BAB V

PENGUJIAN SISTEM

5.1 Pengujian Model Face Recognition

Pengujian dilakukan untuk menilai kinerja model pengenalan wajah. Pengujian ini dilakukan dengan membandingkan dua gambar yang mirip, dan melihat apakah model dapat mengenali kedua gambar tersebut sebagai gambar yang sama.

Pengujian ini dapat dilakukan dengan berbagai cara. Salah satu cara yang umum digunakan adalah dengan menggunakan metrik jarak *Euclidean*. Jarak *Euclidean* adalah jarak antara dua titik dalam ruang vektor. Dalam kasus ini, dua titik tersebut adalah *embedding* dari dua gambar yang dicocokkan.

1. *Output*: Jika gambarnya cocok

```
1/1 [==============================] - 5s 5s/step
1.0
1/1 [==============================] - 0s 148ms/step
1.0
It's same, welcome in!
0.6306113
```

2. *Output:*Jika gambar nya tidak cocok

```
1/1 [==============================] - 0s 327ms/step
1.0
1/1 [==============================] - 0s 217ms/step
1.0
It's not same, please go away
1.0517054
```

3. *Output:*Jika gambar yang di *input* sama

```
1/1 [==============================] - 0s 109ms/step
1.0
1/1 [==============================] - 0s 123ms/step
1.0
It's same, welcome in!
0.0
```

Perbedaan utama antara ketiga output adalah waktu pemrosesan dan jarak *Euclidean* antar *embedding*. *Output* pertama memiliki waktu pemrosesan yang paling lama, diikuti oleh *output* kedua, dan *output* ketiga yang paling cepat. *Output* ketiga juga memiliki jarak Euclidean yang 0, yang berarti kedua gambar identik secara matematis dalam ruang embedding model.

Perbedaan waktu pemrosesan bisa disebabkan oleh faktor-faktor seperti:

1. Perbedaan hardware atau lingkungan komputasi yang digunakan.
2. Perbedaan kondisi gambar yang diproses (misalnya, ukuran gambar, tingkat noise).
3. Fluktuasi kecil dalam proses komputasi model.

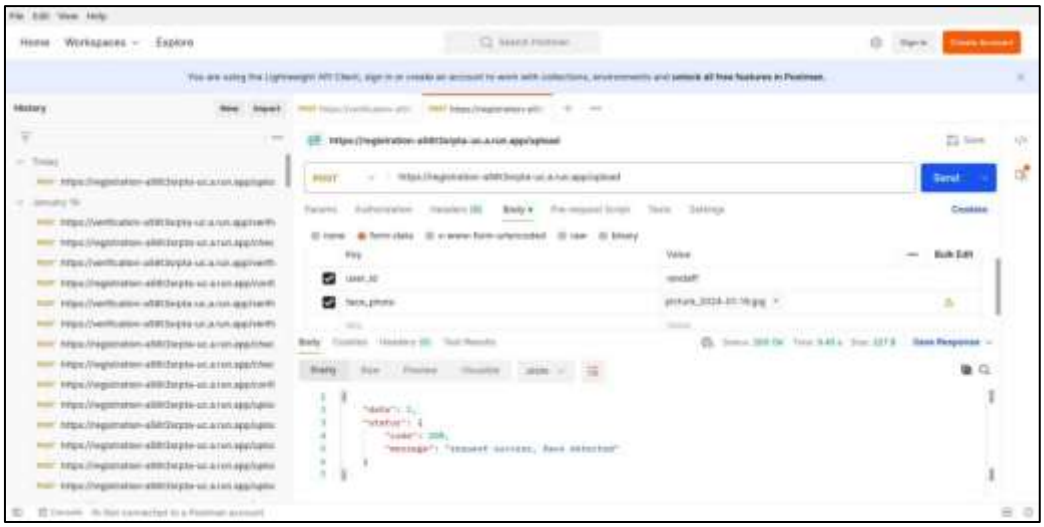
Perbedaan jarak *Euclidean* bisa disebabkan oleh faktor-faktor seperti:

1. Perbedaan tingkat kemiripan gambar.
2. Perbedaan kualitas gambar.
3. Perbedaan sumber gambar.

Secara keseluruhan, ketiga output tersebut menunjukkan bahwa model yang digunakan dapat dengan akurat mencocokkan dua gambar yang mirip.

5.2 Pengujian *API* menggunakan *Postman*

Pengujian *API* menggunakan *Postman* adalah cara yang efektif untuk menguji fungsionalitas dan kinerja *API. Postman* adalah alat yang mudah digunakan dan menyediakan berbagai fitur untuk membantu menguji *API.*



5.2.1 API Registrasi /*upload*

Gambar 5.1 Pengujian *API URL /upload*

URL BASE : https://registration-a56t3srpta-uc.a.run.app SUB

URL : */upload*

Upload Foto deskripsi : 1 *Picture per* 1 *upload*, Format *Picture* JPG & JPEG, *max upload filesize* 1 MB

Header Authorization:

Key : Authorization

Secret : "Bearer Secret"

Parameter:

Key : face_photo Key : user_id

Parameter details:

face_photo : file.jpg / file.jpeg

user_id : string / varchar(100)

Return Value:

{"data":1, "status":
{
"code":200,
"message":"request success, face detected"
}
}

{"data":0, "status":
{
"code":400,
"message":"error message"
}
}

{"data":None, "status":
{
"code":422,
"message":"wrong file extensions"
}
}

5.2.2 API Registrasi /*confirm*

Gambar 5.2 Pengujian *API URL /confirm*

Confirm deskripsi : setelah 4 foto sukses terupload, lakukan konfirmasi ke database dengan mengirimkan id_user ke sub url ini.

Header Authorization:
Key : Authorization
Secret : Bearer Secret

Parameter

Key : user_id

Parameter details

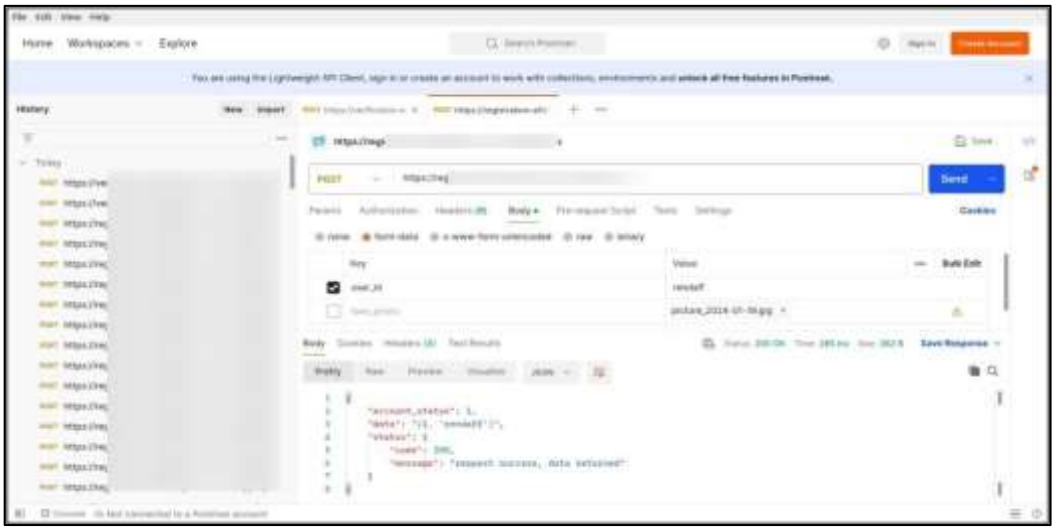
user_id : string / varchar(100)

return value:

{"data": 1,
"status": {
"code": 200,
"message": "request success, upload confirmed"
},
"confirmation_status" : 1, "user_id": user_id
}

{"data": 0,
"status": {
"code": 200,
"message": "request success, upload failed, photo face not confirmed"
},
"confirmation_status" : 0, "user_id": user_id

5.2.3 API Registrasi /*check*



Gambar 5.3 Pengujian *API URL /check*

Check deskripsi : mengambil status verifikasi apakah foto wajah user sudah di proses oleh *API* atau belum. perkiraan waktu proses verifikasi wajah adalah 5 menit setelah proses registrasi dan konfirmasi *upload. return array* berisi 2 *value* dengan *index* ke-0 adalah *user id* & *index* ke-1 adalah status verifikasi.

status verifikasi kode :

2 : akun masih dalam proses *upload* atau proses *upload* belum di confirm

1 : akun sudah di verifikasi

0 : akun belum verikasi & proses upload sudah selesai

Header Authorization

Key : Authorization

Secret : Bearer Secret

Parameter

Key : user_id

Parameter details

user_id : string / varchar(100)

return value:

{"data": (user_id, status), "status": {
"code": 200,
"message": "request success, data returne"
},
}
{"data": (),
"status": {
"code": 404,
"message": "user id not found"
},
}

5.2.4 API */verification*

Gambar 5.4 Pengujian *API URL /verification*

URL BASE : https://verification-a56t3srpta-uc.a.run.app

SUB URL : /verification

Header Authorization:
Key : Authorization
Secret : "Bearer Secret"

Parameter:
Key : face_photo Key : user_id

Parameter details:
face_photo : file.jpg / file.jpeg user_id : string / varchar(100)

Return Value:
{"data": -1 "status":
{
"code":200,

"message":"face false"
}
}

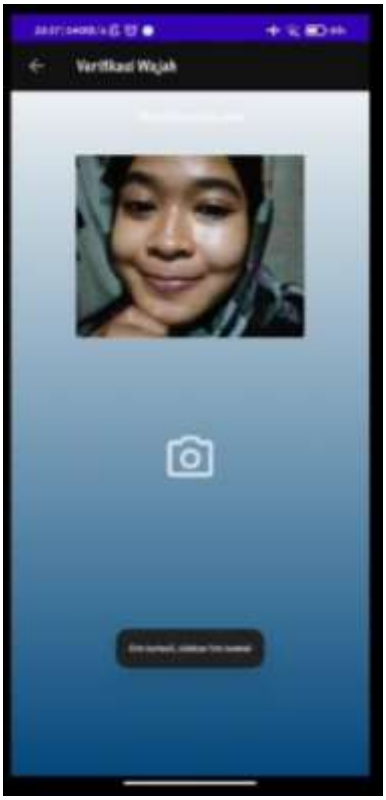
{"data": 1 "status":
{
"code":200,
"message":"face confirmed"
}
}

{"data": 0 "status":
{
"code":200,
"message":"face undetected"
}
}

{"data": diatas angka 1 "status":
{
"code":400,
"messageerror message"
}
}

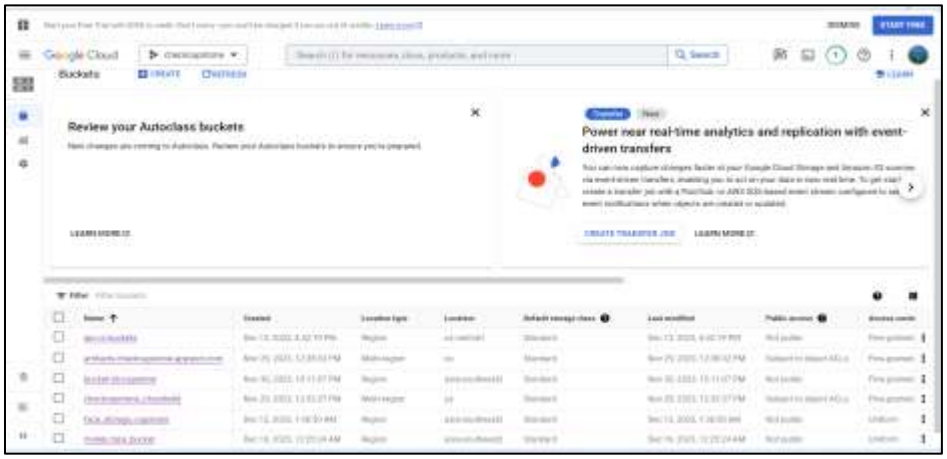
5.3 Implementasi Face Recognition GCP pada Smart Ambulance Service Center

Setelah dilakukan pengujian *API* pada *postman*, sekarang waktunya untuk melakukan pengujian pada aplikasi yang telah dibangun. Pada Gambar 5.5 tampilan implementasi *Face Recognition* pada *Smart Ambulance Service Center**:***



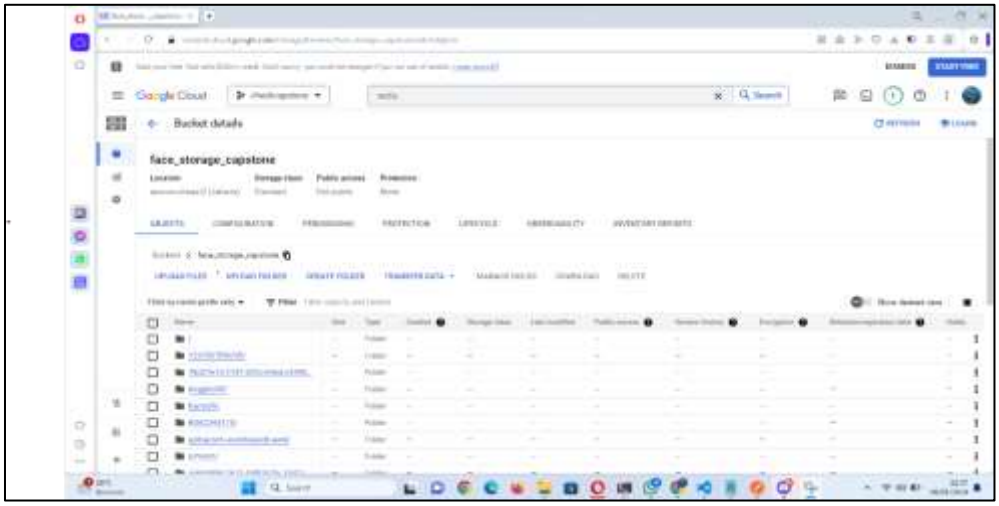
Gambar 5.5 Implementasi pada Aplikasi

Selanjutnya melihat pada *GCP Cloud Storage* apakah data gambar yang telah dimasukan tersimpan atau tidak. Pada Gambar 5.6 tampilan data pada *buckets Cloud Storage GCP*:



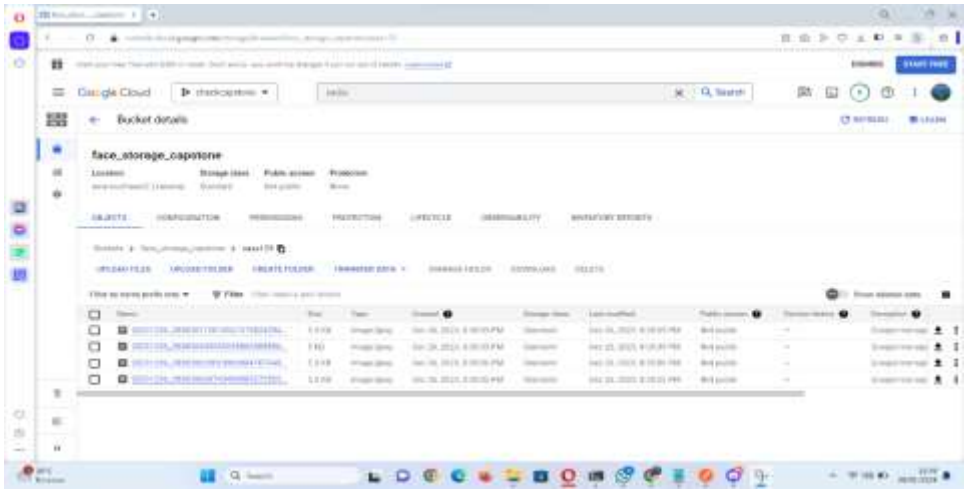
Gambar 5.6 Tampilan data *buckets Cloud Storage GCP*

Dapat dilihat ada banyak sekali *bucket* yang ada pada *Cloud Storage* Gambar 5.6**,** yang digunakan pada *face recognition* adalah *bucket* bernama *face_storage_capstone* dan *model_face_bucket.* Adapun Gambar 5.7 merupakan data yang ada pada *bucket face_storage.*



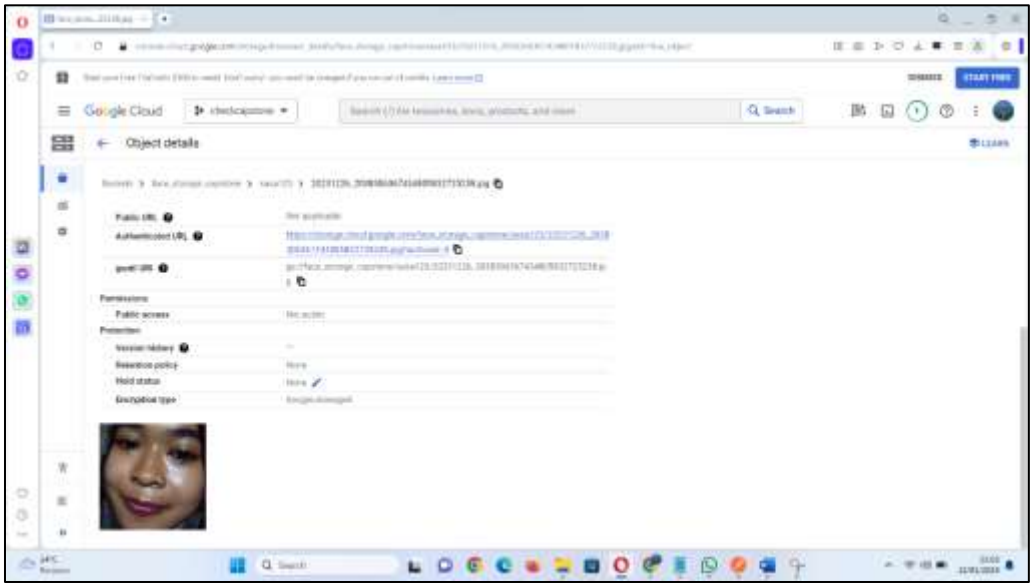
Gambar 5.7 Tampilan *face cloud storage* pada *GCP*

Ada banyak sekali data *bucket* yang telah masuk pada Gambar 5.7 akan tetapi, pada pengujian hanya mencari data yang sudah diinputkan pada Gambar 5.5. Adapun setiap data berisikan file gambar yang dapat dilihat pada Gambar 5.8 berikut:



Gambar 5.8 Tampilan *face cloud storage* berdasarkan username

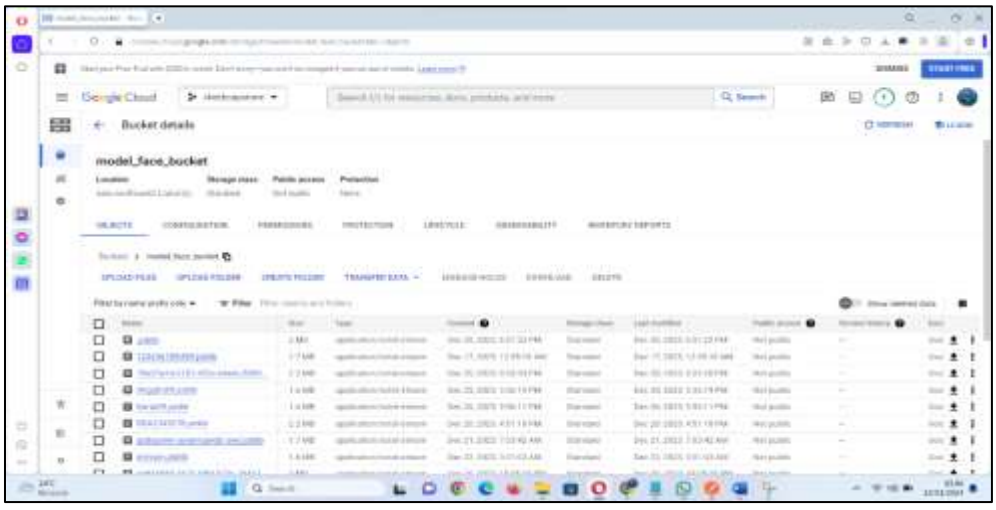
Dapat dilihat bahwa gambar-gambar pada Gambar 5.8 ekstensi file nya adalah jpeg beserta informasi-informasi lain yang terkait dengan foto. Jika kita mengklik salah satu file



gambar tersebut akan menampilkan informasi data yang dapat dilihat pada Gambar 5.9 berikut:

Gambar 5.9 Tampilan detail data berdasarkan username

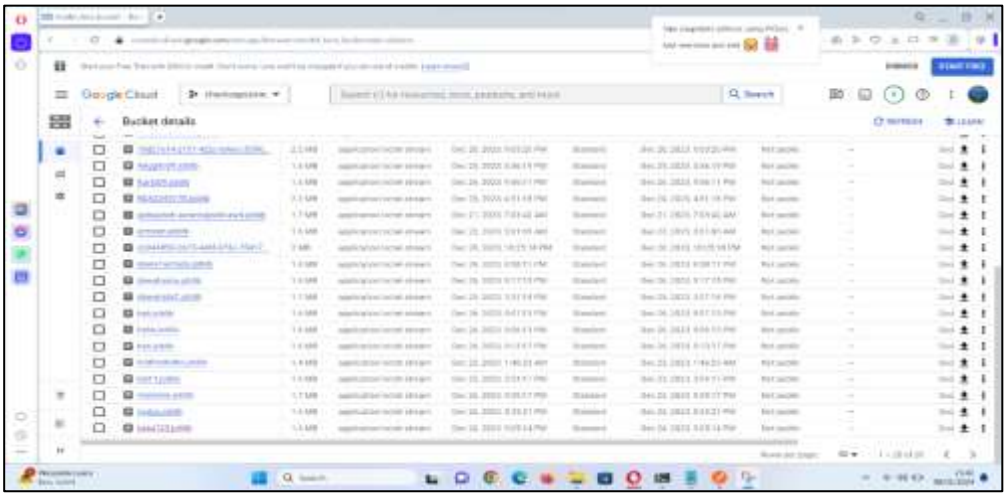
Terlihat bahwa foto telah di ekstraksi menjadi ukuran yang telah ditentukan dan bentuk wajah nya sama dengan Gambar 5.5**,** yang artinya pengujian berhasil dilakukan. Sekarang kita akan melihat model dari foto dengan menggunakan *model_face_bucket* pada *Cloud Storage*,



dapat dilihat pada Gambar 5.10 berikut:

Gambar 5.10 *Bucket model_face_bucket*

Terlihat banyak sekali model yang telah jadi dalam bentuk .joblib, sekarang kita perlu mencari model yang berkaitan dengan pengujian kita, username bernama sasa123, dapat dilihat pada Gambar 5.11 berikut:



Gambar 5.11 Tampilan model *face joblib Cloud Storage* sasa123

Terbukti bahwa pengujian implementasi *face recognition* pada *Smart Ambulance Service Center* berhasil terintegrasi dengan *GCP*.

5.4 Analisis Waktu Implementasi Face Recognition pada *Smart Ambulance Service Center*

Berdasarkan perhitungan manual, waktu yang dibutuhkan untuk registrasi dan verifikasi *face recognition* pada *Smart Ambulance Service Center* adalah sebagai berikut:

1. Registrasi: 46 detik terdiri dari 4 foto
2. Verifikasi: 14 detik terdiri dari 1 foto

Waktu tersebut diperoleh dengan cara *screen record* proses registrasi dan verifikasi, kemudian menonton ulang rekaman dan menghitung waktu yang dibutuhkan untuk setiap langkah.

Secara keseluruhan, waktu yang dibutuhkan untuk registrasi dan verifikasi *face recognition* pada *Smart Ambulance Service Center* adalah 1.14 detik. Waktu tersebut masih tergolong wajar untuk proses registrasi dan verifikasi *face recognition*.