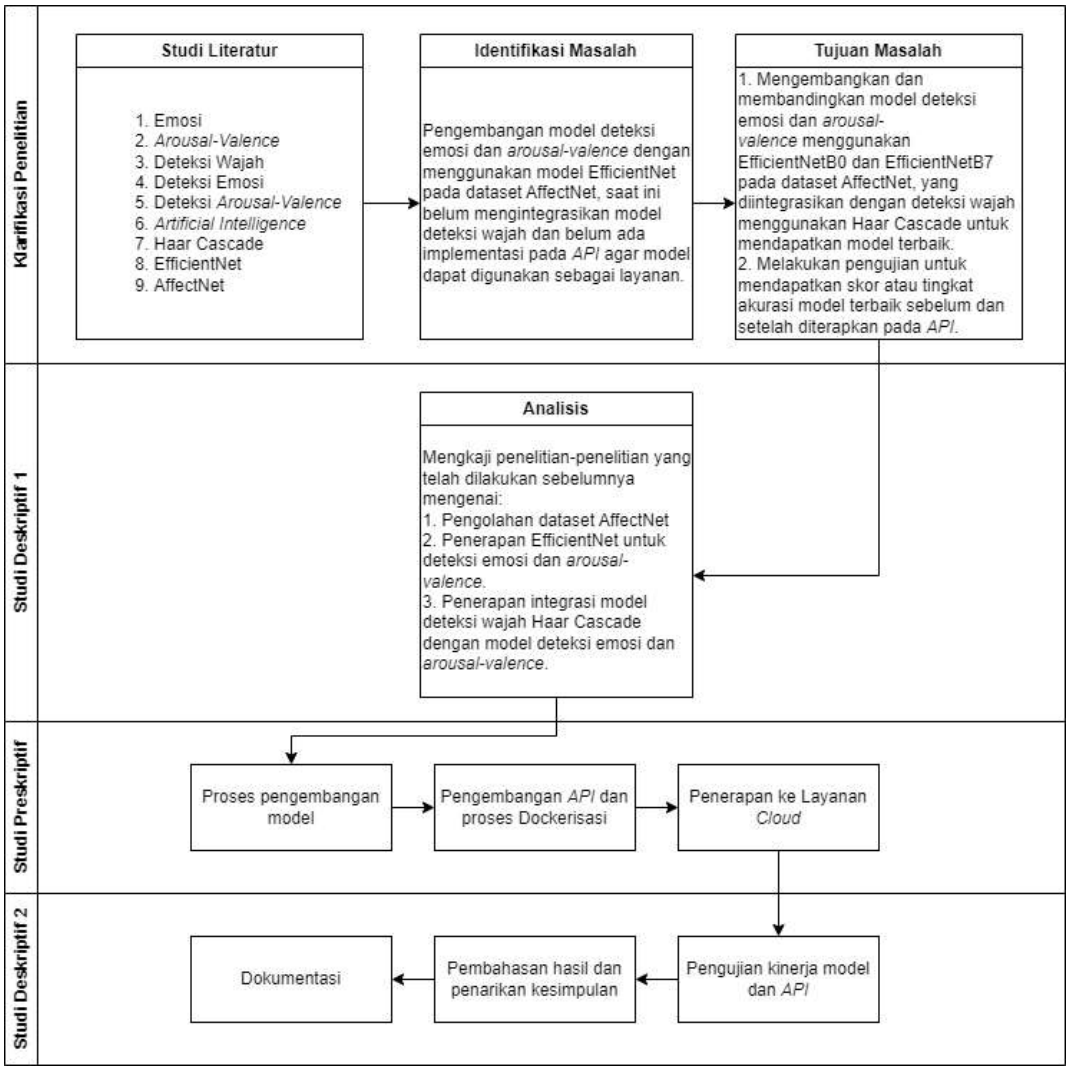
BAB III
METODE PENELITIAN

3.1 Desain Penelitian

Metode yang digunakan pada penelitian ini adalah metode DRM (Design Research Methodology). Metode DRM adalah sebuah kerangka kerja penelitian yang dirancang untuk membantu para peneliti dalam melakukan penelitian dengan desain yang sistematis dan terstruktur, DRM memberikan tujuan untuk membuat dan menguji produk pada penelitian yang sedang dilakukan (Yüksel dkk., 2023). Berikut adalah desain penelitian berdasarkan metode DRM yang dapat dilihat pada Gambar 3.1:



Gambar 3.1 Desain Penelitian

3.2 Klarifikasi Penelitian

Tahap ini memiliki tujuan untuk membahas dan memahami permasalahan yang sedang diteliti secara mendalam. Melakukan tinjauan terhadap studi literatur yang berkaitan atau relevan mengenai emosi, *arousal-valence*, deteksi wajah, deteksi emosi, deteksi *arousal-valence*, *artificial intelligence*, Haar Cascade, EfficientNet, dan AffectNet. Selanjutnya, dilakukan identifikasi masalah yang berkaitan dengan deteksi emosi dan *arousal-valence* yang saat ini masih belum mengintegrasikan model deteksi wajah dan implementasi pada API sebagai layanan untuk mendapatkan hasil dan kinerja yang optimal.

3.3 Studi Deskriptif 1

Tahap ini merupakan proses untuk membahas dan menganalisis pada penelitian-penelitian sebelumnya yang berkaitan dengan model deteksi wajah, deteksi emosi, dan deteksi *arousal-valence*. Selain itu, tahap ini juga akan dilakukan perbandingan terhadap penelitian-penelitian sebelumnya untuk mengetahui kelebihan dan kekurangannya dengan tujuan memahami hal atau faktor apa yang harus ditekankan untuk pengembangan penelitian selanjutnya.

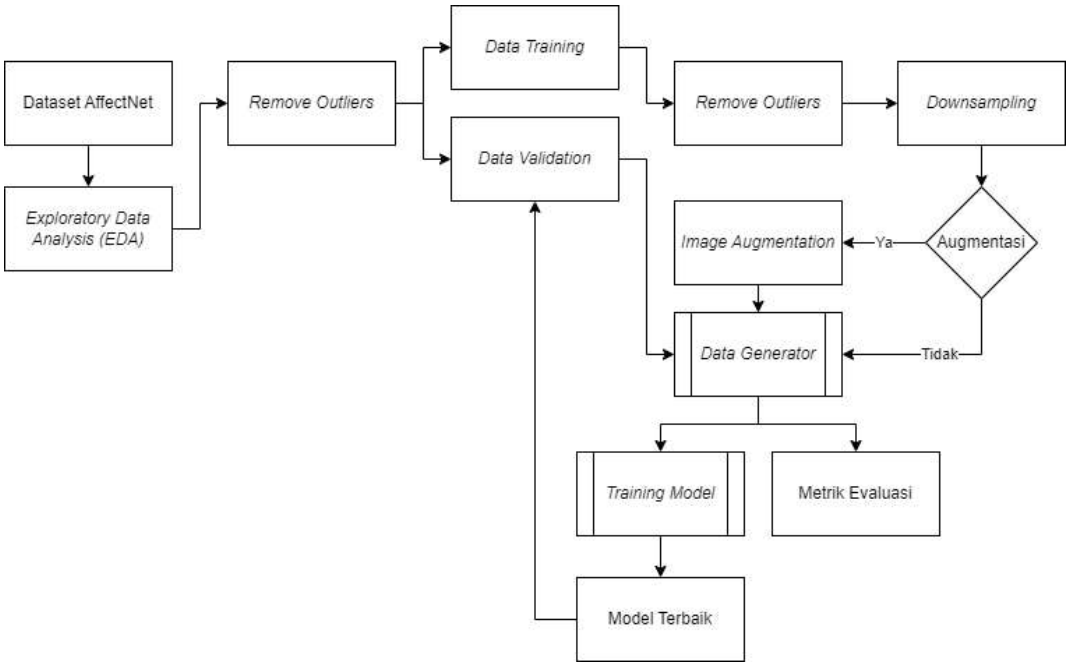
3.4 Studi Preskriptif

Tahap ini merupakan proses untuk melakukan dan mengimplementasi metode yang akan digunakan untuk menyelesaikan masalah penelitian. Tahap-tahap yang akan dilakukan pada studi preskriptif ini untuk mengimplementasikan model deteksi wajah dengan deteksi emosi dan *arousal-valence* pada API dengan penerapan Haar Cascade dan EfficientNet yaitu pengembangan model, pengembangan API, proses *Dockerisasi*, dan menerapkan atau *deploy* ke layanan *cloud*.

3.4.1 Pengembangan Model

Pengembangan model dilakukan dengan beberapa tahap mulai dari *Exploratory Data Analysis* sampai metrik evaluasi. Fokus dari pengembangan model adalah untuk melatih model deteksi emosi dan *arousal-valence* menggunakan EfficientNet versi ringan yaitu EfficientNetB0 dan berat yaitu EfficientNetB7 seperti pada dokumentasi resmi Tan dan Le (2019), menerapkannya pada dataset AffectNet dan mengintegrasikannya dengan Haar Cascade untuk

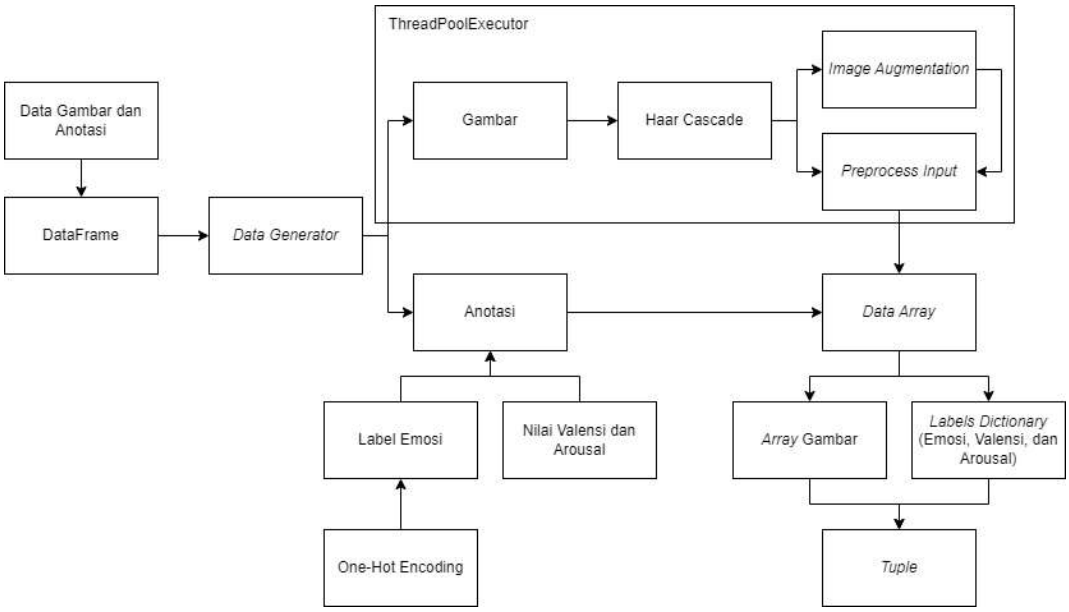
mendeteksi wajah. Model Haar Cascade yang digunakan adalah model haarcascade_frontalface_alt2.xml yang merupakan model alternatif dan dapat mendeteksi wajah dengan orientasi yang berbeda serta memiliki sensitivitas yang lebih tinggi. Dalam tahap pengembangan model, dilakukan pencarian untuk mendapatkan model terbaik untuk diimplementasikan pada API. Tahap-tahap pengembangan model dapat dilihat pada gambar berikut:



Gambar 3.2 Proses Pengembangan Model Deteksi Emosi dan *Arousal-Valence*

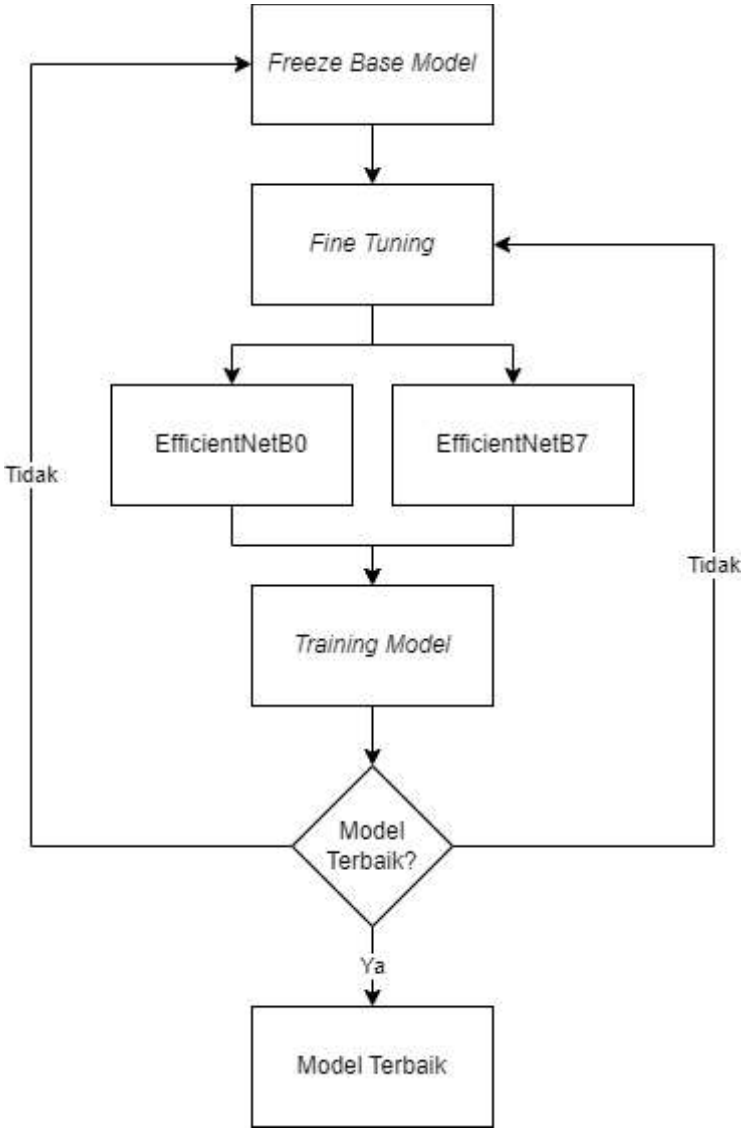
Pada Gambar 3.2 di atas dapat dilihat proses pengembangan model deteksi emosi dan *arousal-valence* yang dimulai dari pengumpulan data yang didapatkan dari dataset AffectNet, lalu dilakukan eliminasi terhadap data-data yang tidak diperlukan, pembagian data menjadi *data training* atau set pelatihan dan *data validation* atau set pengujian, untuk set pengujian berdasarkan dokumentasi resmi pada penelitian Mollahosseini dkk. (2019) menggunakan set validasi sebagai dasar perbandingan yaitu berisikan 500 data untuk setiap kategori emosi. Pada set pelatihan akan dilakukan eliminasi kembali agar mendapatkan data terbaik untuk pelatihan, lalu diterapkan metode *downsampling* untuk menyeimbangkan data agar tidak cenderung terhadap kategori yang memiliki data lebih banyak, dan dilakukan augmentasi gambar apabila diperlukan. Data-data pada set pelatihan tersebut selanjutnya diterapkan pada *data generator* untuk mempersiapkan data agar dapat

dilakukan proses-proses dalam model. Setelah model selesai dilakukan pelatihan akan dilakukan pencarian model terbaik dengan menggunakan set pengujian yang diterapkan pada *data generator* untuk dilakukan evaluasi menggunakan metrik evaluasi dan mengontrol model terbaik yang didapatkan pada saat pelatihan. Berikut merupakan proses-proses dalam *data generator* dan *training model* atau pelatihan model secara rinci:



Gambar 3.3 Proses *Data Generator*

Pada Gambar 3.3 di atas dapat dilihat proses-proses pada *data generator* dengan proses pertama yaitu dari data gambar dan anotasi AffectNet yang telah diproses menjadi DataFrame, data tersebut akan dibagi menjadi dua proses dalam *data generator*, untuk yang pertama persiapan data gambar dengan melakukan Haar Cascade dan atau menggunakan augmentasi gambar untuk selanjutnya dilakukan *preprocessing* agar sesuai dengan format pengembangan model menggunakan TensorFlow, dan untuk proses yang kedua akan dilakukan persiapan anotasi-anotasi meliputi label emosi yang dibentuk menjadi *one-hot encoding* serta nilai valensi dan arousal. Selanjutnya data yang sudah dipersiapkan disimpan dalam bentuk *array* yang berisikan *array* gambar dan *dictionary* dari anotasi-anotasinya serta dibentuk menjadi satu *tuple* untuk menyimpan data-data tersebut. Berikut proses dari *training model* atau pelatihan model yang mencari model terbaik untuk selanjutnya diimplementasikan pada API:



Gambar 3.4 Proses *Training* Model atau Pelatihan Model

Pada Gambar 3.4 di atas dapat dilihat proses-proses pada *training model* atau pelatihan model dengan proses pencarian model terbaik untuk selanjutnya diimplementasikan pada API. Model akan melakukan pelatihan model kembali apabila model tersebut masih belum mencapai model yang optimal dengan kembali ke proses penggunaan versi model dan metode yang digunakan. Hasil dari beberapa eksperimen untuk pencarian model terbaik selanjutnya akan digunakan untuk analisis perbandingan dari kedua model tersebut yaitu EfficientNetB0 dan EfficientNetB7. Parameter-parameter yang digunakan pada proses *training* model atau pelatihan model merujuk pada beberapa penelitian sebelumnya seperti jumlah

epoch, *optimizer* Adam, dan lain sebagainya termasuk parameter augmentasi gambar (Yen dan Li, 2022).

3.4.2 Pengembangan API dan Proses Dockerisasi

Tahap ini adalah proses untuk membangun API untuk melayani dan menjalankan model yang telah dikembangkan agar pengembang selanjutnya dapat menjalankan model tanpa perlu untuk menyesuaikan *preprocessing* yang diperlukan model. Tahap ini juga mencakup proses dockerisasi agar API yang telah dibangun dapat lebih portabel dan dapat dijalankan tanpa perlu menyesuaikan *environment* untuk menjalankan API.

3.4.3 Penerapan ke Layanan *Cloud*

Tahap ini merupakan proses untuk menerapkan hasil Dockerisasi API yang telah dikembangkan untuk dilakukan *deploy* ke layanan *cloud*. Hasil Dockerisasi API tersebut akan dilakukan *push* ke repositori Google Cloud (Artifact Registry) untuk selanjutnya dijalankan dengan melakukan *deploy* dalam Google Cloud Run dan mendapatkan tautan untuk selanjutnya dapat digunakan oleh sisi klien.

3.5 Studi Deskriptif 2

Tahap ini merupakan tahap terakhir yang memiliki tujuan untuk mengukur tingkat keberhasilan dari temuan yang didapat pada penelitian yang telah dilakukan yang selanjutnya dievaluasi dan melakukan penarikan kesimpulan. Dalam tahap ini, akan dilakukan pengujian menggunakan data yang telah dibagi sebelumnya sebagai data *validation* atau set pengujian agar dapat diketahui nilai akurasi atau ketepatan model yang telah dikembangkan dengan beberapa metrik evaluasi sesuai dengan *output* dari model seperti *confusion matrix*, *precision*, *recall*, dan *f1-score* untuk *output* klasifikasi emosi serta *MAE*, *MSE*, dan *RMSE* untuk *output* nilai valensi dan arousal. Selain itu, dalam tahap ini juga dilakukan pengujian terhadap performa model setelah dilakukan implementasi pada API, dengan melihat perbandingan antara hasil pengujian pada set pengujian sebelum dan setelah diterapkan pada API, serta melakukan pengujian untuk mengukur performa API.

3.5.1 *Confusion Matrix*

Confusion Matrix adalah alat yang digunakan untuk mengevaluasi kinerja model klasifikasi. *Confusion Matrix* berbentuk tabel akan memberikan gambaran mengenai seberapa baik kinerja model yang telah dikembangkan.

Tabel 3.1
Confusion Matrix

	Hasil Prediksi Positif	**Hasil Prediksi Negatif**
Aktual Positif	TP (*True Positive*)	FN (*False Negative*)
Aktual Negatif	FP (*False Positive*)	TN (*True Negative*)

Berikut merupakan penjabaran dari empat sel pada Tabel 3.1:

- TP (*True Positive*): Label data yang secara aktual positif dan diprediksi positif oleh model.
- FN (*False Negative*): Label data yang secara aktual positif dan diprediksi negatif oleh model.
- FP (*False Positive*): Label data yang secara aktual negatif dan diprediksi positif oleh model.
- TN (*True Negative*): Label data yang secara aktual negatif dan diprediksi negatif oleh model.

Confusion Matrix ini menggambarkan kesalahan dari hasil model klasifikasi dengan memasukkan hasil klasifikasi atau prediksi ke dalam Tabel 3.1, sehingga dapat membantu dalam menghitung tingkatan akurasi atau penilaian dari kinerja sebuah model seperti akurasi, *precision*, *recall*, dan *f1-score* (Grandini dkk., 2020).

3.5.2 Akurasi

Akurasi merupakan metrik untuk mengukur model yang telah dikembangkan dapat melakukan klasifikasi atau prediksi data dengan benar (Grandini dkk., 2020). Pengukuran akurasi model dapat dilakukan menggunakan rumus berikut:

$$Akurasi = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

3.5.3 *Precision*

Precision merupakan nilai atau skor perbandingan antara TP (*True Positive*) dengan semua hasil prediksi yang bernilai positif (benar-benar positif) (Grandini

dkk., 2020). Pengukuran nilai atau skor *precision* pada model dapat dilakukan menggunakan rumus berikut:

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

3.5.4 *Recall*

Recall merupakan nilai atau skor perbandingan antara TP (*True Positive*) dengan label data aktual atau sebenarnya (diprediksi dengan benar) (Grandini dkk., 2020). Pengukuran nilai atau skor *recall* pada model dapat dilakukan menggunakan rumus berikut:

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

3.5.5 *F1-Score*

F1-Score merupakan metrik yang menggabungkan *precision* dan *recall* untuk mengevaluasi kinerja model yang memberikan keseimbangan antara *precision* dan *recall* (Grandini dkk., 2020). Pengukuran nilai atau skor *f1-score pada model* dapat dilakukan menggunakan rumus berikut:

$$f - 1\ score = \frac{2 * (precision * recall)}{precision + recall} \quad (4)$$

3.5.6 *Mean Absolute Error (MAE)*

Mean Absolute Error (MAE) merupakan metrik untuk mengukur model yang telah dikembangkan dan dihitung sebagai rata-rata dari nilai absolut dari selisih antara nilai prediksi dan nilai aktual (Chicco dkk., 2021). Pengukuran *MAE* model dapat dilakukan menggunakan rumus berikut:

$$MAE = \frac{1}{n}\sum_{i=1}^{n} |y_i - \hat{y}| \quad (5)$$

y_i sebagai nilai sebenarnya dan $\hat{y}$ sebagai nilai hasil prediksi

3.5.7 *Mean Squared Error (MSE)*

Mean Squared Error (MSE) merupakan metrik untuk mengukur model yang telah dikembangkan dan dihitung sebagai rata-rata dari nilai kuadrat dari selisih

antara nilai prediksi dan nilai aktual (Chicco dkk., 2021). Pengukuran *MSE* model dapat dilakukan menggunakan rumus berikut:

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2 \quad (6)$$

3.5.8 *Root Mean Squared Error (RMSE)*

Root Mean Squared Error (RMSE) merupakan metrik untuk mengukur model yang telah dikembangkan dan merupakan akar kuadrat dari *MSE* (Chicco dkk., 2021). Pengukuran *RMSE* model dapat dilakukan menggunakan rumus berikut:

$$RMSE = \sqrt{MSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y})^2} \quad (7)$$

3.6 Partisipan

Partisipan yang akan terlibat dalam penelitian ini yaitu pengguna virtual yang melakukan *request* pada API secara bersamaan dalam waktu satu detik sekali. Jumlah yang akan terlibat yaitu 35 orang sesuai dengan jumlah yang dipaparkan oleh (Rößler dkk., 2021) untuk mencoba penerapan deteksi emosi yang sudah dikembangkan sebelumnya. Partisipan yang terlibat akan disiapkan melalui alat bernama *Locust* untuk melakukan pengujian pada API.

3.7 Waktu dan Tempat Penelitian

Penelitian ini akan dilakukan pada rentang Maret 2024 – Juli 2024 dengan pengujian yang dilakukan secara daring.

3.8 Lingkungan Komputasi

Penelitian ini dilakukan dalam lingkungan komputasi sebagai berikut:

1) Perangkat Keras
   a) Prosesor Intel Core i3-6100U
   b) RAM 16GB DDR4
   c) Intel® HD Graphics 520
   d) SSD 240GB
2) Perangkat Lunak
   a) Sistem Operasi Windows 10

b) Python sebagai bahasa pemrograman
c) Google Colaboratory dengan *runtime* TPUv2, 8 *cores*, dan 8GB memori HBM per *core* digunakan untuk lingkungan pengembangan model.
d) Docker digunakan untuk membuat kontainer hasil dari pengembangan API.
e) Google Cloud dengan Cloud Run *resources* 4 GiB memori, 2 CPU, 100 *maximum concurrent requests/instance*, dan 5 *maximum number of instances* digunakan untuk layanan *cloud*.

3) Daftar *Library*
a) TensorFlow dan Keras merupakan *framework deep learning* yang digunakan untuk melakukan pemrosesan data, pembangunan, dan pelatihan model deteksi emosi dan *arousal-valence*.
b) pandas digunakan untuk mengolah, memanipulasi, analisis data tabular, dan menyimpan data dalam bentuk dokumen.
c) NumPy digunakan untuk pemrosesan data numerik dan hal-hal yang berhubungan dengan *array* dan objek.
d) Matplotlib dan seaborn digunakan untuk visualisasi data.
e) OpenCV digunakan untuk pemrosesan gambar dan implementasi Haar Cascade.
f) scikit-learn digunakan untuk pengujian model menggunakan metrik evaluasi.
g) Flask digunakan untuk pengembangan API.
h) Locust digunakan untuk pengujian API dalam layanan *cloud*.