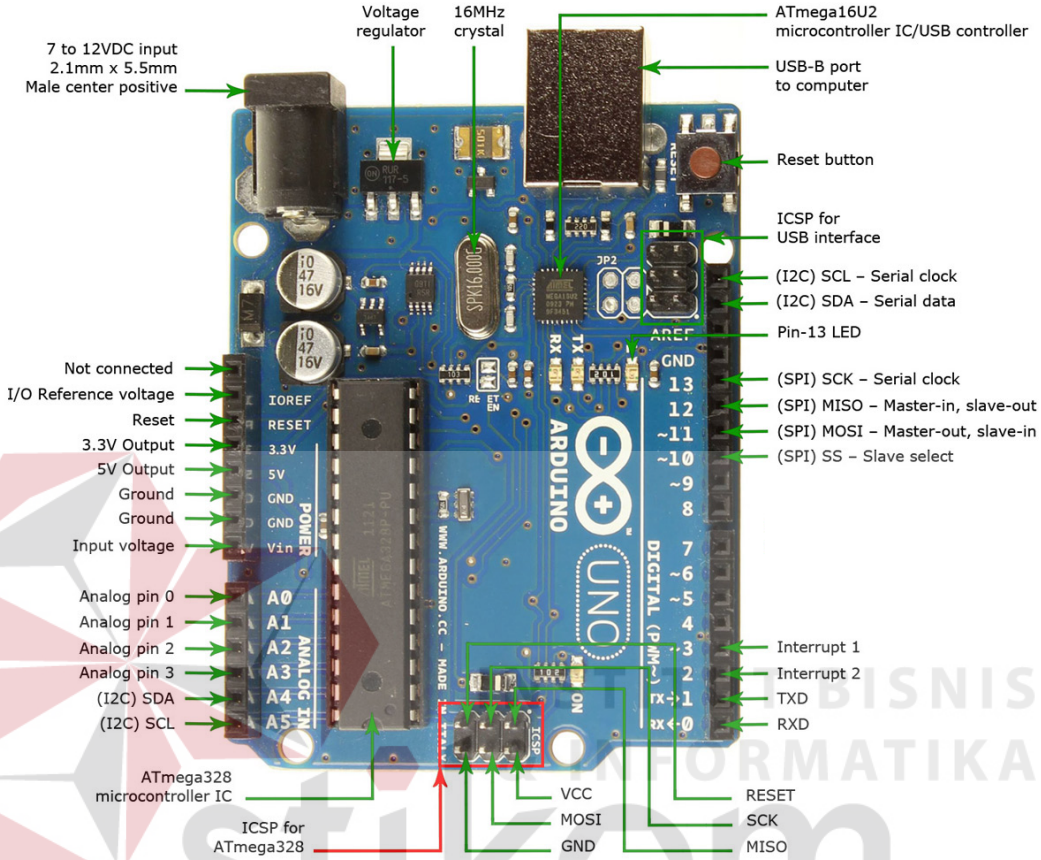
3. Klasifikasi Hasil

   Pada bagian klasifikasi ini sistem melakukan penghitungan algoritma setelah objek terpisah dengan latar belakang dan melakukan kalsifikasi apakah objek tersebut adalah wajah.

2.6 Arduino Uno

Arduino Uno adalah sebuah *board* mikrokontroler yang berbasis pada mikrokontroler ATmega328. Suatu mikrokontroler bekerja dengan mengeksekusi perintah- perintah dalam suatu program yang diunggah ke dalam *board*. Arduino Uno memiliki 14 buah *pin* yang dapat difungsikan sebagai *input/output* digital, sehingga dapat dihubungkan dengan perangkat *input* seperti *sensor* untuk membaca kondisi dalam suatu radius tertentu, selain itu juga dapat dihubungkan dengan perangkat *output* lain seperti motor DC dan lampu LED. Selain itu juga Arduino Uno memiliki 6 pin analog, dan tombol reset. Mikrokontroler ini dapat beroperasi pada tegangan 12 Volt yang dapat diaktifkan melalui kabel USB atau berasal dari

tegangan catu daya *eksternal* seperti *baterai*. Berikut ini detail mengenai Arduino Uno.



Gambar 2.8 Spesifikasi dari Arduino Uno

Pada gambar 2.8 terdapat port – port yang mempunyai kegunaan masing, berikut adalah penjelasan singkatnya :

1. USB Port adalah penghubung antara aplikasi VS yang terdapat pada laptop dengan arduino yang terhubung secara serial.
2. SCL – Serial Clock dan SDA – Seria Data adalah port yang berfungsi membawa informasi melalui komunikasi I^2C (*Inter Integrated Circuit*) yang dihubungkan dengan mikrokontroller master dan slave

3. SCK – MISO – MOSI – SS adalah penghubung antara arduino dengan RFID *reader* dimana data yang dikirm dari MISO (*Master In Slave Out*) adalah data gelombang yang masuk dari RFID *tag card* yang kemudian dilanjutkan pengiriman data ke IC mikrokontroler yang terdapat pada arduino melalui MOSI (*Master Out Slave In*), dan SS (*Slave Select*) digunakan sebagai mengaktifkan port *slave*.

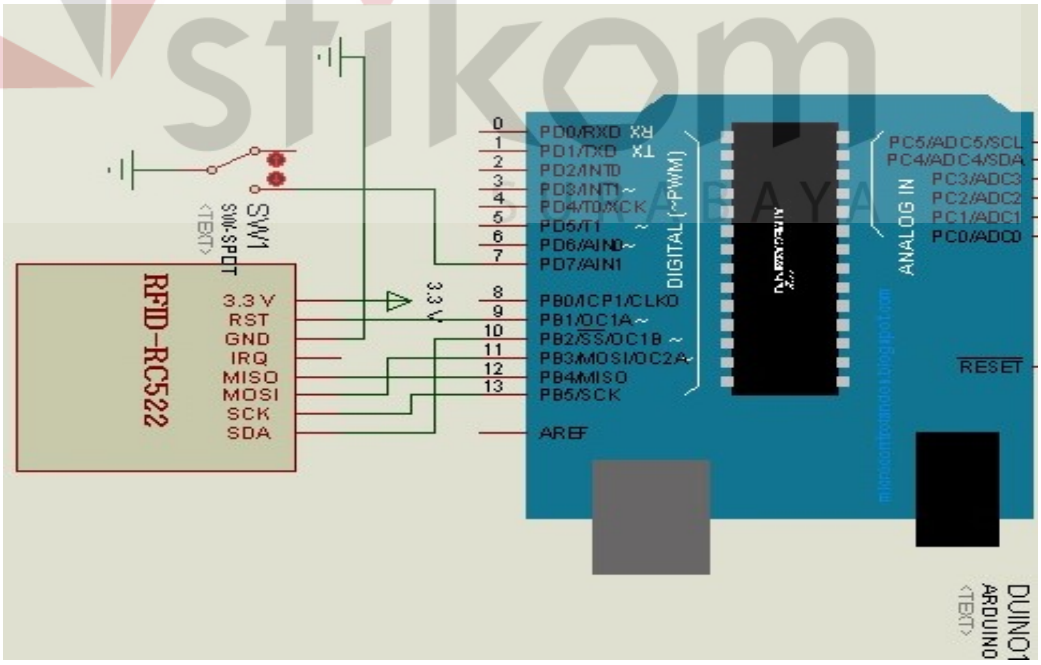
2.7 RFID (*Radio Frequency Identification*)

RFID adalah proses identifikasi frekuensi gelombang radio. RFID menggunakan frekuensi radio untuk membaca informasi dari sebuah alat yang disebut RFID *tag Card*. Sebuah sistem RFID terdiri dari RFID *Reader* dan RFID *tag Card*. RFID *Reader* dan RFID *tag Card* tersedia dalam bermacam-macam jenis, khusus untuk RFID *tag Card* setiap kartu memiliki data ASCII yang berbeda-beda. Fungsi umum dari RFID *Reader* adalah sebagai penerima gelombang radio (RF), sedangkan fungsi umum dari RFID *tag Card* sebagai pemancar gelombang radio (RF). RFID *Reader* hanya dapat menangkap data RFID *tag Card* yang telah disesuaikan.

RFID *tag Card* akan mengenali diri sendiri ketika mendeteksi sinyal dari alat yang kompatibel, yaitu RFID *Reader*. RFID merupakan teknologi identifikasi yang fleksibel, mudah digunakan dan sangat cocok untuk operasi otomatis. RFID mengkombinasikan keunggulan yang tidak tersedia pada teknologi identifikasi yang lain. RFID dapat disediakan dalam alat yang hanya dapat dibaca saja (*Read Only*) atau dibaca dan ditulis (*Read/Write*), tidak memerlukan kontak langsung maupun jalur cahaya untuk dapat beroperasi, dapat berfungsi pada berbagai variasi

kondisi lingkungan, dan menyediakan tingkat integritas data yang tinggi. Sebagai tambahan, karena teknologi ini sulit dipalsukan, maka RFID dapat menyediakan tingkat keamanan yang tinggi. Pada sistem RFID, umumnya *tag Card* ditempelkan pada suatu obyek. Ketika *tag Card* ini melalui medan listik yang dihasilkan oleh RFID *Reader* yang sesuai, *tag Card* akan mentransmisikan informasi yang ada pada *tag Card* kepada RFID *Reader*, sehingga proses identifikasi dapat dilakukan. RFID terdiri dari tiga komponen, di antaranya adalah:

- RFID *tag Card*: Alat yang menyimpan informasi untuk identifikasi objek. RFID *tag Card* juga sering disebut *transponder.*
- RFID *Reader*: Alat yang kompatibel dengan *tag Card* RFID yang berkomunikasi secara wireless dengan *tag Card*
- Antena: Alat untuk mentransmisikan sinyal RF antara RFID *Reader* dengan RFID *tag Card.* (Diredja,2010)



Gambar 2.9 Port RFID *Reader* terhubung dengan Arduino Uno

2.8 WebCamera / Kamera Web

Webcam atau kamera web, pada dasarnya adalah sebuah kamera digital yang terhubung ke komputer, yang berfungsi untuk mengambil citra yang akan diolah oleh komputer. Pada awalnya webcam digunakan sebagai alat komunikasi yang menampilkan rentetan citra dan dapat diakses melalui world wide web. Namun, seiring perkembangannya webcam digunakan juga untuk keperluan lainnya. (Perkasa, Richard Therzian, 2014)

2.9 Microsoft Visual Studio

Microsoft Visual Studio merupakan sebuah perangkat lunak lengkap (suite) yang dapat digunakan untuk melakukan pengembangan aplikasi, baik itu aplikasi bisnis, aplikasi personal, ataupun komponen aplikasinya, dalam bentuk aplikasi *console*, aplikasi *Windows*, ataupun aplikasi Web. Visual Studio mencakup kompiler, SDK, *Integrated Development Environment* (IDE), dan dokumentasi (umumnya berupa MSDN *Library*). Kompiler yang dimasukkan ke dalam paket Visual Studio antara lain Visual C++, Visual C#, Visual Basic, dan Visual Basic.