SPECTA Journal of Technology
Vol.6, No.1, Maret-April 2022, E-ISSN : 2622-9099 | P-ISSN : 2549-2713
DOI : 10.35718/specta.v6i1

Pengembangan Aplikasi Deteksi Pelanggaran Protokol Covid-19 dengan metode Convolutional Neural Network berbasis Arsitektur MVVM

Muhamad Reinaldy Hermawan[1], Muhammad Nabil Akbar Pratama[2], Muhammad Raihan Rahman[3], Ariyadi[4]
Program Studi Informatika, Jurusan Matematika dan Teknologi Infromasi, Institut Teknologi Kalimantan [1,2,3,4]
11181051@student.itk.ac.id[1], 11181059@student.itk.ac.id[2], 11181061@student.itk.ac.id[3], ariyadi@lecturer.itk.ac.id[4]

Article Info

Article history:

Submitted September 2021
Revised October 2021
Accepted April 2022
Published April 2022

Keyword:

Android
Cloud
VisiTrack

ABSTRACT

In these modern times, there are many public places that have many visitors every day. However, because the ratio of officers is not comparable to that of visitors, a tool is needed to detect violations that occur in public places. So, the Android application VisiTrack is proposed. This application detects violations that occur in public places. This application detects violations that occur in public places through CCTV cameras that have been connected to machine learning models on cloud services. When a violation is detected, the cloud service will send a notification to the Android application so that the violation can be resolved by the officer at work. This application also implements the MVVM architectural pattern so that development can be carried out more easily and documentation can be followed easily. The finished result of this application has an accuracy rate of 95.59% and has succeeded in sending notifications of detected violations to the officers' mobile phones.

Kata Kunci:

Android
Cloud
VisiTrack

ABSTRAK

Pada zaman modern ini banyak tempat umum yang memiliki banyak pengunjung setiap harinya. Namun, dikarenakan rasio petugas yang tidak sebanding dengan pengunjung, maka dibutuhkanlah sebuah alat pembantu untuk mendeteksi pelanggaran yang terjadi di tempat umum. Maka dikembangkanlah aplikasi *Android VisiTrack*. Aplikasi ini mendeteksi pelanggaran yang terjadi di tempat umum lewat kamera CCTV yang telah disambungkan dengan model pembelajaran mesin pada layanan *cloud*. Ketika pelanggaran berhasil dideteksi, maka layanan *cloud* akan mengirimkan notifikasi kepada aplikasi *Android* sehingga pelanggaran yang terjadi bisa diatasi oleh petugas yang sedang bekerja. Aplikasi ini juga menerapkan pola arsitektur MVVM sehingga pengembangan dapat dilakukan dengan lebih mudah dan dokumentasi dapat diikuti dengan mudah. Hasil jadi dari aplikasi ini memiliki tingkat akurasi sebesar 95.59% dan sudah berhasil untuk mengirimkan notifikasi pelanggaran yang terdeteksi ke telepon genggam petugas.

1. PENDAHULUAN

1.1. Latar Belakang

Saat ini banyak sekali tempat-tempat umum yang memiliki banyak pengunjung atau tamu setiap harinya. Seperti taman umum, tempat parkir, kantor, pusat perbelanjaan, bandara, dan lainnya. Pada tahun 2020 tempat umum mulai mengalami pengurangan pengunjung akibat pandemi yang menimpa hampir seluruh negara di dunia. Namun, tidak jarang juga petugas keamanan pada tempat umum tersebut memiliki rasio yang sangat tidak sebanding dengan pengunjung yang datang. Karena hal tersebut, teknologi dapat membantu kinerja petugas keamanan seperti komputer dan kamera yang sudah murah dan mudah digunakan.

Rasio pengunjung yang lebih banyak dibandingkan dengan petugas yang ada membuat timbulnya pelanggaran-pelanggaran yang dilakukan oleh pengunjung, khususnya pelanggaran yang berhubungan dengan protokol kesehatan. Pada 23 Juni 2021 tercatat ada 15.308 kasus baru dalam sehari (CNN Indonesia, 2021). Sehingga pelanggaran protokol kesehatan merupakan hal krusial yang perlu diselesaikan dengan secepatnya. Pelanggaran ini juga yang merupakan alasan lain mengapa para pemilik tempat umum melakukan pemasangan kamera CCTV. Penerapan kamera CCTV sudah dilakukan, tetapi masih banyak pelanggaran yang tidak terdeteksi oleh petugas dan kamera. Masalah keamanan merupakan permasalahan yang umum pada tempat umum yang sampai sekarang masih belum dapat diatasi. Maka penyusun mengusulkan untuk membuat sebuah sistem di mana kamera CCTV akan langsung mendeteksi pelanggaran yang terjadi secara *Real Time* menggunakan model pembelajaran mesin dan mengirimkan notifikasi langsung pada aplikasi android pada telepon pribadi petugas yang sedang bekerja. Dengan solusi ini diharapkan dapat membantu mengurangi pelanggaran yang terjadi pada tempat umum dan juga dapat membantu meringankan kerja dari petugas keamanan pada tempat umum.

Maka tujuan penelitian yang ingin dicapai adalah sebagai berikut:

1. Membuat model pemelajaran mesin yang dapat mendeteksi pelanggaran pada tempat umum khususnya terkait protokol kesehatan.
2. Membuat aplikasi android untuk melihat data pelanggaran yang terjadi pada tempat umum.
3. Menghubungkan model pemelajaran mesin pada *cloud service* dan juga aplikasi android sehingga pelanggaran dapat dikirimkan secara *Real-time*.

1.2. Studi Literatur

1.2.1.XML

XML atau *Extensible Markup Language* adalah kelas objek data yang berbentuk dokumen dan sebagian menggambarkan perilaku program komputer yang memprosesnya. XML adalah profil aplikasi atau terbatas dari Standard *Generalized Markup Language* yang secara konstruksi dokumen XML sesuai dengan dokumen SGML. Dokumen XML terdiri dari unit penyimpanan yang disebut entitas, yang berisi data yang diuraikan atau tidak diuraikan. Data yang diurai terdiri dari karakter-karakter, sebagian membentuk data karakter, dan sebagian lagi merupakan *markup*. *Markup* mengkodekan deskripsi tata letak penyimpanan dokumen dan struktur logis. XML menyediakan mekanisme untuk memaksakan batasan pada tata letak penyimpanan dan struktur logis (Bray, Paoli, Sperberg-McQueen, Maler, & Yergeau, 2008). XML hampir sama dengan HTML (*HyperText Markup Language*) tetapi perbedaannya terdapat pada *Case-Sensitivity* di mana XML merupakan bahasa yang *Case-Sensitive*. Android menggunakan XML untuk melakukan desain aplikasi karena XML adalah bahasa yang ringan sehingga tidak membuat *layout* menjadi berat.

1.2.2.Pola Arsitektur

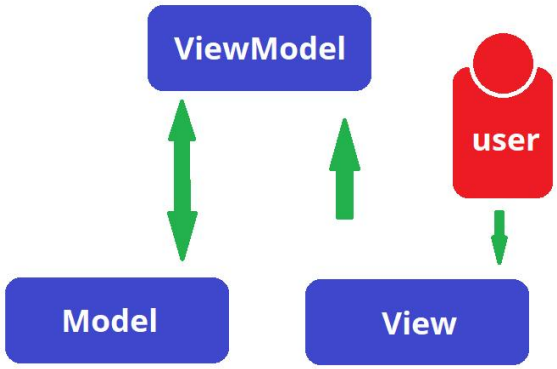
Pola arsitektur berhubungan erat dengan pola desain. Pola desain merupakan sebuah solusi umum yang telah teruji dan bisa digunakan kembali untuk menyelesaikan suatu masalah yang sering terjadi pada perancangan perangkat lunak. Pada pengembangan aplikasi, beberapa permasalahan yang bersifat berulang dapat diatasi dengan pola desain. Beberapa contoh jenis pola desain adalah *Creational* yang berhubungan dengan penciptaan suatu objek, *Structural* yang

Submitted September 2021, Revised October 2021, Accepted April 2022, Published April 2022

berhubungan dengan komposisi suatu objek, dan *Behavioral* yang berhubungan dengan komunikasi antar objek. Jadi, penggunaan pola desain ditentukan oleh permasalahan apa yang akan diatasi.

Berbeda dengan pola desain yang merupakan sebuah solusi yang lebih spesifik, pola arsitektur memiliki skala yang lebih besar tentang bagaimana pengorganisasian keseluruhan komponen aplikasi. Pola ini berfokus kepada pembagian tugas yang lebih jelas antar komponennya seperti logika dan desain. Beberapa contoh pola arsitektur adalah MVP (*Model-View-Presenter*), MVC (*Model-View-Controller*)¬, dan MVVM (*Model-View-ViewModel*). Perbedaan utamanya adalah pada hubungan antar komponen dan peran masing-masing komponen.

Pola MVVM membagikan tanggung jawab kepada *Model*, *View*, dan *ViewModel*. *View* akan bertanggung jawab untuk hal-hal yang berhubungan dengan tampilan dan UI. *Model* akan bertanggung jawab terkait hal-hal yang berhubungan dengan data. Sedangkan *ViewModel* adalah komponen yang menyimpan dan mengambil data dari *Model* dan kemudian menampilkannya ke *View*. MVVM sering digunakan pada pengembangan aplikasi Android karena pada Google I/O 2017, Google langsung merekomendasikan dan memberikan dukungan *library* untuk penggunaan *pattern* MVVM (Putra, 2019).



Gambar 1. Ilustrasi arsitektur MVVM

1.2.3.Injeksi Dependensi

Injeksi Dependensi (DI) merupakan sebuah teknik yang digunakan pada pemrograman dan sangan cocok diterapkan pada pengembangan aplikasi Android. Dengan mengikuti prinsip-prinsip DI, maka arsitektur aplikasi yang baik telah diterapkan. Keuntungan implementasi DI adalah penggunaan kembali kode, kemudahan dalam pemfaktoran ulang, dan kemudahan dalam pengujian (Google, 2020). DI bisa digunakan pada pemrograman Android dengan menggunakan pustaka *Dagger* atau *Koin*. Pustaka *Koin* adalah DI yang dibuat 100% *Kotlin* oleh Arnaud Giulani bersama dengan komunitas. Sedangkan pustaka *Dagger* merupakan pustaka yang dibuat oleh *Square* dan kini diteruskan oleh Google. Pustaka ini sudah banyak digunakan karena sudah lama dibuat. Pustaka ini akan otomatis membuat kode untuk injeksi saat diolah daripada menuliskannya satu per satu sehingga kode lebih mudah dites dan lebih mudah digunakan kembali.

2. METODE PENELITIAN

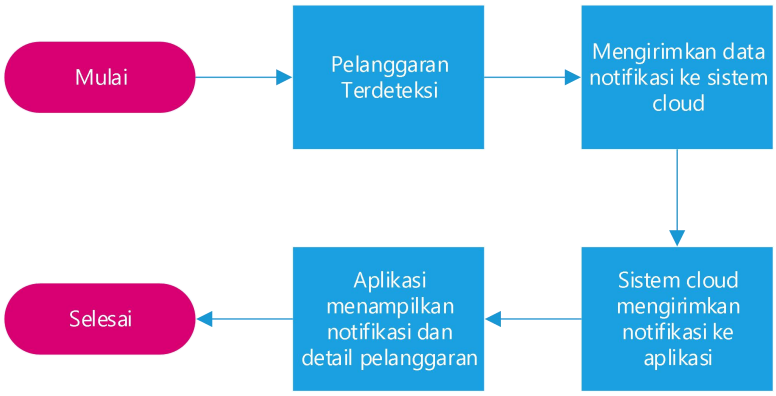
2.1. Pengumpulan Data

Pengumpulan data dilakukan dengan menggunakan 2 *dataset* utama, yaitu *dataset* untuk mendeteksi objek manusia dan untuk klasifikasi penggunaan masker. *Dataset* gambar digunakan untuk mendeteksi objek manusia dan dilakukan dengan cara mengambil gambar dari rekaman video CCTV yang ada di internet. Kemudian didapatkan data untuk pelatihan sebanyak 1.057 dan data untuk evaluasi sebanyak 125. Lalu, *dataset* untuk melakukan deteksi penggunaan masker digunakan komposisi data pelatihan sebanyak 350 data dan juga 350 data untuk yang tidak menggunakan masker, serta juga 306 data untuk melakukan validasi.

2.2. Metode yang Digunakan

Submitted September 2021, Revised October 2021, Accepted April 2022, Published April 2022

Alur deteksi yang digunakan pada sistem aplikasi dan model pemelajaran mesin yang berada di *cloud* adalah sebagai berikut.



Gambar 2. Diagram Alur Aplikasi

Ketika pelanggaran terdeteksi, maka data pelanggaran akan dikirimkan ke sistem *cloud* dari kamera CCTV. Setelah data sampai ke sistem *cloud* maka langsung akan diteruskan dalam bentuk notifikasi ke aplikasi yang sudah di *install* di telepon genggam pengguna. Lalu, pengguna akan melihat notifikasi dan melihat detail pelanggaran di aplikasi. Kemudian untuk model pemelajaran mesin yang digunakan pada sistem ini ada dua yaitu model pendeteksi manusia dan juga model pendeteksi masker.

2.3. Prosedur dan Implementasi

Pengembangan aplikasi ini dilakukan dengan bahasa pemrograman *Kotlin* dan dengan menggunakan Pola arsitektur MVVM. Pada awalnya akan dibuat desain dari aplikasi itu sendiri lewat *Figma*. Setelah selesai maka struktur data dari aplikasi akan dibangun dengan memanfaatkan Injeksi Dependensi agar kode dapat lebih mudah digunakan, lebih mudah dilakukan faktor ulang, dan lebih mudah dites. Pada sisi *cloud* akan dibuat sebuah *virtual device* yang memuat model dari pemelajaran mesin dan akan dihubungkan dengan kamera CCTV. Lalu API akan dibuat agar aplikasi dapat mengakses data pelanggaran yang ada di *cloud* dan *Firebase* digunakan untuk mengirimkan *Push Notification* pada aplikasi.

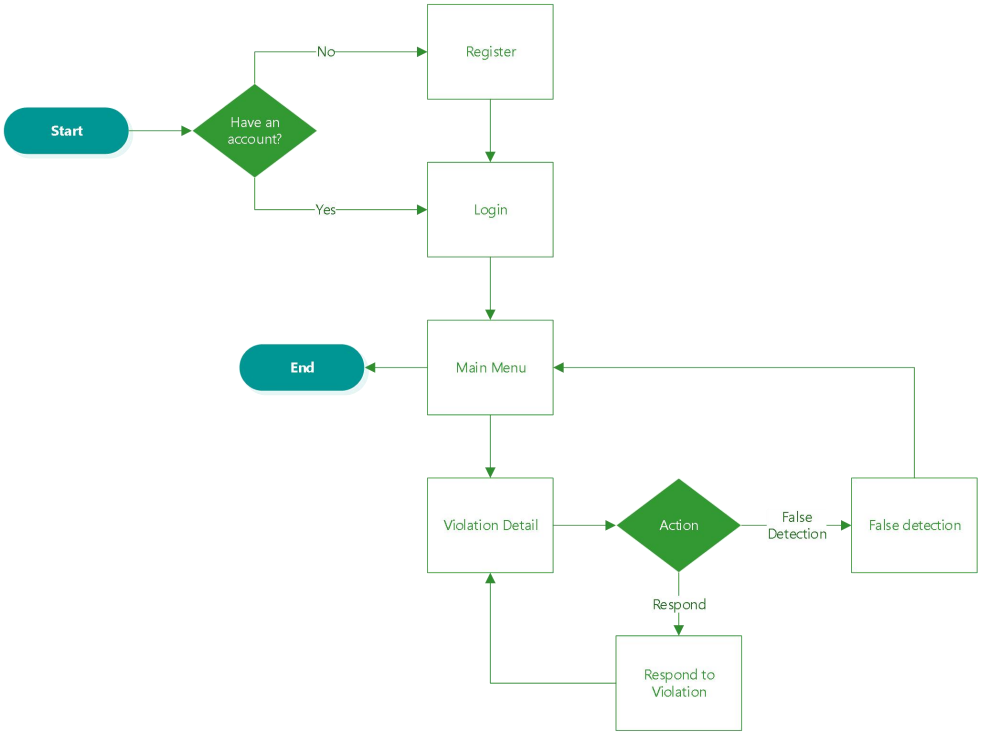
3. HASIL DAN ANALISIS

Aplikasi yang dibuat bernama *VisiTrack*. *VisiTrack* merupakan sebuah aplikasi yang memanfaatkan pembelajaran mesin dan menyambungkan teknologi CCTV sebagai alat bantu deteksi pelanggaran pada tempat umum dan langsung mengirimkan notifikasi dan data pelanggaran yang terdeteksi ke telepon genggam petugas keamanan pada tempat tersebut. Penyusun mengerjakan dengan tim dan membagi tugas masing-masing keahlian. Penyusun bertanggungjawab atas pembuatan aplikasi Android *VisiTrack* dengan bahasa pemrograman *Kotlin*. Aplikasi akan mengakses data yang telah diolah dan disimpan pada layanan *cloud* dari *Google Cloud Service* lewat bantuan API dan *Firebase*.

3.1. Alur Aplikasi

Alur aplikasi yang dibuat pada aplikasi *VisiTrack* pada program Google Bangkit 2021 adalah seperti pada gambar 3.

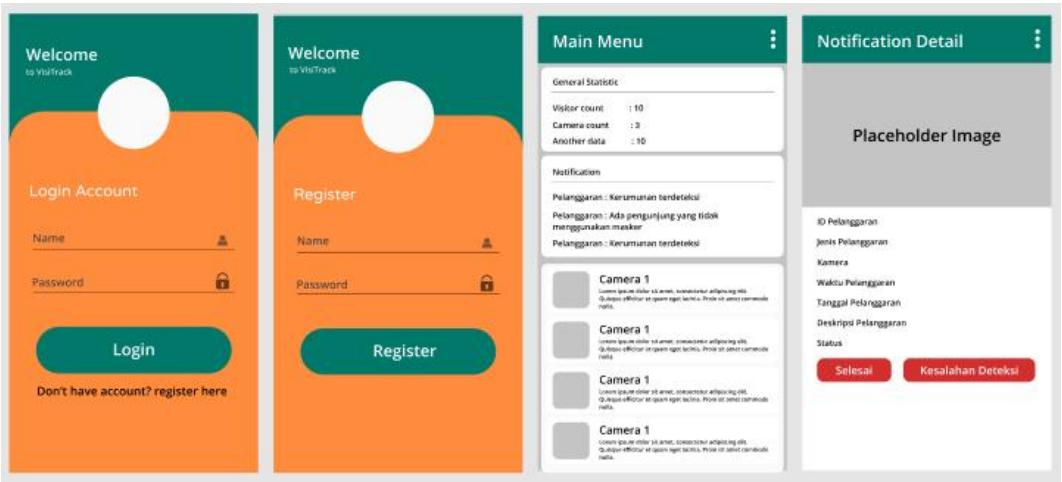
Submitted September 2021, Revised October 2021, Accepted April 2022, Published April 2022



Gambar 3. Alur Aplikasi

3.2. Desain Mockup Aplikasi

Desain *mockup* aplikasi dibuat untuk menunjukkan bagaimana aplikasi akan dibuat dan sebagai acuan desain awal aplikasi. Desain *mockup* aplikasi dibuat pada sebuah platform bernama *Figma*. Desain awal aplikasi yang telah dibuat adalah seperti pada gambar 4.



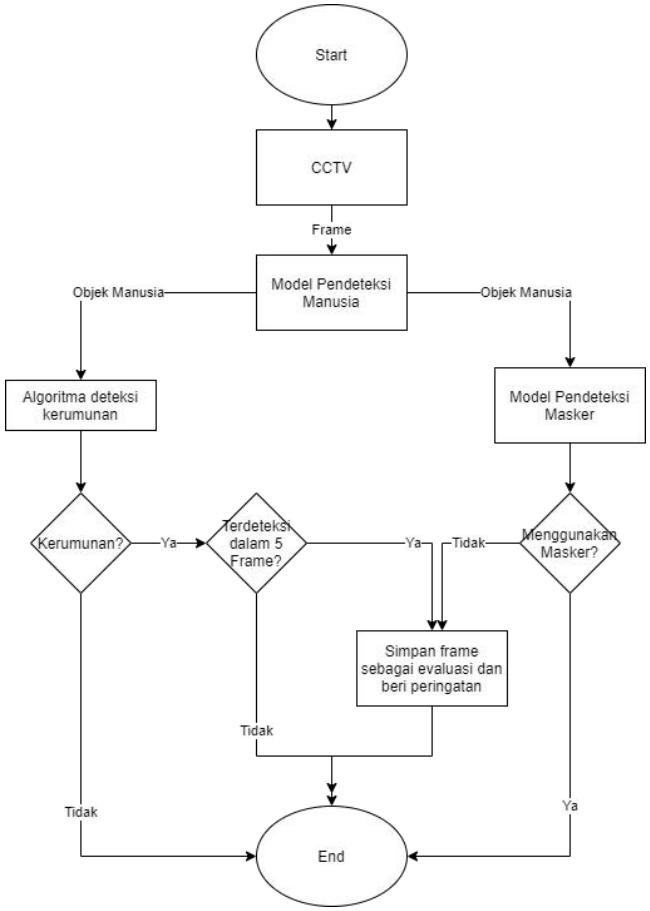
Gambar 4. Desain Awal Aplikasi

Pada gambar 4 terlihat ada 4 halaman yang sudah dibuat. Halaman yang sudah dibuat merupakan halaman autentikasi, halaman pendaftaran, halaman utama, dan halaman notifikasi. Halaman autentikasi akan muncul paling awal jika pengguna belum pernah melakukan autentikasi sebelumnya. Halaman pendaftaran berfungsi untuk mendaftarkan akun pengguna baru dan akan muncul jika pengguna menekan tombol daftar pada halaman autentikasi. Halaman utama merupakan halaman utama yang berisi data-data yang terlibat dalam aplikasi. Halaman notifikasi berfungsi untuk menampilkan data detail notifikasi dan akan muncul jika pengguna menekan salah satu notifikasi yang ada pada halaman utama.

Submitted September 2021, Revised October 2021, Accepted April 2022, Published April 2022

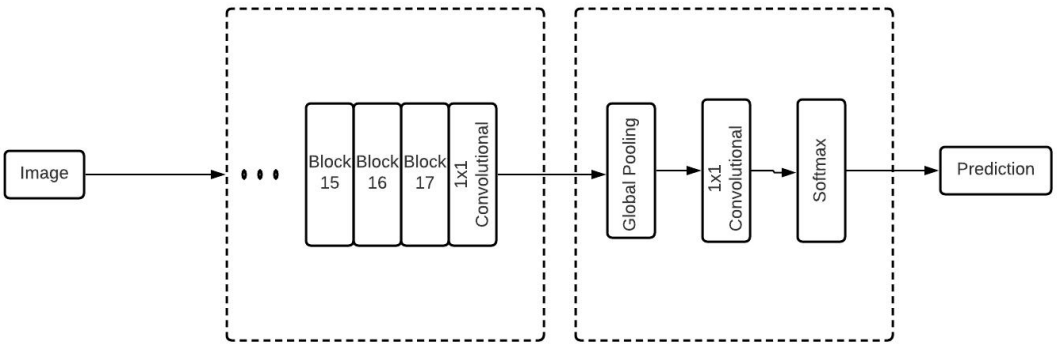
3.3. Pembuatan Model CNN

Model yang akan dibuat pada sistem ini ada dua, yaitu model pendeteksi manusia dan model pendeteksi penggunaan masker.



Gambar 5. Diagram Blok Proses Pendeteksi Pelanggaran Protokol Kesehatan

Yang pertama dilakukan adalah proses pembuatan model pendeteksi manusia. Model ini akan mendeteksi setiap objek berupa manusia yang terdapat pada suatu gambar, dan menyimpan data lokasi x dan y dari objek tersebut pada gambar. Model ini akan ditrain menggunakan metode CNN dengan basis SSD Mobilenet V2 Coco. Model akan di-*train* menggunakan objek manusia agar model dapat mengenali objek tersebut pada suatu gambar dan mengembalikan nilai berupa lokasi objek pada gambar tersebut.



Gambar 6. Arsitektur Mobilenet

Kemudian dilakukan proses pembuatan model untuk mendeteksi pengguna masker. Hasil dari model ini merupakan sebuah nilai desimal dari 0 sampai 1, jika nilai hasil dari klasifikasi lebih dari 0.5 maka gambar tersebut akan diklasifikasikan sebagai tidak menggunakan masker dan jika kurang dari atau sama dengan 0.5 maka akan diklasifikasikan sebagai menggunakan masker. Model pengklasifikasian ini menggunakan metode *convolutional neural network*. Dengan layer-layer yang digunakan adalah seperti pada tabel di bawah.

Tabel 1. Layer Model Pengklasifikasian Penggunaan Masker

Layer	Parameter	Nilai
Conv2D	Filter	2
	Kernel Size	(3,3)
	Aktivasi	Relu
	Input	(150, 150, 3)
MaxPooling2D	Pool Size	(2,2)
	Stride	(2,2)
Conv2D	Filter	32
	Kernel Size	(3,3)
	aktivasi	relu
MaxPooling2D	Pool Size	(2,2)
	Stride	(2,2)
Conv2D	Filter	64
	Kernel Size	(3,3)
	aktivasi	relu
MaxPooling2D	Pool Size	(2,2)
	Stride	(2,2)
Flatten	-	-
Dense	Unit	512
	Aktivasi	Relu
Dense	Unit	1
	Aktivasi	Sigmoid

Kemudian dilakukan *training* pada model tersebut dengan loss menggunakan metode *binary crossentropy,* optimasi Adam dengan nilai *learning rate* sebesar 0,001. Dengan jumlah pengulangan sebanyak 50 kali didapatkan nilai akurasi 95,59%. Kemudian model tersebut diekstrak menjadi sebuah file dengan format h5.

3.4. Implementasi Pola Arsitektur pada Aplikasi

Pola arsitektur digunakan dengan menggunakan arsitektur MVVM (*Model-View-ViewModel*). Digunakan arsitektur tersebut karena penyusun memiliki referensi lebih banyak dengan arsitektur ini dan juga arsitektur ini paling cocok dengan pengembangan aplikasi Android. *ViewModel* juga paling cocok jika digunakan dengan *LiveData*.

3.5. Menghubungkan Aplikasi dengan Layanan Cloud

Notifikasi dikirimkan melalui layanan *Firebase Cloud Messaging*. FCM digunakan karena implementasinya lebih mudah karena data disimpan pada *Google Cloud Service*. Notifikasi dikirimkan jika pelanggaran terdeteksi dari kamera CCTV. FCM langsung mengirimkan tipe pelanggaran dan langsung terlihat ke telepon genggam petugas. Ketika notifikasi ditekan maka aplikasi akan terbuka dan pelanggaran terbaru akan terlihat. FCM digunakan pada aplikasi dengan melakukan *subscribe* notifikasi dan menggunakan *token* yang akan dikirimkan saat proses autentikasi.

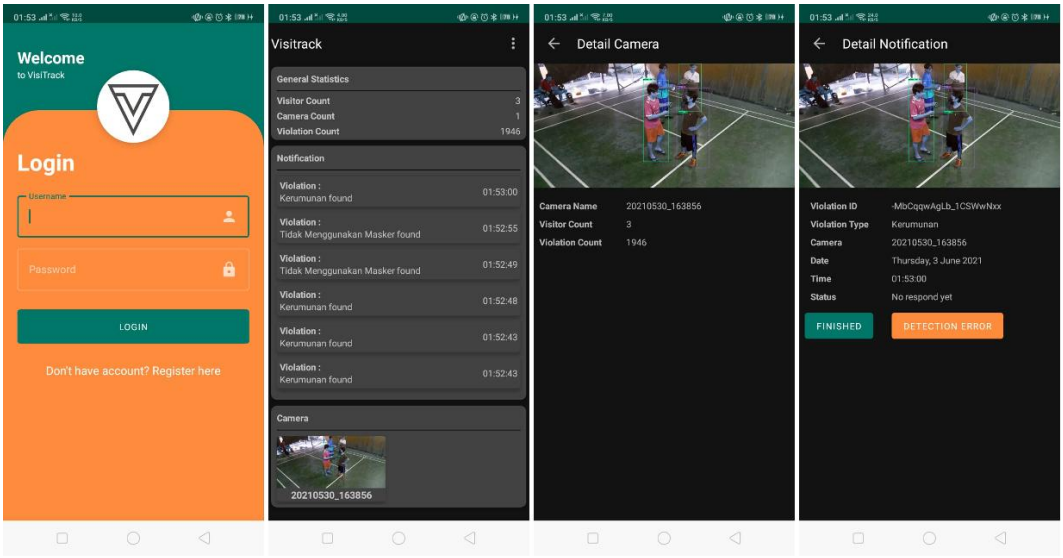
Untuk data pada basis data *cloud* diakses dengan menggunakan API yang dibuat pada layanan *cloud*. API akan mengirimkan JSON sesuai dengan data yang diminta oleh aplikasi lewat *Retrofit*. Setelah mendapatkan data, maka JSON tadi akan diproses atau diuraikan menjadi data

Submitted September 2021, Revised October 2021, Accepted April 2022, Published April 2022

yang sesuai dengan Model yang telah dibuat pada aplikasi. Pada proses autentikasi juga digunakan proses yang sama. Data pengguna akan dikirimkan menggunakan *Retrofit* dan API akan memproses data tersebut lalu mengirimkan balik data *token* yang digunakan sebagai autentikasi pengguna.

3.6. Hasil Akhir Aplikasi

Hasil akhir aplikasi telah dibangun dapat dilihat pada gambar 7 berikut ini.



Gambar 7. Tampilan Akhir Aplikasi.

Pada gambar 7 terlihat ada beberapa halaman pada hasil akhir aplikasi. Halaman autentikasi merupakan halaman pertama yang akan muncul jika pengguna belum pernah melakukan autentikasi sebelumnya. Halaman utama berfungsi untuk menampilkan data-data yang dibutuhkan oleh pengguna. Halaman detail kamera berfungsi untuk menampilkan data detail kamera dan statistik pada kamera tersebut. Halaman detail notifikasi berfungsi untuk menampilkan data detail notifikasi yang telah terdeteksi.

Pengujian dilakukan dengan menggunakan video yang di *looping* sehingga mensimulasikan kamera CCTV pada tempat umum. Video ini direkam pada sebuah lapangan bulutangkis *indoor* dengan pencahayaan yang baik dan didalam video terdapat pelanggaran protokol kesehatan yang harusnya akan terdeteksi oleh model pemelajaran mesin. Video tadi dipasangkan pada sistem *cloud* lalu model pemelajaran mesin dipasangkan untuk mendeteksi pelanggaran. Lalu dilakukan juga validasi data dengan dataset yang telah dikumpulkan sehingga didapatkan persentase akurasi dari model pemelajaran mesin yang disimpan pada sistem *cloud* memiliki nilai 95,59% dengan menggunakan metode CNN dan *Mobilenet* SSD v2.

4. KESIMPULAN

Kesimpulan yang didapatkan pada program yang telah dibangun yaitu sebagai berikut.

1. Pengembangan model pemelajaran mesin berhasil dengan akurasi 95.59% yang didapatkan dengan melakukan *transfer learning* dari *Mobilenet* SSD v2 dan menggabungkan dengan metode CNN.
2. Pengembangan aplikasi android sudah bekerja dan dapat menampilkan pelanggaran yang terdeteksi yang dibuat dengan menerapkan pola arsitektur MVVM dan juga menerapkan Injeksi Dependensi.
3. Aplikasi android dan model pemelajaran mesin sudah terhubung menggunakan API dan juga *Firebase* sehingga data dapat dikirimkan langsung secara *Real-time* saat pelanggaran terdeteksi di kamera CCTV.
4. Kelebihan pada sistem ini adalah mudah dikembangkan lebih lanjut dan menambahkan studi kasus yang lebih banyak dan bahkan tidak berhubungan dengan Covid-19.
5. Kekurangan pada sistem ini adalah masih sedikitnya tipe pelanggaran yang ada.

Submitted September 2021, Revised October 2021, Accepted April 2022, Published April 2022

UCAPAN TERIMA KASIH

Terima kasih saya berikan kepada tim Google Bangkit yang sudah memfasilitasi dan memberikan materi-materi yang berhubungan dengan pemrograman Android dan memberikan kesempatan saya mengerjakan dan memberi mentor terkait penyelesaian program ini. Terima kasih juga saya berikan kepada rekan satu tim saya pada proyek ini dan juga mentor dan dosen yang telah memberikan bimbingan.

REFERENSI

[1] Bray, T., Paoli, J., Sperberg-McQueen, C. M., Maler, E., & Yergeau, F. (2008). *Extensible Markup Language (XML) 1.0*. Retrieved Juli 2013, 2021, from https://www.w3.org/TR/REC-xml/

[2] CNN Indonesia. (2021, June 24). *Menkes Enggan Prediksi Akhir Pandemi: Selalu Salah*. Retrieved from CNN Indonesia: https://www.cnnindonesia.com/nasional/20210624084510-20-658692/menkes-enggan-prediksi-akhir-pandemi-selalu-salah

[3] Google. (2020, June 10). *Dependency injection in Android*. Retrieved Juli 13, 2021, from https://developer.android.com/training/dependency-injection

[4] Putra, S. A. (2019, May 25). *Android MVVM*. Retrieved from Medium: https://medium.com/codelabs-unikom/android-mvvm-part-1-pengenalan-mvvm-ebeeb397b427

Submitted September 2021, Revised October 2021, Accepted April 2022, Published April 2022