Sistem Pengendalian Akses Berbasis *Face-Recognition* dengan Face-API.js dan Algoritma *Manhattan Distance*

Bambang Warsuta, Rizki Elisa Nalawati*, Dewi Yanti Liliana, Malisa Huzaifa, Rahma Maulida Shaliha
Teknik Informatika dan Komputer, Politeknik Negeri Jakarta
Depok, Jawa Barat
bambang.warsuta@tik.pnj.ac.id, rizkielisa@tik.pnj.ac.id, dewiyanti.liliana@tik.pnj.ac.id, malisa.huzaifa@tik.pnj.ac.id, rahma.maulidashaliha.tik19@mhsw.pnj.ac.id

Diterima: 5 Juni 2024. Disetujui: 30 September 2024. Dipublikasikan: 30 September 2024

Abstract *- This research proposes a face recognition-based access control system combining the Manhattan Distance algorithm and the face-api.js library. The primary contributions of this research include the integration of the Manhattan Distance algorithm into the facial recognition system, the utilization of face-api.js to simplify development, and a comprehensive performance evaluation. The system has successfully implemented Manhattan Distance to measure facial feature similarity. The system has been evaluated using various metrics such as accuracy, precision, and recall. Test results show good performance with accuracy scores exceeding 95% for face detection and 100% for face recognition, especially when combined with face-api.js, even with limited datasets.*

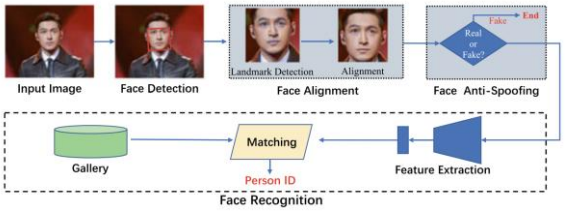
Keywords: Access Control System, Manhattan Distance, Face-API.js, Face Recognition, Convolutional Neural Network, Javascript

Abstrak**--** **Penelitian ini mengusulkan sebuah sistem pengendalian akses berbasis *face-recognition* dengan menggabungkan algoritma *Manhattan Distance* dan *library* dari *face-api.js*. Kontribusi utama penelitian ini adalah integrasi algoritma *Manhattan Distance* dalam sistem pengenalan wajah, penggunaan *face-api.js* untuk mempermudah pengembangan, serta evaluasi kinerja yang komprehensif. Sistem ini telah berhasil mengimplementasikan *Manhattan Distance* untuk mengukur kemiripan fitur wajah. Sistem ini telah dievaluasi menggunakan berbagai metrik seperti akurasi, presisi, dan *recall*. Hasil uji menunjukkan kinerja yang baik dengan nilai *akurasi* mencapai 95 ke atas untuk deteksi wajah dan 100% untuk pengenalan wajah, terutama saat dikombinasikan dengan *face-api.js*, bahkan dengan *dataset* yang terbatas.**

Kata kunci: Sistem Pengendalian Akses, Manhattan Distance, Face-API.js, Face Recognition, Convolutional Neural Network, Javascript

I. PENDAHULUAN

Face recognition bertujuan mengidentifikasi atau memverifikasi identitas seseorang melalui gambar atau video wajah untuk kebutuhan otentikasi biometrik, keamanan finansial, pengendalian akses, dan pengawasan cerdas [1] [2]. Pada konteks pengendalian akses, pengenalan wajah digunakan untuk mengotorisasi akses seseorang ke informasi pribadi yang aman untuk masuk ke lokasi yang ditentukan [3]. Sehingga, teknik ini dapat terintegrasi dalam sistem pengendalian otomatis di berbagai organisasi atau fasilitas untuk memberikan kontrol yang lebih kuat [4].



Gambar 1. Alur kerja *face recognition* [1]

Alur kerja *face recognition*, secara umum dapat dilihat pada gambar 1. Tahapan dimulai dengan *input image, face detection* sampai dengan melakukan ekstraksi dan mencocokan antara *dataset* di-*gallery* dengan hasil ekstraksi [1].

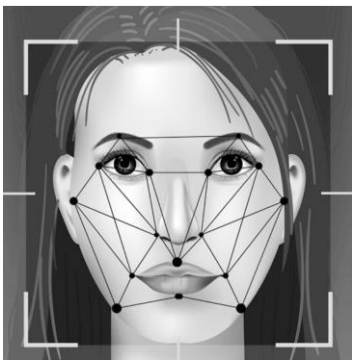
Pada tahapan *face recognition,* tingkat akurasi bergantung dari kemampuan algoritma untuk mengenali individu yang sudah dikenal. Semua bentuk pengenalan biometrik seperti *face recognition* memiliki tingkat kesalahan yang memengaruhi akurasi metode dan kinerja keseluruhan antara *dataset* yang dihitung dan sistem yang digunakan di dunia nyata [3]. Kondisi tersebut menjadi tantangan yang perlu diatasi, seperti ketidaksejajaran, variasi pose, variasi pencahayaan, dan variasi ekspresi [5] [6].

Berbagai metode *distance measure* dapat diimplementasikan untuk mengukur kesamaan antara dua objek pada proses *face recognition*, diantaranya *Euclidean Distance*, *Manhattan Distance,* dan *Minkowski Distance* [7]. Setiap pendekatan tersebut memiliki kelebihan dan kekurangan. Pada *Euclidean Distance* kelebihannya adalah mudah untuk diimplementasikan dan diuji. Berbeda dengan *Manhattan Distance,* kelebihannya adalah mudah diterapkan secara umum ke terhadap dimensi yang lebih tinggi sedangkan kekurangan dari keduanya adalah tidak berfungsi dengan baik untuk data gambar dan klasifikasi dokumen [7].

Pada penelitian ini, *Manhattan Distance* dipilih sebagai bentuk kontribusi untuk memperkaya pengetahuan dalam penggunaan metode *distance measure* pada sistem pengendalian akses yang dibuat dan dikombinasikan dengan model *face-api.js* untuk mengotorisasi individu melalui pengenalan wajah. Model *face-api.js* merupakan salah satu alat yang baik untuk mendeteksi wajah dan dapat digunakan dalam pengembangan sistem [8].

II. METODE PENELITIAN

Algoritma *face recognition* dapat didefinisikan sebagai sebuah proses atau seperangkat aturan yang diikuti untuk menghitung atau menganalisis karakteristik wajah, terutama oleh komputer. Proses pengenalan pola kemudian menyusun struktur wajah keseluruhan, menggunakan jarak antara pusat pupil mata, hidung, mulut, dan tepi rahang termasuk dagu [9]. Pengukuran-pengukuran ini, yang dihitung oleh algoritma *face recognition* dan diilustrasikan pada gambar 2, disimpan dalam sebuah basis data dan digunakan sebagai perbandingan ketika seseorang berdiri di depan kamera [2] [10].



Gambar 2. Gambaran algoritma *face-recognition* untuk mengenal pola dan jarak pada struktur ajah [3]

Seperti yang sudah dibahas sebelumnya, bahwa pada penelitian ini akan memanfaatkan algoritma *Manhattan Distance*. Algoritma khusus tersebut digunakan untuk melakukan pengukuran pola jarak pada struktur wajah yang dikombinasikan dengan model *face-api.js* untuk mengotorisasi wajah pada sistem pengendalian akses yang dibuat [11].

Face-api.js menyediakan sejumlah model *machine learning* yang berbasis *JavaScript* dan *Tensorflow.js* . Dengan *library* ini, kita dapat membangun aplikasi web yang mampu melakukan berbagai tugas pengolahan citra wajah, seperti deteksi wajah, pelacakan titik-titik wajah, dan pengenalan wajah [8].

Tahapan pengembangan dari sistem pengendalian akses adalah sebagai berikut.

1. Studi Pustaka

Sebelum memulai pengembangan sistem berbasis web untuk *face recognition,* dilakukan studi pustaka. Pada tahap ini, dianalisis penelitian-penelitian sebelumnya yang terkait dengan teknik pengenalan wajah. Selain itu, juga diperdalam pemahaman yang diperlukan untuk mengembangkan proyek yang

mengimplementasikan model *machine learning* dan *deep learning* ke dalam program.

2. Penetapan Model

Pengembangan sistem berbasis web untuk *face recognition* berbasis web ini menggunakan model dari *library face-api.js*. Pemilihan model ini didasarkan pada kesesuaian dengan kebutuhan, kompatibilitas bahasa pemrograman, dan akurasi pengenalan wajah yang tinggi, bahkan dengan data yang minim. Pada tahapan ini, ditentukan model yang akan diterapkan. Model yang ditetapkan mengacu pada *library face-api.js.* Model tersebut dipilih karena cocok dengan kebutuhan dan menggunakan bahasa pemrograman *JavaScript* sesuai dengan yang digunakan pada pengembangan sistem *face recognition* berbasis web. Selain itu, model ini tercatat memiliki nilai akurasi yang sangat tinggi tanpa memerlukan data yang banyak.

3. Pengumpulan dan Pengaturan *Dataset*

Setelah memilih model *face-api.js*, langkah selanjutnya adalah mengevaluasi akurasinya dengan menggunakan dataset gambar. Dataset ini dibagi sesuai kebutuhan untuk memastikan model bekerja dengan baik pada berbagai situasi.

4. Uji Model

Pengujian dilakukan untuk memastikan model *face-api.js* akurat dan menentukan berapa banyak data wajah yang diperlukan agar sistem dapat mengenali pengguna dengan baik.

5. Analisis Kebutuhan Sistem

Tahap ini fokus pada analisis kebutuhan sistem untuk menentukan fitur dan metode *distance measure* yang akan dikombinasikan dengan model *face-api.js*. Selain itu, *tools* yang akan digunakan selama pengembangan juga ditetapkan pada tahap ini.

6. Rancangan Antarmuka sistem

Setelah menentukan kebutuhan aplikasi, langkah selanjutnya adalah mendesainnya. Desain ini berupa *wireframe* untuk merancang tampilan antarmuka setiap halaman aplikasi.

7. Implementasi dan Pengujian

Pada tahap pengembangan, sistem dibangun sesuai dengan desain yang telah dibuat sebelumnya, baik untuk bagian *front-end* maupun *back-end*. Pengujian dilakukan secara berkala untuk memastikan setiap fitur berfungsi dengan baik dan menghasilkan luaran yang sesuai dengan ekspektasi.

III. HASIL DAN PEMBAHASAN

Pada bagian ini, dijelaskan hasil dari perancangan sistem pengendalian akses dengan menggunakan *face recognition*, mulai dari analisis kebutuhan sistem sampai dengan implemetasi dan pengujian sistem.

1. Analisis Kebutuhan Sistem

Analisis kebutuhan dilakukan untuk menentukan fitur dan teknologi yang diperlukan dalam pengembangan sistem *face recognition* berbasis web. Berikut adalah kebutuhan fungsional sistem:

a) Fitur Pengguna: Akses ruangan tanpa login menggunakan *face recognition.*

b) Fitur Admin: Login ke halaman *dashboard*, Mengelola data user (lihat, tambah, ubah, hapus), Mengelola data ruangan (lihat, tambah, ubah, hapus), Melihat laporan akses ruangan, Logout.

Teknologi yang akan digunakan untuk membangun sistem ini meliputi:

a) Front-end: HTML, CSS, JavaScript.

b) Back-end: Python, Flask, PostgreSQL.

c) Model Face Recognition: Face-api.js.

Berbagai alat lainnya digunakan selama proses pengembangan sistem. Figma untuk mendesain *mockup* antarmuka sistem. Pada tahap implementasi, *Visual Studio Code* digunakan untuk menulis kode, Postman untuk memeriksa API, dan Github untuk mencadangkan kode program. Selain perangkat lunak, diperlukan juga perangkat keras berupa laptop dan *webcam*.

Sistem ini menggunakan tiga model *machine learning* untuk fitur pengenalan wajah, yakni: SSD Mobilenet V1 untuk mendeteksi wajah dan menemukan lokasi *bounding box*, *face landmark 68* untuk mengekstrak *keypoint* dari gambar wajah yang telah didapat, dan *face*

recognition untuk mengubah *data keypoints* menjadi *descriptor*. Ketiga model tersebut berasal dari *library face-api.js* dan menggunakan algoritma *Convolutional Neural Network (CNN)* [12].

Proses pengenalan wajahnya sebagai berikut:

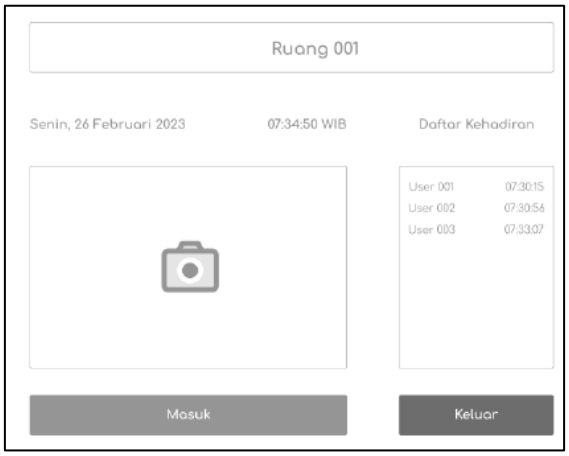
a) Sistem mengekstrak *descriptor* wajah pengguna dengan menggunakan ketiga model machine learning yang disebutkan sebelumnya.
b) *Descriptor* yang diperoleh kemudian dibandingkan dengan seluruh *descriptor* yang terdapat pada basis data menggunakan algoritma *Manhattan Distance*.
c) Pengguna akan dikenali jika memiliki jarak perbandingan terkecil dalam ambang batas tertentu.
d) Jika tidak ada kecocokan di bawah ambang batas yang telah ditentukan, maka aplikasi akan menyatakan bahwa pengguna tidak dikenali.

2. Rancangan Antarmuka Sistem

Wireframe ini merupakan rancangan awal antarmuka sistem *face recognition* berbasis web yang berfokus pada tata letak dan fungsi, tanpa gaya, detail, atau bahkan warna [13]. Tujuan pembuatan *wireframe* ini adalah untuk memberikan gambaran awal tentang tampilan *front-end* yang akan dibuat [14]. Berikut beberapa tampilan desain *wireframe*, seperti pada gambar 3 menunjukkan desain halaman untuk memilih ruangan yang akan diakses oleh penguna dengan menggunakan *face recognition*.

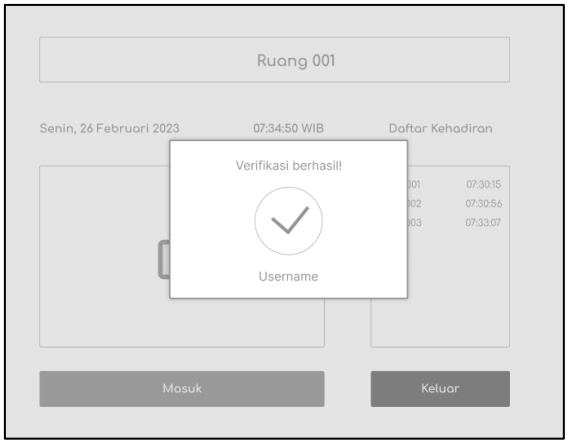
Gambar 3. *Wireframe* halaman memilih ruangan yang akan diakses dengan fitur *face recognition*

Gambar 4 menunjukkan *wireframe* halaman *face recognition* yang akan memverifikasi pengguna dan otorisasi terkait ruangan yang dapat diakses.



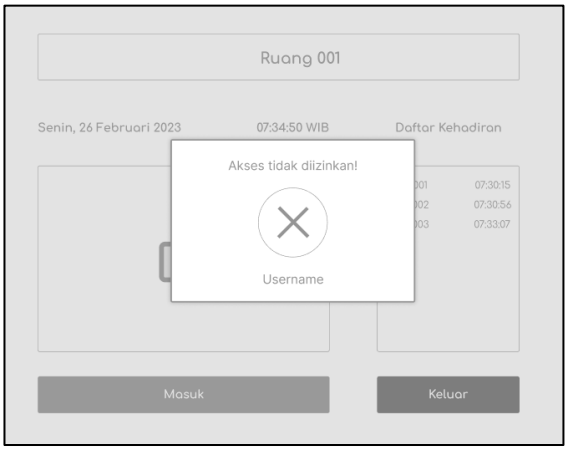
Gambar 4. *Wireframe* halaman *face recognition*

Gambar 5 menunjukkan *wireframe* halaman *face recognition* yang berhasil memverifikasi pengguna.



Gambar 5. *Wireframe* halaman *face recognition* untuk verifikasi berhasil

Gambar 6 menunjukkan *wireframe* halaman *face recognition* apabila wajah pengguna yang mencoba mengakses ruangan dikenali namun tidak memiliki izin akses ke dalam ruangan.



Gambar 6. *Wireframe* halaman *face recognition* untuk verifikasi gagal dan akses tidak diizinkan

3. Implementasi Sistem

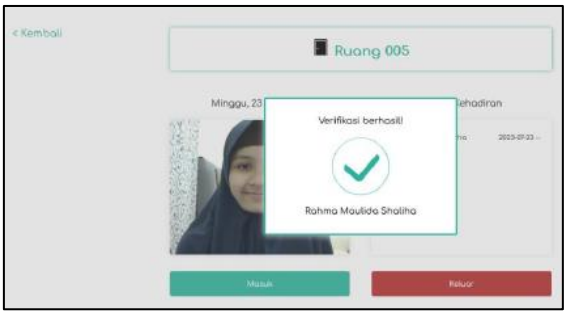
Pada bagian ini dijelaskan hasil implemengasi *front-end* dan *back-end* sistem pengendalian akses serta hasil uji model yang telah dilakukan.

a) Front-end

Front-end merupakan tampilan aplikasi yang dapat dilihat oleh pengguna sekaligus bertugas menyajikan data. Melalui tampilan tersebut, pengguna dapat memperoleh informasi yang dibutuhkan serta menggunakan setiap fungsi dari aplikasi. Berikut ini merupakan tampilan antarmuka dari halaman utama pada aplikasi *face recognition* sistem pengendalian akses berbasis web yang bisa dilihat di gambar 7.

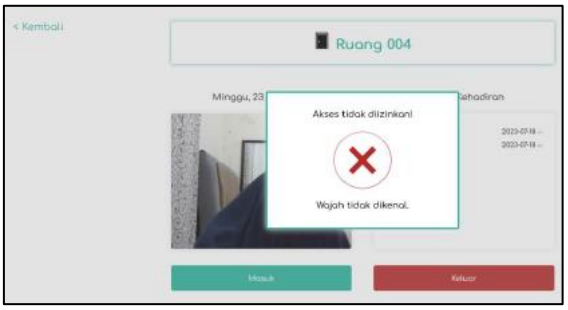
Gambar 7. Halaman memilih ruangan yang akan diakses dengan fitur *face recognition*

Pada gambar 8. *merupakan* tampilan pemberitahuan yang akan muncul ketika wajah pengguna berhasil dikenali dan diverifikasi oleh sistem ketika mengakses ruangan.



Gambar 8. Halaman *face recognition* untuk verifikasi berhasil

Gambar 9 merupakan tampilan pemberitahuan yang akan muncul ketika wajah pengguna yang mencoba mengakses ruangan dikenali namun tidak memiliki izin akses ke dalam ruangan.



Gambar 9. Halaman *face recognition* untuk verifikasi gagal dan akses tidak diizinkan

b) Back-end

Back-end bertanggung jawab untuk mengambil gambar dari basis data untuk mendukung proses pengenalan wajah [15]. Komunikasi antara *back-end* dan *front-end* difasilitasi oleh API (*Application Programming Interface*), yang bertindak sebagai penghubung untuk pertukaran data.

```
18  app.use('/users/', userRouter)
19  app.use('/rooms/', roomRouter)
20  app.use('/attendance/', attendanceRouter)
21  app.use('/permissions', permissionRouter)
22  app.use('/temp/', tempRouter)
23  app.use('/summary/', summaryRouter)
```

Gambar 10. Rute API

Gambar 10 ini mencantumkan semua rute API yang tersedia dalam aplikasi pengenalan

wajah berbasis web. Setiap rute memiliki tujuan yang berbeda-beda. Tidak semua rute terhubung langsung dengan bagian tampilan aplikasi. Beberapa rute dirancang khusus untuk digunakan oleh pengembang atau administrator untuk keperluan internal, seperti pengelolaan data dan *debugging.*

4. Pengujian Sistem

a) Uji Fungsionalitas

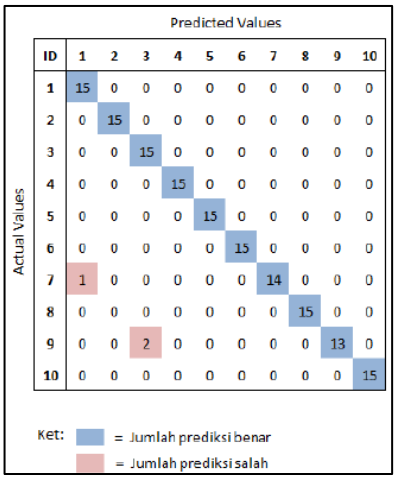
Pengujian fungsionalitas aplikasi dilakukan dengan menggunakan *black-box* untuk memverifikasi apakah setiap fitur dari sistem memenuhi persyaratan fungsional yang telah ditentukan. Berdasarkan hasil pengujian menyeluruh pada semua fitur sistem, dapat disimpulkan bahwa sistem pengendalian akses dengan *face-recognition* berbasis web ini telah memenuhi semua persyaratan fungsional. Alur program yang telah dirancang juga berjalan dengan baik sehingga menghasilkan luaran yang sesuai dengan masukan yang diberikan. Berdasarkan perhitungan keberhasilan pengujian, aplikasi ini dinyatakan berhasil 100%.

b) Uji Model

Untuk mengukur seberapa baik model ini mengenali wajah, performa model deteksi wajah dievaluasi menggunakan *confusion matrix*. Model ini mengunakan 300 foto yang ada wajahnya dan 50 foto tanpa wajah sebagai *input*. Proses deteksi wajah terhadap masing-masing foto tersebut dilakukan menggunakan model *face detection* dari *libary face-api.js*. Lalu jumlah hasil deteksi dicatat dalam matriks. Proses evaluasi dilakukan pada 10 identitas yang dipilih dari *dataset* CelebA. Setiap identitas memiliki 15 sampel gambar wajah. Pengujian dilakukan sebanyak 5 (lima) kali dengan 1-5 foto sebagai *descriptor* acuan. Berikut ini hasil merupakan *confusion matrix* untuk masing-masing jumlah *descriptor* acuan yang ditentukan.

- *Manhattan Distance* - 1 *Descriptor*

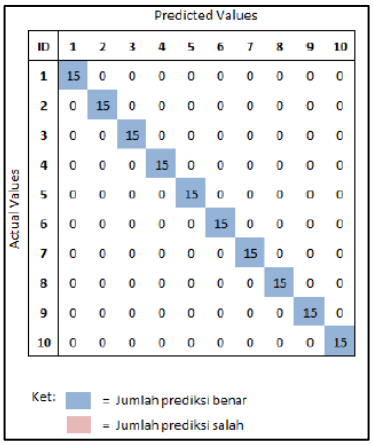
Evaluasi pada gambar 11 menunjukkan tingkat keberhasilan rekognisi 100%. Meskipun terdapat kesalahan pengenalan pada dua id (identitas), secara keseluruhan sistem menunjukkan kinerja yang memuaskan.



Gambar 11. Matriks nilai uji dengan 1 descriptor

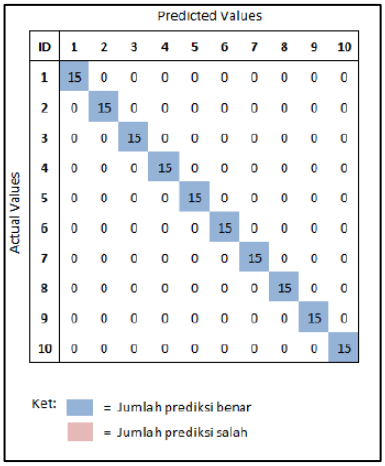
Gambar 12, 13, 14, dan 15 menunjukkan bahwa 15 dari 15 id terekognisi dengan baik 100%.

- *Manhattan Distance* - 2 *Descriptor*



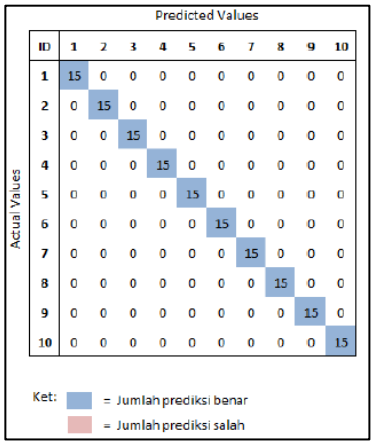
Gambar 12. Matriks nilai uji dengan 2 *descriptor*

- *Manhattan Distance* - 3 *Descriptor*



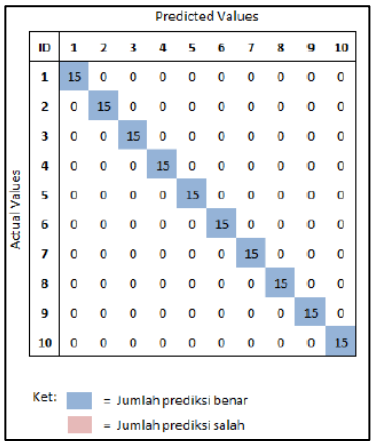
Gambar 13. Matriks nilai uji dengan 3 *descriptor*

- *Manhattan Distance* – 4 *Descriptor*



Gambar 14. Matriks nilai uji dengan 4 *descriptor*

- *Manhattan Distance* – 5 *Descriptor*



Gambar 15. Matriks nilai uji dengan 5 *descriptor*

Berikut ini merupakan catatan *execution time* yang diperlukan selama proses pencocokan dari 5 *descriptor*. Tabel 1 menunjukkan rata-rata waktu eksekusi kurang dari 0,7 ms. Hal ini berarti proses pencocokan *descriptor* cepat.

Tabel 1. *Perbandingan Waktu Eksekusi Pada 5 Descriptor*

Jumlah *descriptor* Acuan	***Execution Time (ms)***		
	Min	**Max**	**Average**
1	0	3	0,61
2	0	4	0,69
3	0	2	0,61
4	0	3	0,64
5	0	4	0,65

Nilai dari matriks performa evaluasi akurasi dan presisi yang pada tabel 2. Data menunjukkan hasil sebesar 95% ke atas untuk model *face detection* dan 100% untuk model *face recognition* yang menunjukkan kinerja memuaskan dari model deteksi wajah.

Tabel 2. Matriks Performa Evaluasi dan Presisi Sistem

Performa	Persentase
Akurasi	95,71%
Presisi	99,65%

Hal ini mengindikasikan bahwa model telah berhasil menggeneralisasi dengan baik pada data uji. Lalu, didapatkan juga jumlah yang data *descriptor* yang optimal untuk aplikasi *face recognition* yang dibuat, yakni sebanyak 2 *descriptor* untuk tiap pengguna.

IV. KESIMPULAN DAN SARAN

Penelitian ini menunjukkan bahwa sistem pengendalian akses dengan *face-recognition* berbasis web yang dikembangkan menggunakan JavaScript telah berhasil diimplementasikan. Fitur inti *face-recognition*, yakni pengenalan wajah, telah diuji dan terbukti efektif meskipun dengan jumlah

data *traning* yang terbatas. Model deteksi wajah dan pengenalan wajah dari *library face-api.js* yang digunakan dalam sistem ini menunjukkan kinerja yang sangat baik, dengan akurasi mencapai 95,71% untuk deteksi wajah dan 100% untuk pengenalan wajah. Hasil penelitian ini juga menunjukkan bahwa jumlah data deskriptor wajah yang optimal untuk aplikasi pengenalan wajah ini adalah 2 per pengguna. Temuan ini mengindikasikan bahwa algoritma jarak *Manhattan* efektif dalam menghitung kemiripan antar *descriptor* wajah dan cepat dari sisi waktu eksekusi yang menunjukkan waktu rata-rata di bawah 0,7ms sehingga cocok diterapkan dalam sistem yang dikembangkan. Namun, untuk meningkatkan keandalan sistem, disarankan untuk melakukan evaluasi lebih lanjut pada algoritma jarak dengan *dataset* yang lebih besar. Sebagai rekomendasi untuk sistem pengendalian akses ruangan, kombinasi antara kartu pintar dan pengenalan wajah dapat menjadi solusi yang lebih aman. Selain itu, pengembangan fitur deteksi *spoofing* sangat penting untuk mencegah penyalahgunaan sistem.

V. REFERENSI

[1] G. Tan Zichang and Guo, "Face Recognition Research and Development," in *Handbook of Face Recognition*, A. K. and D. J. Li Stan Z. and Jain, Ed., Cham: Springer International Publishing, 2024, pp. 3–36. doi: 10.1007/978-3-031-43567-6_1.

[2] L. Alzubaidi *et al.*, "Review of deep learning: concepts, CNN architectures, challenges, applications, future directions," *J Big Data*, vol. 8, no. 1, Dec. 2021, doi: 10.1186/s40537-021-00444-8.

[3] I. Berle, *Face Recognition Technology: Compulsory Visibility and Its Impact on Privacy and the confidentiality of personal Identifiable Images*. doi: https://doi.org/10.1007/978-3-030-36887-6.

[4] R. Robin, F. Hermanto, and W. Chandra, "Face Recognition Implementation as an Attendance Feature on Web-Based Video Conference Application," *Brilliance: Research of Artificial Intelligence*, vol. 3, no. 2, pp. 282–289, Nov. 2023, doi: 10.47709/brilliance.v3i2.3216.

[5] K. H. Teoh, R. C. Ismail, S. Z. M. Naziri, R. Hussin, M. N. M. Isa, and M. S. S. M. Basir, "Face Recognition and Identification using Deep Learning Approach," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Mar. 2021. doi: 10.1088/1742-6596/1755/1/012006.

[6] Q. Meng, S. Zhao, Z. Huang, and F. Zhou, "MagFace: A Universal Representation for Face Recognition and Quality Assessment," Mar. 2021, [Online]. Available: http://arxiv.org/abs/2103.06627

[7] S. Pandit and S. Gupta, "A Comparative Study on Distance Measuring Approaches for Clustering," 2011. doi: 10.7815/ijorcs.21.2011.011.

[8] C. Juliandy, N. Poi Wong, and Darwin, "Modeling Face Detection Application Using Convolutional Neural Network and Face-API for Effective and Efficient Online Attendance Tracking," *Jurnal Online Informatika*, vol. 9, no. 1, pp. 10–17, Apr. 2024, doi: 10.15575/join.v9i1.1203.

[9] P. Payal and M. M. Goyani, "A comprehensive study on face recognition: methods and challenges," Feb. 17, 2020, *Taylor and Francis Ltd.* doi: 10.1080/13682199.2020.1738741.

[10] H. Wu, Y. Cao, H. Wei, and Z. Tian, "Face Recognition Based on Haar like and Euclidean Distance," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Feb. 2021. doi: 10.1088/1742-6596/1813/1/012036.

[11] "Face-API.js." Accessed: Jul. 26, 2024. [Online]. Available: justadudewhohacks.github.io/face-api.js

[12] R. Elisa Nalawati, R. Maulida Shaliha, and M. Danil, "Face Recognition sebagai Control Access Area dengan Face-Api.js dan Euclidean Distance," *Mahyu Danil INNOVATIVE: Journal Of Social Science Research*, vol. 4, pp. 1848–1864, 2024.

[13] F. Staiano, *Designing and prototyping interfaces with Figma: learn essential UX/UI design principles by creating interactive prototypes for mobile, tablet, and desktop*. 2022.

[14] M. Malewicz and D. Malewicz, *Designing User Interfaces*. 2020.

[15] A. Rao, "AttenFace: A Real Time Attendance System using Face Recognition," Nov. 2022, [Online]. Available: http://arxiv.org/abs/2211.07582