```
SET in_learn_rate TO 1e-4

SET epochs TO 20

SET batch_size TO 32

SET file_model TO "projModel.h5"


SET Direktori TO "D:\Kuliah\COVID19_RT_FaceMaskDetector-main- comvis\dataset"

SET Kategori TO ["with_mask","without_mask"]


OUTPUT("[INFO] Loading Gambar")


SET data TO []

SET sample TO []
```

```
FOR category IN Kategori:

    SET jejak TO os.path.join(Direktori,category)

    FOR img IN os.listdir(jejak):

        SET jejak_gmbr TO os.path.join(jejak,img)

        SET gambar TO load_img(jejak_gmbr,target_size=(224,224))

        SET gambar TO img_to_array(gambar)

        SET gambar TO preprocess_INPUT(gambar)

        data.append(gambar)

        sample.append(category)
```

```
SET labl TO LabelBinarizer()

SET sample TO labl.fit_transform(sample)

SET sample TO to_categorical(sample)

SET data TO nump.array(data, dtype="float32")

SET sample TO nump.array(sample)

SET (ujiX, tesX, ujiY, tesY) TO train_test_split(data, sample,test_size=0.20, stratify=sample, random_state=42)



SET augmentasi TO ImageDataGenerator(

        rotation_range=20, #

        zoom_range=0.15,

        width_shift_range=0.2,
    height_shift_range=0.2,

    shear_range=0.15,

    horizontal_flip=True,

    fill_mode="nearest")


SET BasicModel TO MobileNetV2(weights="imagenet",

         include_top=False,

         INPUT_tensor=Input(shape=(224, 224, 3)))


SET ModelKepala TO BasicModel.output

SET ModelKepala TO AveragePooling2D(pool_size=(7, 7),strides TO 2)(ModelKepala)

SET ModelKepala TO Flatten(name="flatten")(ModelKepala)
SET ModelKepala TO Dense(128, activation="relu")(ModelKepala)

SET ModelKepala TO Dense(64, activation="relu")(ModelKepala)

SET ModelKepala TO Dropout(0.5)(ModelKepala)

SET ModelKepala TO Dense(2, activation="softmax")(ModelKepala)


SET model TO Model(INPUTs=BasicModel.INPUT, outputs=ModelKepala)


FOR layer IN BasicModel.layers:

    SET layer.trainable TO False


OUTPUT("[INFO] Compile Model")
```

```
SET outpt TO Adam(lr=in_learn_rate, decay=in_learn_rate / epochs)

model.compile(loss="binary_crossentropy", optimizer=outpt,

        metrics=["accuracy"])



OUTPUT("[INFO] Training data")

SET train TO model.fit(

        augmentasi.flow(ujiX, ujiY, batch_size=batch_size),

        steps_per_epoch=len(ujiX) // batch_size,

        validation_data=(tesX, tesY),

        validation_steps=len(tesX) // batch_size,

   callbacks=[EarlyStopping(patience=2)],

        epochs=epochs)
```

```
OUTPUT("[INFO] Laporan Predict")

SET predict_model TO model.predict(tesX, batch_size=batch_size)

SET predict_model TO nump.argmax(predict_model, axis=1)

OUTPUT(classification_report(tesY.argmax(axis=1), predict_model,
    target_names=labl.classes_))

OUTPUT("[INFO] Menyimpan Model Deteksi Masker")
model.save(file_model)
```

```
SET value TO len(train.history['loss'])

shw.style.use("ggplot")

shw.figure()

shw.plot(nump.arange(0, value), train.history["loss"], label="Uji_loss")

shw.plot(nump.arange(0, value), train.history["val_loss"], label="nilai_loss")

shw.plot(nump.arange(0, value), train.history["accuracy"], label="Uji_accuracy")

shw.plot(nump.arange(0, value), train.history["val_accuracy"], label="nilai_accuracy")

shw.title("Pengujian Loss dan Akurasi")

shw.xlabel("Nilai Epoch")

shw.ylabel("Loss/Akurasi")

shw.legend(loc="lower left")
shw.savefig("plot",dpi=800)
```