BAB IV

HASIL DAN PEMBAHASAN

Pada bagian ini akan membahas mengenai program *Self Check-In* dan proses uji coba sistem yang telah dibuat berdasarkan pada prosedur pembuatan yang telah dijelaskan pada baba sebelumnya. Untuk mengetahui tingkat keberhasilan sistem dalam proses mendeteksi maupun mengenali wajah seseorang menggunakan metode *Viola-jones* sebagai pendeteksi wajah dan algoritma *Local Binary Pattern Histogram (LBPH)* untuk pelabelan dan mengidentifikasi wajah (*face recognition*).

1.1. Pembuatan *Coding*

1.1.1. *Coding* Proses Deteksi Wajah

```
import cv2
import numpy as np
import sqlite3

def active_camera(self):
    cam=cv2.VideoCapture(0)
    faceDetect=cv2.CascadeClassifier("D:\Software TA\opencv\sources\data\haarcascades\haarcascade_frontalface_default.xml")
    sampleNum=0;
    nopassport = self.nopassport_lineEdit.text()
    nama = self.uname_lineEdit.text()
    while(True):
        ret,img=cam.read();
        gray=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
        faces=faceDetect.detectMultiScale(gray,1.3,5);
        x=0
        y=0
        for(x,y,w,h) in faces:
            sampleNum=sampleNum+1;
            cv2.rectangle(img,(x,y),(x+w,y+h),(0,0,255),2)
            cv2.imwrite("dataSet/user."+str(nopassport)+"."+str(nama)+str(sampleNum)+".jpg",gray[y:y+h, x:x+w])
        cv2.imshow("face",img);
        cv2.waitKey(1000)& 0xff
        if sampleNum>2:
            connection = sqlite3.connect("data.db")
            namaFile="dataSet/user."+str(nopassport)+"."+str(nama)+str(sampleNum)+".jpg"
            foto = "UPDATE USERS SET FOTO='{0}' WHERE NOPASSPORT='{1}'".format(namaFile,nopassport)
            cv2.imwrite(namaFile,gray[y:y+h,x:x+w])
            connection.execute(foto)
            connection.commit()
            connection.close()


            break
    cam.release()
    cv2.destroyAllWindows()
```

Gambar 4.1. *Coding* deteksi wajah

Pada Gambar 4.1. diatas menampilkan *coding* yang digunakan untuk mendeteksi wajah. Hal pertama yang harus dilakukan adalah menuliskan kode

import cv2, import numpy as np yang dimaksudkan bahwa disini kita menggunakan library OpenCV dan NumPy. Pada coding diatas menggunakan source yang diambil dari modul OpenCV dan menggunakan bahasa pemrograman python 2.7. Source yang ada didalam modul OpenCV merupakan file external yang berisi barisan algoritma untuk mencari wajah pada gambar atau video. File external ini berbasis XML, file ini dinamakan HaarCascade. Kemudian kita masukkan pada program, untuk mengimportnya kita menggunakan CascadeClassifier method.

Akan tetapi sebelumnya harus mengubah gambar yang telah dicapture menjadi gambar warna hitam putih, disini untuk membuat gambar atau video menjadi lebih mudah diproses menggunakan algoritma. Untuk mengubah gambar menjadi hitam putih digunakan code gray=cv2.cvtColor(img, cv2.COLOR_BGR2GRAY). Kode ini yang akan mengubah gambar menjadi hitam putih yan terletak pada variable gray, kemudian baru memasukkan code untuk deteksi wajah yang telah dijelaskan sebelumnya.

Langkah selanjutnya yaitu menerapkan kode berikut ini:

```
faces=faceDetect.detectMultiScale(gray,1.3,5)
```

Pada kode tersebut menerapkan algoritma dari face kedalam gray, yang merupakan versi hitam putih. Pada bagian belakang terdapat 2 parameter angka yang berfungsi sebagai angka keakuratan dari algoritma. Disini kita dapat mengubahnya dan bias menyesuaikan, akan tetapi umumnya digunakan angka 1.3 dan 5. Variable wajah tersebut akan menghasilkan titik X wajah dan titik Y wajah, lebar wajah dan tinggi wajah.

Untuk menggambar sebuah kotak di wajah yang terdeteksi maka digunakan For loop untuk setiap variable wajah.

```
for(x,y,w,h) in faces:
    sampleNum=sampleNum+1;
```

```
cv2.rectangle(img,(x,y),(x+w,y+h),(0,0,255),2)
```

pada setiap variable wajah yang terambil aka nada gambar kotak pada wilayah wajah tersebut. Pada koding ini digunakan rectangle method untuk menggambar kotak pada area wajah yang terdeteksi. Pada kode diatas dideklarasikan 5 parameter. Parameter yang pertama merupakan letak dimana akan menggambar yaitu pada img bukan gray karena gray hanya digunakan untuk mendeteksi muka, tetapi disini menggambar pada gambar yang berwarna atau gambar asli. Kemudian parameter keduanya yaitu tuple (x,y) yang merupakan titik kiri atas dari sebuah persegi. Parameter ketiga yaitu tuple (x+w, y+h) ini berguna untuk mendefisikan titik kanan bawah persegi, sehingga akan terbentuklah persegi dengan 2 titik. Parameter ke empat adalah tuple untuk warna OpenCV menggunakan warna BGR yaitu Blue, Green, Red dan kebalikan dari RGB. Nilai warna terkuat yaitu 255, jadi pada code diatas menggunaka gambar garis kotak persegi menggunakan warna merah. Dan parameter terakhir yaitu angka 2 yang merupakan ketebalan garis persegi.
Setelah muka terdeteksi maka akan mengcapture sebanyak 10 kali yang kemudian akan tersimpan didatabase.

```
cv2.imwrite("dataSet/user."+str(nopassport)+"."+str(nama)+str(sampleNum)
+".jpg",gray[y:y+h, x:x+w])
```

Kode diatas merupakan kode untuk menyimpan gambar yang telah dicapture. Gambar disimpan dengan bentuk file jpg.,contohnya user.1234.bella.1.jpg yang terletak pada folder dengan nama dataset.

Kemudian fungsi cv2.waitkey untuk mempertahankan window agar tetap menampilkan gambar,. Fungsi cam.release() adalah untuk memberikan kode exit ke kamera. Sedangkan fungsi cv2.destroyAllWindow() adalah untuk menutup window lain yang sedang terbuka.

1.1.2. *Coding Face Recognition*

```
import cv2,os
import numpy as np
from PIL import Image
import pickle
import sqlite3
nopassport= self.nopass_lineEdit.text()
namapenerbangan = self.namapener_lineEdit.text()
conn=sqlite3.connect("data.db")
cmd="SELECT * FROM USERS WHERE nopassport='{0}'".format(nopassport)
cursor=conn.execute(cmd)
profile =None
for row in cursor:
    profile=row
connection.close()


faceDetect=cv2.CascadeClassifier("D:\Software TA\opencv\sources\data\haarcascades\haarcascade_frontalface_default.xml");
rec=cv2.createLBPHFaceRecognizer()
rec.load("recognizer\\trainningData.yml")
path="dataSet"
cam=cv2.VideoCapture(0)
font=cv2.cv.InitFont(cv2.cv.CV_FONT_HERSHEY_COMPLEX_SMALL,2,1,0,2)
while(True):
    ret,img=cam.read();
    gray=cv2.cvtColor(img,cv2.COLOR_BGR2GRAY)
    faces=faceDetect.detectMultiScale(gray, scaleFactor=1.2, minNeighbors=5, minSize=(100, 100), flags=cv2.CASCADE_SCALE_IMAGE);
    x=0
    y=0
    for (x,y,w,h) in faces:
        cv2.rectangle(img,(x,y),(x+w,y+h),(0,0,255),2)
        nopassport,conf=rec.predict(gray[y:y+h,x:x+w])
        cv2.cv.PutText(cv2.cv.fromarray(img),"Nama:"+str(profile[1]),(x,y+h+20),font,(0, 0, 255))
    cv2.imshow("Face",img)
    cv2.waitKey(100)& 0xff
    if 120<x<500 and 100<y<380:
        cv2.imwrite("dataFoto/user."+str(nopassport)+"."+str(namapenerbangan)+".jpg",gray[y:y+h,x:x+w])
        break;
```

Gambar 4.2. *Coding Face Recognition*

Pada Gambar 4.2. Coding Face Recognition diatas merupakan coding proses pelabelan dan identifikasi wajah (*face recognition*).

faceDetect=cv2.CascadeClassifier("D:\SoftwareTA\opencv\sources\data\haarcascades\haarcascade_frontalface_default.xml)
cam=cv2.VideoCapture(1)
font=cv2.cv.InitFont(cv2.cv.CV_FONT_HERSHEY_COMPLEX_SMALL,2,1,0,2)

Dengan membuat cascadeclasifier menggunakan haar cascade untuk mendeteksi wajah. Selanjutnya adalah kode untuk membuat ojek video capture menggunakan angka 1 karna pada aplikasi ini menggunakan kamera eksternal. Sedangkan jika menggunakan angka 0 itu artinya menggunakan kamera internal atau bawaan laptop.

Selanjutnya adalah untuk menentukan font (ukuran) karena kita akan melabelkan nama pemilik wajah pada gambar sehingga kita membutuhkan font untuk teks. Parameter pada pembuatan font diatas adalah nama font, skala horizontal, skala vertical, spasi , ketebalan garis, tipe garis. Berikut ini kodenya:

pada kode ini untuk memprediksi nomor passport pengguna dan keakuratan dari prediksi masing-masing, kemudian kode selanjutnya adalah penulisan identitas nama pengguna yang terletak dibawah wajah.

```
rec=cv2.createLBPHFaceRecognizer()
rec.load("recognizer\\trainningData.yml")
```

Yang merupakan perintah untuk membuat objek recognizer menggunakan OpenCV library dan memuat data training, yang telah disimpan satu folder dengan proses training sebelumnya.

```
nopassport= self.nopass_lineEdit.text()
namapenerbangan = self.namapener_lineEdit.text()
conn=sqlite3.connect("data.db")
cmd="SELECT * FROM USERS WHERE nopassport='{0}'".format(nopassport)
    cursor=conn.execute(cmd)
    profile =None
    for row in cursor:
       profile=row
    connection.close()
```

Proses diatas dilakukan dengan memasukkan nomor passport yang telah tersimpan pada database dan pelabelan wajah sesuai dengan nomor passport yang telah tersimpan pada database. Yang kemudian akan dicapture oleh kamera dan akan disimpan lagi pada database login.

1.1.3. *Coding database sign-up*

1) *Create Database*

```
import sqlite3

def createTable():
    connection = sqlite3.connect('data.db')
    connection.execute("CREATE TABLE USERS(NOPASSPORT TEXT NOT NULL, NAMA TEXT NOT NULL, TTL TEXT, EMAIL TEXT, NOKTP TEXT, ALAMAT TEXT, FOTO TEXT)")
    connection.commit()



    connection.close
createTable()
```

Gambar 4.3. *Coding Create Database Sign-Up*

Pada gambar 4.3. diatas merupakan coding membuat database yang menyimpan data para pemilik passport saat membuat akun. Kode connection = sqlite3.connect('data.db') sebagai skrip yang terhubung ke database baru yang disebut data.db. kemudian connection.execute("CREATE TABLE USERS(NOPASSPORT TEXT NOT NULL, NAMA TEXT NOT NULL, TTL TEXT, EMAIL TEXT, NOKTP TEXT, ALAMAT TEXT, FOTO TEXT)") merupakan kode untuk perintah pembuatan tabel pada database.

2) *Insert Database*

```
def insertData(self):
    nopassport = self.nopassport_lineEdit.text()
    nama = self.uname_lineEdit.text()
    email = self.email_lineEdit.text()
    ttl = self.ttl_lineEdit.text()
    noktp = self.noktp_lineEdit.text()
    alamat = self.alamat_lineEdit.text()
    connection = sqlite3.connect("data.db")
    format_str = """INSERT INTO USERS (nopassport, nama, ttl, email, noktp, alamat)
    VALUES ("{0}", "{1}", "{2}", "{3}", "{4}", "{5}");"""
    sql = format_str.format(nopassport, nama, ttl, email, noktp, alamat)
    connection.execute(sql)
    connection.commit()
    connection.close()
```

Gambar 4.4. Coding memasukkan data *Sign-up*

Pada gambar 4.4. diatas merupakan perintah untuk mengisi atau memasukkan data baru penumpang ke database. Pada coding diatas dapat dijelaskan bahwa ada 4 jenis data yang harus diisi.

3) *Update* Foto Pada Databse

Pada skrip Gambar 4.5. dibawah ini merupakan perintah untuk memperbarui alamat foto yang tertera pada database.

```
connection = sqlite3.connect("data.db")
foto = "UPDATE USERS SET FOTO='{0}' WHERE NOPASSPORT={1}".format(namaFile,nopassport)
namaFile="dataSet/user."+str(nopassport)+"."+str(nama)+str(sampleNum)+".jpg"
cv2.imwrite(namaFile,gray[y:y+h,x:x+w])
connection.execute(foto)
connection.commit()
connection.close()
```

Gambar 4.5.. Coding Update Foto pada Proses *Sign-up*

1.1.4. Coding database Login

1) Create Database Login

```
import sqlite3

def createTable():
    connection = sqlite3.connect('login.db')
    connection.execute("CREATE TABLE USERS(NOPASSPORT TEXT NOT NULL, NAMAPENERBANGAN TEXT, FOTO TEXT)")
    connection.commit()
    connection.close
createTable()
```

Gambar 4.6. Coding Create Database Login

Pada Gambar 4.6. Coding Create Datbase Login diatas merupakan coding membuat database yang menyimpan data para pemilik passport saat membuat akun. Kode connection = sqlite3.connect('login.db') sebagai skrip yang terhubung ke database baru yang disebut login.db. Kemudian connection.execute("CREATE TABLE USERS(NOPASSPORT TEXT NOT NULL, NAMAPENERBANGAN TEXT, FOTO TEXT)") merupakan kode untuk perintah pembuatan tabel pada database.

2) *Insert Database Login*

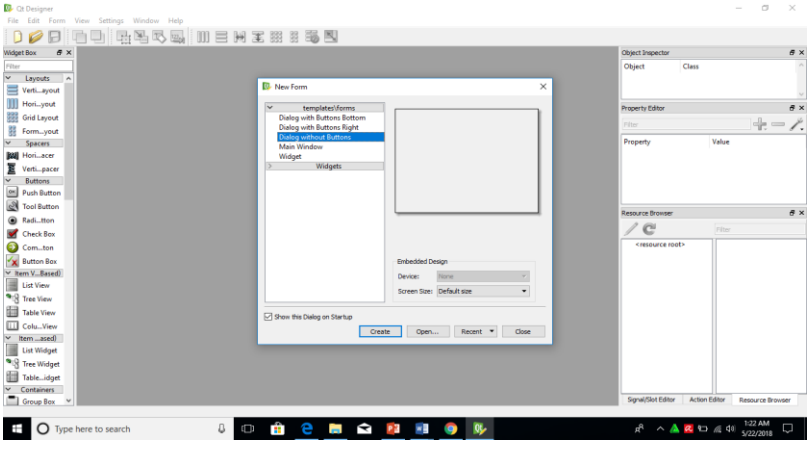
Pada gambar 4.7. coding dibawah ini merupakan perintah untuk mengisi atau memasukkan data baru penumpang yang melakukan check-in ke database. Pada coding diatas dapat dijelaskan bahwa ada 2 jenis data yang harus diisi. Yang akan tersimpan pada databse sebagai data penumpang yang melakukan Check-in.

```
def insertData(self):
    nopassport= self.nopass_lineEdit.text()
    namapenerbangan = self.namapener_lineEdit.text()
    connection = sqlite3.connect("login.db")
    login = """INSERT INTO USERS (nopassport, namapenerbangan, foto) VALUES ("{0}", "{1}", "{2}");"""
    foto = "dataFoto/user."+str(nopassport)+"."+str(namapenerbangan)+".jpg"
    sql =login.format(nopassport, namapenerbangan,foto)
    connection.execute(sql)
    connection.commit()
    connection.close()
```

Gambar 4.7. Coding Memasukkan Data Login

1.2. *Interface* Sistem

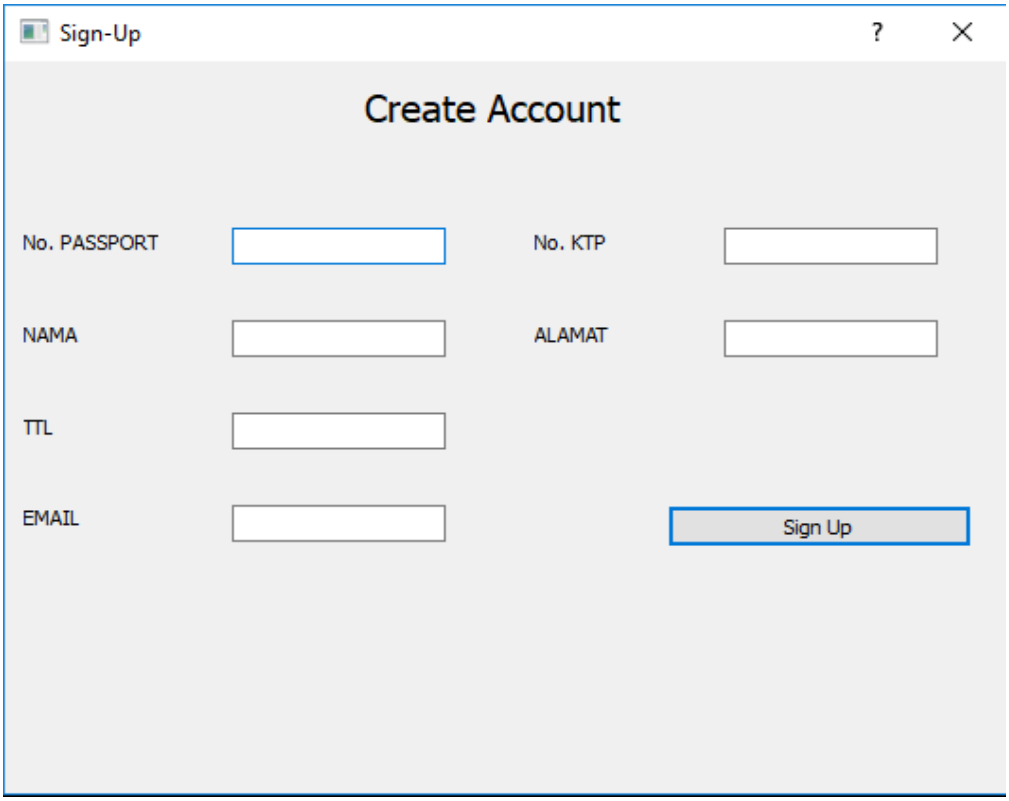
Interface sistem yang digunakan pada aplikasi ini menggunakan PyQt4 yang merupakan salah satu *library* pada python 2.7. Dengan menggunakan PyQt4maka akan memudahkan dalam membuat suatu *interface* karena pada PyQt4 berbasis GUI yaitu hanya dengan memilih item-item pada program untuk membuat tampilan untuk aplikasi ini. Berikut ini ditujukkan pada Gambar 4.8. yang merupakan gambar tampilan PyQt4 dalam membuat *interface* program:



Gambar 4.8. Tampilan PyQt4

Pada aplikasi ini memiliki beberapa tampilan form yaitu:

1. Form Tampilan Operasional Bandara



Gambar 4.9. Tampilan *Sign-up*

Pada Gambar 4.9. diatas yang merupakan gambar tampilan pada saat proses pendaftaran passport dilakukan agar bias melakukan *self check-in*. *Form sign-up* atau form daftar dilakukan untuk mendaftarkan akun agar bias melakukan *self check-in* / *login*. Bagi para pengguna yang belum memiliki akun harus melakukan proses pendaftaran akun terlebih dahulu.

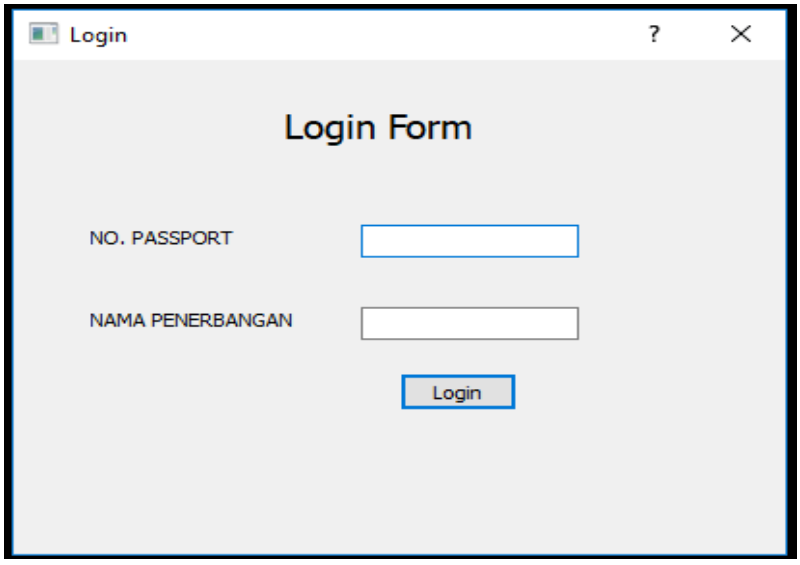
Seorang pengguna sistem memiliki akun yang terdaftar dalam sistem maka sistem akan memperbolehkan pengguna untuk melakukan self check-in atau login. Oleh sebab itu maka setiap pengguna harus memiliki dan melakukan pendaftaran agar data bias terdeteksi pada saat melakukan *self check-in* atau *login.*

2. Tampilan saat Proses Deteksi Wajah Sign-up

Gambar 4.10. Tampilan saat Proses Deteksi Wajah Sign-up

Pada Gambar 4.10. diatas merupakan tampilan kamera saat proses sign-up, yang merupakan proses deteksi wajah yang dilakukan untuk mengambil sample wajah yang akan digunakan untuk proses pengenalan wajah. Yang mana nantinya foto yang telah diambil akan disimpan di sebuah folder yang letaknya akan tersimpan di database.

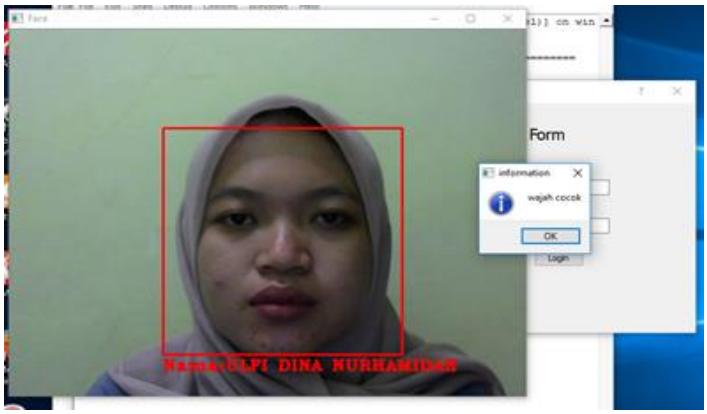
3. Tampilan *Login*



Gambar 4.11. Tampilan *Login*

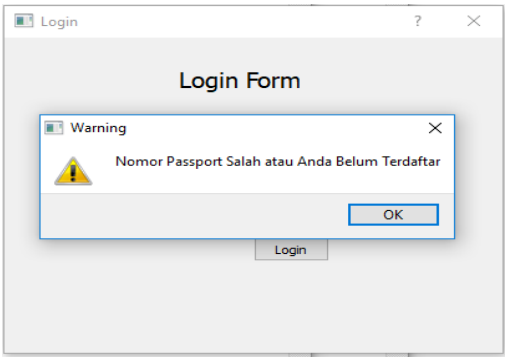
Pada Gambar 4.11. diatas merupakan tampilan login / form login. *Form login* merupakan form untuk melakukan *login* atau *self check-in*. Jadi di *form* ini para penumpang yang akan melakukan *self check-in* harus mengisi agar bisa melakukan *check-in*. Hanya pemilik passport yang memiliki akun yang dapat melakukan *self check-in* ini.

4. Tampilan Pengenalan Wajah saat Proses *Login*



Gambar 4.12. Tampilan Pengenalan Wajah saat Proses Login

Pada Gambar 4.12. diatas merupakan tampilan saat proses pengenalan saat proses login. Proses ini dilakukan setelah pengguna mengisi data pada tampilan sebelumnya kemudian menekan tombol login maka akan muncul tampilan sepertigambar diatas. Akan tetapi jika data tidak dikenali atau pengguna salah memasukkan data maka akan ada tampilan seperti ditunjukkan pada Gambar 4.13. dibawah ini:



Gambar 4.13. Tampilan Jika Data Tidak Sesuai

1.3. Pengujian dan Analisis

1.3.1. Pengujian Pengenalan Wajah

Pada proses ini proses pendeteksian wajah merupakan proses akhir dari aplikasi self check-in menggunakan metode viola-jones. Pengujian wajah ini dilakukan setelah proses pendeteksian dan cropping wajah berhasil dilakukan. Sebelum sampai pada tahap pengenalan wajah ini dilakukan sebuah tahap *trained face* dan label nama. *Trained face* ini berasal dari hasil penambahan wajah dari cropping image wajah / scale image. Sedangkan label nama diperoleh dari input database yang dilakukan sebelum proses cropping image wajah.

Dalam penelitian ini penulis melakukan uji coba sampel sekitar 20 wajah referensi pemilik yang berbeda. Untuk pengujian proses pengenalan wajah disini penulis menggunakan beberapa parameter yang menjadi factor diluar kebiasaan yang memungkinkan dapat mempengaruhi proses hasil dari pengenalan wajah. Hal ini dilakukan agar dapat menguji keakurasian dari pengenalan wajah, antara lain:

1.3.1.1. Pengujian Pengenalan Wajah berdasarkan Posisi, Ekspresi dan Aksesoris.

Pada bagian ini merupakan hasil pengujian pengenalan wajah berdasarkan posisi, ekspresi wajah dan aksesoris yang memungkinkan dipakai apabila melakukan proses pengenalan wajah ini. Berikut adalah hasilnya yang dapat kita lihat dibawah ini:

1. Pengujian pada saat wajah kondisi normal

Gambar 4.14. Pengujian pada Saat Wajah Kondisi Normal

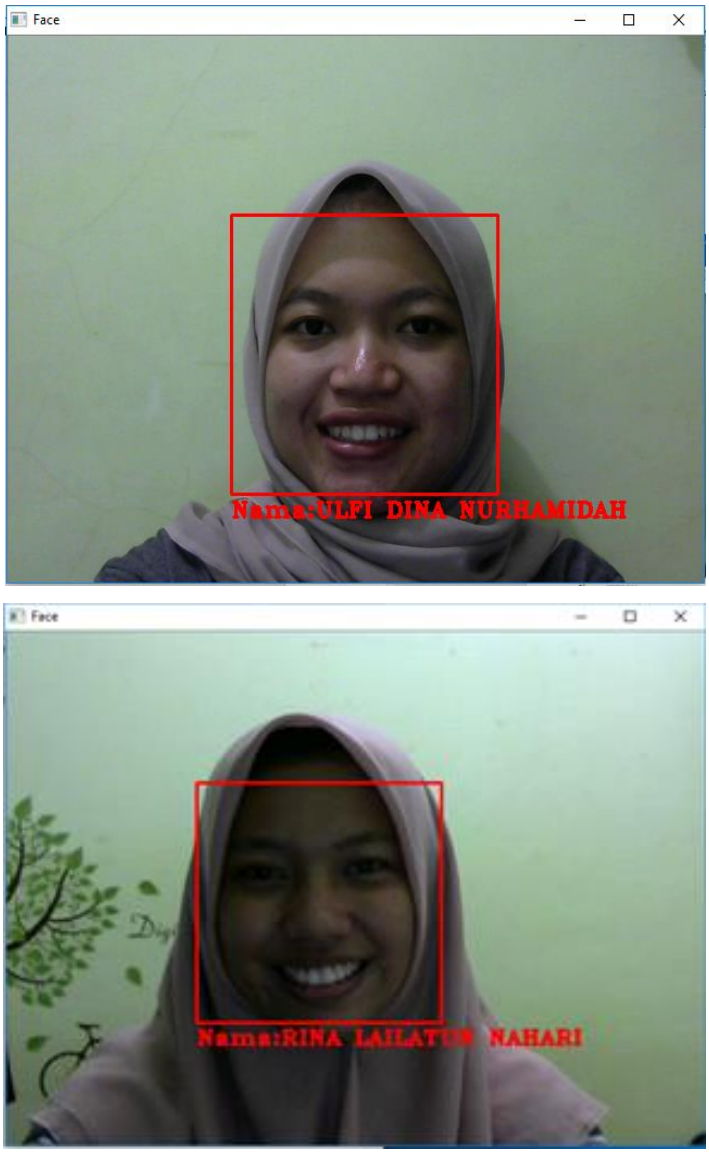
Pada Gambar 4.14. diatas merupakan hasil dari pengujian pengenalan wajah pada saat kondisi normal yaitu menunjukan bahwa aplikasi berhasil mengenali wajah dengan benar pada saat kndisi normal. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.

2. Pengujian pada saat memejamkan mata

Gambar 4.15. Pengujian saat Memejamkan Mata

Pada Gambar 4.15 diatas merupakan pengujian saat memejamkan mata, dari kedua gambar dapat dilihat bahwa wajah terdeteksi dan mampu melakukan proses pengenalan wajah. Sehingga pada proses ini aplikasi masih mampu mendeteksi dan melakukan pengenalan wajah. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.

3. Pengujian saat Kondisi Wajah Tertawa



Gambar 4.16. Pengujian saat Gaya Wajah Tertawa

Dari Gambar 4.16. merupakan hasil pengujian pengenalan wajah dari kondisi wajah saat wajah tertawa menunjukkan bahwa aplikasi berhasil mendeteksi wajah dan mengenali dengan benar wajah pada saat kondisi

tersebut. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.

4. Pengujian wajah pada saat menggunakan kacamata gelap

   Gambar 4.17. dibawah ini menunjukkan hasil pengujian wajah yang dilakukan pada saat menggunakan kaca mata hitam. Pada gambar dapat dilihat bahwa hasil yang di dapat pada kedua gambar sama. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.

Gambar 4.17. Pengujian Saat Menggunakan Kaca Mata Hitam

Pada gambar yang pertama wajah terdeteksi dan dapat melakukan pengenalan wajah. Akan tetapi pada saat percobaan dengan orang yang berbeda menunjukkan hasil bahwa muka yang terdeteksi juga dan dapat dikenali. Sehinngga dapat disimpulkan bahwa proses pengenalan wajah menggunakan kacamata bisa dilakukan, akan tetapi lebih efektifnya yaitu tidak menggunakan kacamata agar proses identifikasi oleh petugas lebih mudah dan akurat.

5. Pengenalan wajah menggunakan masker

Gambar 4.18. Pengujian saat Menggunakan Masker

Hasil pengujian pengenalan wajah pada saat menggunakan masker seperti yang ditunjukkan pada Gambar 4.18. diatas menunjukkan bahwa wajah tidak terdeteksi atau gagal mengenali wajah. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.

6. Pengujian pengenalan wajah saat wajah hadap samping dan miring kesamping.

Gambar 4.19. Pengujian Saat Wajah Hadap Samping

Gambar 4.20. Pengujian Pada Saat Wajah Miring

Dari Gambar 4.19 dan 4.20. diatas merupakan hasil pengujian posisi wajah menghadap kesamping dan posisi wajah menghadap kesamping menunjukkan bahwa aplikasi tidak dapat mengenali wajah dengan benar atau wajah tidak terdeteksi. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.

1.3.1.2. Pengujian Pengenalan Wajah berdasarkan Jarak

Pada bagian ini penguji melakukan pengujian berdasarkan jarak, disini membahas jarak minimum dan jarak maksimal aplikasi mampu mengenali wajah. Berikut adalah hasil dari pengujian yaitu:

1. Pengujian Pengenalan Wajah saat Jarak 20 cm

Gambar 4.21. Pengujian Pengenalan Wajah saat Jarak 20 cm

Pada Gambar 4.21. diatas merupakan hasil yang didapat pada saat posisi wajah dan kamera berjarak 20 cm. Hasil yang di dapat yaitu wajah tidak terdeteksi dan juga tidak dikenali. Jadi dapat disimpulkan bahwa proses deteksi wajah tidak berhasil sehingga proses pengenalan wajah tidak dapat

di proses saat wajah dan kamera berjarak 20cm. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.

2. Pengujian Pengenalan Wajah saat Jarak 30 cm

Gambar 4.22. Pengujian Pengenalan Wajah saat Jarak 30 cm

Pada gambar 4.22. diatas merupakan hasil dari proses pengujian pengenalan wajah saat wajah dan kamera berjarak 30cm. Dapat dilihat bahwa wajah terdeteksi dan dapat dikenali, tetapi pada jarak ini wajah terlihat sangat dekat dengan kamera. Sehingga menjadikan posisi ini kurang efisien untuk dilakukan saat mendeteksi wajah. Karena pengguna harus melakukan dengan posisi yang tidak nyaman. Kemudian pada

percobaan ini posisi wajah dengan kamera pada posisi tengah. Jadi dapat disimpulkan bahwa pada jarak 30cm proses ini dapat melakukan deteksi wajah dan pengenalan wajah.

3. Pengujian Pengenalan Wajah saat Jarak 60 cm

Gambar 4.23. Pengujian Pengenalan Wajah saat Jarak 60 cm

Pada gambar 4.23. diatas adalah saat pengujian pengenalan wajah saat posisi antara wajah dan kamera berjarak 60 cm, pada jarak ini wajah wajah dapat terdeteksi dan dapat dikenali. Pada jarak ini adalah posisi yang paling tepat saat melakukan deteksi wajah dan pengenalan wajah, karena jaraknya yang tidak terlalu dekat dan tidak terlalu jauh. Sehingga posisi pengguna

dalam melakukan capture wajah lebih nyaman dan foto yang dihasilkan juga lebih proposional. Yang mana pada percobaan ini posisi wajah dengan kamera pada posisi sejajar. Jadi dapat disimpulkan bahwa pada jarak 60cm antara wajah dan kamera aplikasi dapat mendeteksi dan dapat melakukan proses pengenalan wajah.

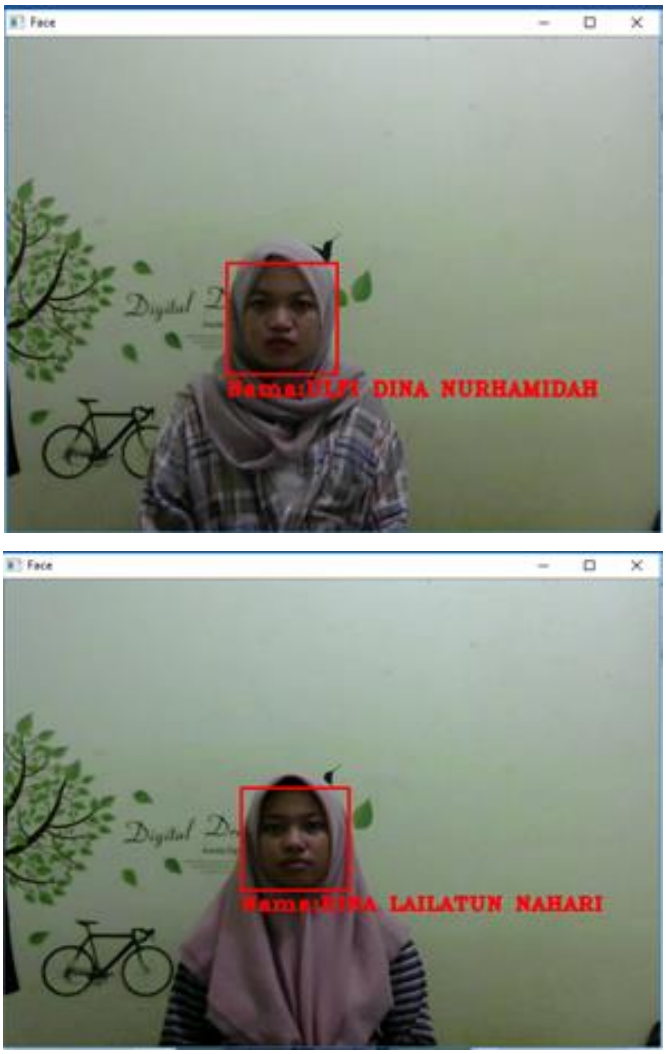
4. Pengujian Pengenalan Wajah saat Jarak 90 cm

Gambar 4.24. Pengujian Pengenalan Wajah saat Jarak 90 cm

Pada Gambar 4.24. diatas merupakan gambar hasil pengujian pengenalan wajah saat jarak 90 cm. Pada percobaan ini posisi wajah dengan kamera pada posisi sejajar. Hasil yang didapat pada saat 90 cm yaitu wajah terdeteksi dan dapat mengenali wajah. Jadi pada proses ini

dapat disimpulkan bahwa proses deteksi dan pengenalan wajah berhasil dilakukan.

5. Pengujian Pengenalan Wajah saat Jarak 150 cm



Gambar 4.25. Pengujian Pengenalan Wajah saat Jarak 150 cm

Pada Gambar 4.25. diatas merupakan gambar hasil pengujian pengenalan wajah saat jarak 150 cm. Hasil yang didapat pada saat 150 cm yaitu wajah terdeteksi dan dapat mengenali wajah. Akan tetapi pada jarak ini posisi wajah terlalu jauh dari kamera sehingga posisi ini kurang efektif

dan kurang akurat dalam mengidentifikasi wajah. Dan memungkinkan terjadinya kesalah dalam melakukan pengenalan wajah karena jarak kamera dan wajah yang jauh. Yang mana pada percobaan ini posisi wajah dengan kamera pada posisi sejajar. Jadi pada proses ini dapat disimpulkan bahwa proses deteksi dan pengenalan wajah berhasil dilakukan akan tetapi kurang efektif.

6. Pengujian Pengenalan Wajah saat Jarak 160 cm

Gambar 4.26. Pengujian Pengenalan Wajah saat Jarak 160 cm

Pada Gambar 4.26. merupakan hasil dari proses pengujian pengenalan wajah berdasarkan saat jarak 160 cm. Yang mana pada percobaan ini posisi wajah dengan kamera pada posisi sejajar. Hasil dari pengujian ini

menunjukkan bahwa wajah masih terdeteksi dan dapat melakukan pengenalan wajah. Akan tetapi jarak ini cukup jauh untuk di implementasikan pada aplikasi ini. Karena memungkinkan adanya kesalah pada proses pengenalan wajah.

7. Pengujian Pengenalan Wajah saat Jarak 170 cm

Gambar 4.27. Pengujian Pengenalan Wajah saat Jarak 170 cm

Pada Gambar 4.27. merupakan hasil dari proses pengujian pengenalan wajah berdasarkan saat jarak 170 cm. Yang mana pada percobaan ini posisi wajah dengan kamera pada posisi sejajar. Hasil dari pengujian ini

menunjukkan bahwa wajah masih terdeteksi dan dapat melakukan pengenalan wajah. Akan tetapi jarak ini cukup jauh untuk di implementasikan pada aplikasi ini. Karena memungkinkan adanya kesalah pada proses pengenalan wajah sehingga hasil yang di dapat mungkin kurang akurat begitu pula dengan hasil gambar yang di dapat tidak seakurat pada jarak yang lebih dekat.

8. Pengujian Pengenalan Wajah saat Jarak 175 cm

Dari gambar yang ditunjukkan pada Gambar 4.28. diatas merupakan hasil pengujian pengenalan wajah saat jarak 175 cm. Pada jarak ini proses deteksi wajah tidak stabil dalam melakukan pendeteksian. Kadang wajah terdeteksi dan kadang tidak terdeteksi akan tetapi lebih condong pada saat wajah tidak terdeteksi.

Gambar 4.28. Pengujian Pengenalan Wajah saat Jarak 175 cm

Yang mana pada percobaan ini posisi wajah dengan kamera pada posisi sejajar. Sehingga dapat disimpulkan bahwa pada percobaan pengujian pengenalan wajah saat jarak 175 cm wajah tidak terdeteksi sehingga tidak melakukan proses pengenalan wajah.

9. Pengujian Pengenalan Wajah saat Jarak 180 cm

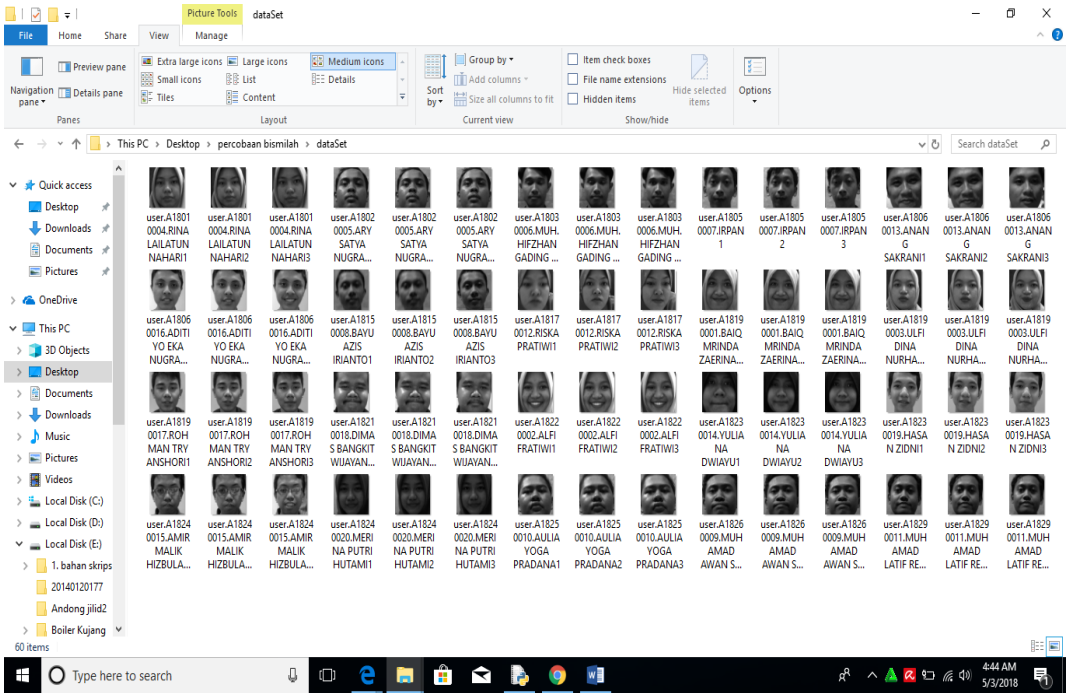
Gambar 4.29. Pengujian Pengenalan Wajah saat Jarak 180 cm

Pada gambar 4.29. diatas merupakan pengujian pengenalan wajah saat jarak 180 cm. dapat dilihat pada saat jarak 180 cm, wajah tidak terdeteksi sehingga proses pengenalan wajah tidak berhasil. Hal ini dikarenakan adalah jarak yang terlalu jauh antara kamera dan wajah. Yang mana pada percobaan ini posisi wajah dengan kamera pada posisi sejajar. Sehingga pada proses ini dapat disimpulkan bahwa pada jarak 180 cm aplikasi tidak dapat mendeteksi wajah.

1.3.2. Evaluasi Hasil Pengujian Aplikasi

Pada program yang diuji yang telah disimpan pada folder image, yaitu jumlah foto yang ikut pada pengujian berjumlah 3 citra pada saat ppendeteksian wajah dan 1 citra pada saat pengenalan wajah. Sedangkan untuk jumlah total orang yang melakukan sign-up dan check-in / login adalah 20 orang. Untuk lebih jelasnya mengenai citra referensi wajah yang telah digunakan dalampengujian ini dapatt dilihat pada gambar berikut:

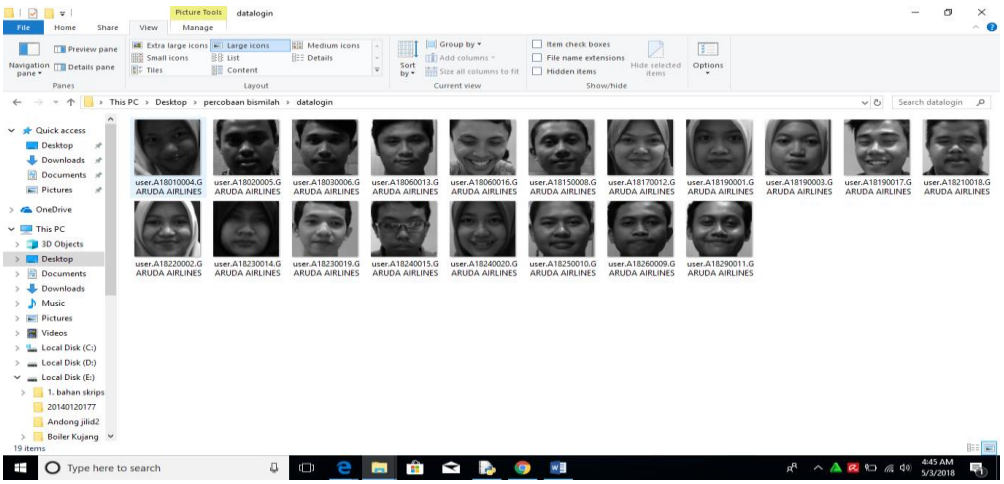
1.3.2.1. Citra Wajah Referensi Sign-up



Gambar 4.30. Wajah Referensi saat Sign-up

Pada gambar 4.30. diatas adalah merupakan gambar hasil dari proses sign-up, yang dilakukan saat melakukan pendaftaran akun. Foto yang diambil per wajah ada 3 capture, hal ini dilakukan agar foto lebih akurat. Foto ini disimpan pada sebuah folder. Sample – sample foto ini yang nantinya akan dijadikan patokan untuk proses pengenalan wajah. Jika wajah cocok dengan sample foto dan data cocok dengan yang tersimpan di database maka akan terdeteksi dan proses pengenalan wajah berhasil.

1.3.2.2. Citra Wajah Chek-In / Login

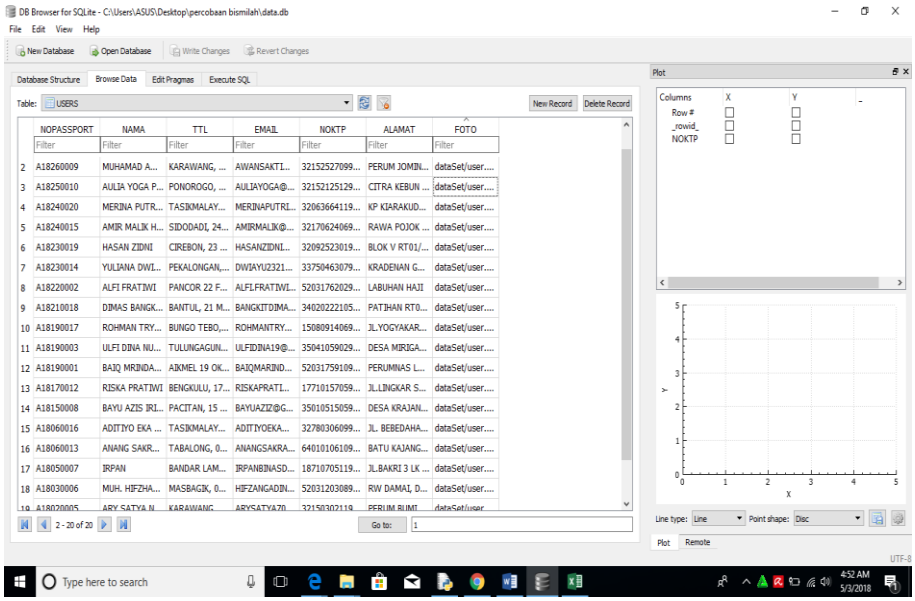


Gambar 4.31. Wajah saat Login

Pada gambar 4.30. diatas adalah hasil dari capture pada saat melakukan login/ check-in. jika gambar terdeteksi maka gambar akan disimpan di sebuah folder, gambar yang tersimpan di folder memiliki format user.nopassport.namapenerbangan. Gambar yang telah dicapture dan dsimpan jumlahnya 1. Jadi setiap pengguna yang melakukan login/check-in akan menyimpan 1 gambar wajah yang disimpan pada folder.

1.3.2.3. Database Sign-Up

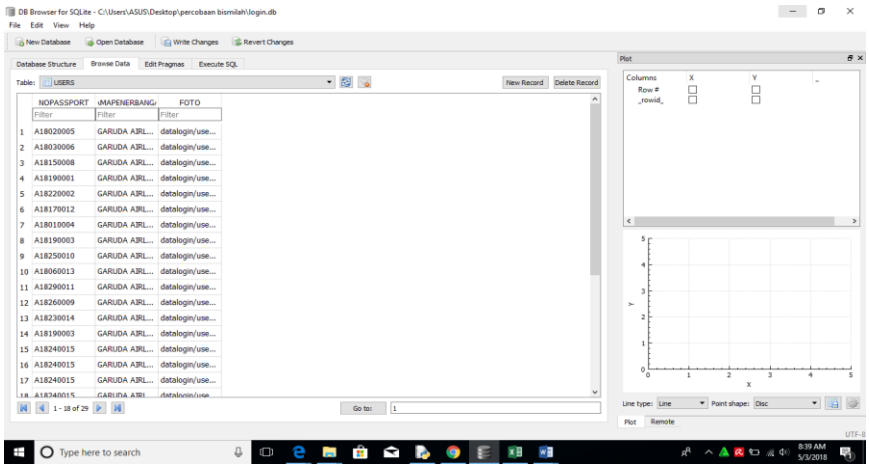
Gambar 4.31. dibawah ini merupakan database pengguna yang telah melakukan pendaftaran. Data yang tersimpan berupa nomor passport, nama, tempat/tanggal lahir, email, alamat, foto. Pada bagian foto merupakan data alamat alamat tepat foto disimpan, format foto yang tersimpan yaitu user.nopasspr Database ini sebagai acuan untukmelakukan proses login/check-in. Apabila pada saat melakukan login/check-in data masukan tidak cocok dengan dengan data yang ada didatabse ini maka proses login tidak bias dilanjutkan.



Gambar 4.32. Database Sign-up

1.3.2.4. Database Check-in / Login

Gambar dibawah ini merupakan gambar penyimpanan database pada saat proses login.



Gambar 4.33. Databse saat Login

Pada Gambar 4.33. diatas merupakan *database* proses saat sukses melakukan proses login/check-in maka data akan tesimpan di databse ini. Terdapat beberapa data seperti nomorpassport, namapenerbangan dan juga foto. Format foto yang tersimpan pada saat login ini adalah user.nopassport.namapenerbangan yang disimpan di folder tersendiri.

1.3.2.5. Hasil Pengujian Pengenalan Wajah berdasarkan Posisi, Ekspresi dan Aksesoris

Di bawah ini adalah tabel dari hasil pengujian pengenalan wajah berdasarkan posisi, ekspresi, dan aksesoris. Yang dapat dipaparkan pada tabel berikut ini:

4.1. Tabel Hasil Pengujian Pengenalan Wajah berdasarkan Posisi, Ekspresi dan Aksesoris

No.	Deskripsi Wajah	Status Hasil
1.	Pengujian pada saat wajah kondisi normal	Berhasil
2.	Pengujian pada saat memejamkan mata	Berhasil
3.	Pengujian saat wajah gaya tertawa	Berhasil
4.	Pengujian wajah pada saat menggunakan kacamata gelap	Berhasil
5.	Pengenalan wajah menggunakan masker	Gagal
6.	Pengujian pengenalan wajah saat wajah hadap samping dan miring kesamping.	Gagal

Berdasarkan tabel 4.1. diatas, setelah dilakukan 5 tipe pengujian hasilnya pada saat pengujian menggunakan masker dan saat posisi wajah miring dan hadap samping pengujiannya gagal atau wajah tidak terdeteksi. Sehingga proses pengenalan wajah tidak dapat diproses atau gagal. Sedangkan yang tipe lainnya yaitu saat kondisi normal, memejamkan mata, tertawa, dan memakai kacamata berhasil atau dapat mendeteksi wajah dan melakukan proses pengenalan wajah. Yang pada percobaan ini posisi wajah dengan kamera pada posisi tengah.

1.3.2.6. Hasil Pengujian Pengenalan Wajah berdasarkan Jarak

Di bawah ini adalah tabel dari hasil pengujian pengenalan wajah berdasarkan jarak. Yang dapat dipaparkan pada tabel berikut ini:

4.2. Tabel Hasil Pengujian Pengenalan Wajah berdasarkan Jarak

No.	Jarak	Status Hasil
1.	Pengujian Pengenalan Wajah saat Jarak 20 cm	Gagal
2.	Pengujian Pengenalan Wajah saat Jarak 30 cm	Berhasil
3.	Pengujian Pengenalan Wajah saat Jarak 60 cm	Berhasil
4.	Pengujian Pengenalan Wajah saat Jarak 90 cm	Berhasil
5.	Pengujian Pengenalan Wajah saat Jarak 150 cm	Berhasil
7.	Pengujian Pengenalan Wajah saat Jarak 160 cm	Berhasil
8.	Pengujian Pengenalan Wajah saat Jarak 170 cm	Berhasil
9.	Pengujian Pengenalan Wajah saat Jarak 175 cm	Gagal
10.	Pengujian Pengenalan Wajah saat Jarak 180 cm	Gagal

Berdasarkan tabel 4.2. diatas dalam proses pengujian berdasarkan jarak antara posisi kamera dengan wajah yaitu pada jarak 20 cm wajah belum bisa terdeteksi dan akan mulai terdeteksi sekitar jarak 30 cm, dan pada posisi efektif yaitu pada jarak 60 cm. Kemudian jarak terjauh wajah bisa terdeteksi yaitu pada jarak sekitar 150 cm , 160cm, 170 cm. Akan tetapi pada jarak 175cm dan 180 cm wajah sudah tidak terdeteksi. Jadi posisi minimum agar wajah bisa terdeteksi adalah pada jarak 30 cm dan jarak maksimum yaitu sekitar 170 cm. Yang mana pada percobaan ini posisi wajah dengan kamera pada posisi sejajar.