LAMPIRAN

Lampiran 1. Surat Keterangan Penelitian

PEMERINTAH KABUPATEN BULELENG
DINAS PENANAMAN MODAL DAN PELAYANAN
TERPADU SATU PINTU
Lantai 3 Pasar Banyuasri, Kelurahan Banyuasri, Kecamatan Buleleng
Telp. (0362) 22063 Singaraja
Alamat e-mail : dpmptsp@bulelengkab.go.id
Website : dpmptsp.bulelengkab.go.id

Nomor : 503/042/REK/DPMPTSP/2026
Lamp :
Perihal : **Surat Keterangan Penelitian**

Kepada :
Yth. Direktur Perumda Pasar Argha Nayottama Buleleng

di-
Tempat

I. Dasar :
1. Peraturan Menteri Dalam Negeri RI Nomor : 3 Tahun 2018 tentang Penerbitan Surat Keterangan Penelitian;
2. Peraturan Menteri Dalam Negeri RI Nomor : 138 Tahun 2017 tentang Penyelenggaraan Pelayanan Terpadu Satu Pintu Daerah;
3. Surat dari Wakil Dekan Bidang Akademik Fak. Teknik dan Kejuruan Undiksha Singaraja Nomor 158/UN48.11.1/DI.03.00/2025 Tanggal 20 Januari 2026 Perihal Surat Permohonan Pengambilan Data

II. Setelah mempelajari dan meneliti rencana kegiatan yang diajukan, maka dapat diberikan Surat Keterangan Penelitian Kepada :
Nama : Gusti Nyoman Satya Sudnarsan
NIK : 5102051806040002
Pekerjaan : Mahasiswa
Alamat : Banjar Sudimara Kaja, Desa Sudimara, Kecamatan Tabanan
Bidang / Judul : Klasifikasi Jenis-Jenis Ikan Konsumsi di Pasar Banyuasri Singaraja Menggunakan Convolutional Neural Network dengan Arsitektur VGG-16 dan MobileNetV2
Jumlah Peserta : 1 orang
Lokasi : Pasar Banyuasri Singaraja
Lamanya : 6 bulan (26 Januari - 30 Juni 2026)

III. Dalam melakukan kegiatan agar yang bersangkutan mematuhi ketentuan sebagai berikut :
1. Sebelum mengadakan kegiatan agar melapor kepada Kepala Dinas Penanaman Modal dan PTSP Kabupaten Buleleng atau Pejabat yang Berwenang;
2. Tidak dibenarkan melakukan kegiatan yang tidak ada kaitannya dengan bidang/ judul dimaksud, apabila melanggar ketentuan akan dicabut ijinnya dan menghentikan segala kegiatannya;
3. Menaati segala ketentuan perundang-undangan yang berlaku serta mengindahkan adat istiadat dan budaya setempat;
4. Apabila masa berlaku Surat Keterangan Penelitian / Ijin ini telah berakhir, sedangkan pelaksanaan kegiatan belum selesai maka perpanjangan Surat Keterangan Penelitian / Ijin agar ditujukan kepada Instansi pemohon;

Demikian Surat Keterangan Penelitian ini dibuat untuk dipergunakan sebagaimana mestinya.

DITETAPKAN : SINGARAJA
PADA TANGGAL : 20 Januari 2026

Ditandatangani secara elektronik oleh
Kepala Dinas
Penanaman Modal dan Pelayanan
Terpadu Satu Pintu
Gede Ngurah Dharma Seputra, ST, M.AP
Pembina Tingkat I (IV/b)
NIP. 197405282006121006

Tembusan ini disampaikan kepada Yth:
1. Kepala Dinas Penanaman Modal dan PTSP Prov. Bali
2. Kepala Badan Kesatuan Bangsa dan Politik Kabupaten Buleleng
3. Yang Bersangkutan

Dokumen ditandatangani secara elektronik menggunakan sertifikat Elektronik yang diterbitkan oleh Balai Sertifikasi Elektronik, Badan Siber Dan Sandi Negara

Lampiran 2. Dokumentasi Pengambilan Data

Lampiran 3. *Source Code Preprocessing*

```
import tensorflow as tf

import numpy as np

import pandas as pd

from tensorflow.keras.preprocessing.image import
ImageDataGenerator

from tensorflow.keras.callbacks import ReduceLROnPlateau

from tensorflow.keras.applications.mobilenet_v2 import
preprocess_input

from sklearn.metrics import classification_report,
confusion_matrix, accuracy_score


from google.colab import drive

drive.mount('/content/drive')


dataset_path = "/content/drive/MyDrive/dataset_ikan"

train_path = dataset_path + "/train"

test_path = dataset_path + "/test"


img_size = (224, 224)

batch_size = 16


train_datagen = ImageDataGenerator(

    preprocessing_function=preprocess_input,

    rotation_range=20,

    width_shift_range=0.1,

    height_shift_range=0.1,

    zoom_range=0.2,

    shear_range=0.1,
```

```
    horizontal_flip=True,

    brightness_range=[0.85, 1.15],

    fill_mode='nearest',

    validation_split=0.2

)


test_datagen =
ImageDataGenerator(preprocessing_function=preprocess_input
)


train_generator = train_datagen.flow_from_directory(

    train_path, target_size=img_size,
batch_size=batch_size,

    class_mode='categorical', subset='training'

)

val_generator = train_datagen.flow_from_directory(

    train_path, target_size=img_size,
batch_size=batch_size,

    class_mode='categorical', subset='validation'

)

test_generator = test_datagen.flow_from_directory(

    test_path, target_size=img_size,
batch_size=batch_size,

    class_mode='categorical', shuffle=False

)

class_names = list(train_generator.class_indices.keys())

num_classes = len(class_names)
```

Lampiran 4. *Source Code* Pembentukan Model *VGG-16*

```
from tensorflow.keras.applications import VGG16
from tensorflow.keras import layers, models
import tensorflow as tf

base_model = VGG16(
    weights='imagenet',
    include_top=False,
    input_shape=(224, 224, 3)
)

for layer in base_model.layers:
    layer.trainable = False

x = base_model.output
x = layers.GlobalAveragePooling2D()(x)
x = layers.BatchNormalization()(x)
x = layers.Dense(256, activation='relu')(x)
x = layers.Dropout(0.5)(x)
output = layers.Dense(num_classes,
activation='softmax')(x)

model = models.Model(inputs=base_model.input,
outputs=output)

model.compile(
    optimizer=tf.keras.optimizers.Adam(1e-4),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

model.summary()
```

Lampiran 5. *Source Code* Pelatihan Model *VGG-16*

```
history = model.fit(
    train_generator,
    validation_data=val_generator,
    epochs=20,
    callbacks=callbacks,
    class_weight=class_weights
)

for layer in base_model.layers[:-8]:
    layer.trainable = False

for layer in base_model.layers[-8:]:
    layer.trainable = True

model.compile(
    optimizer=tf.keras.optimizers.Adam(1e-5),
    loss='categorical_crossentropy',
    metrics=['accuracy']
)

history_ft = model.fit(
    train_generator,
    validation_data=val_generator,
    epochs=25,
    callbacks=callbacks,
    class_weight=class_weights
)
```

Lampiran 6. *Source Code* Pembentukan Model *MobileNetV2*

```
import tensorflow as tf
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras.layers import (
Dense, GlobalAveragePooling2D, Dropout, BatchNormalization
)
from tensorflow.keras.models import Model

base_model = MobileNetV2(
    weights='imagenet',
    include_top=False,
    input_shape=(224, 224, 3)
)

for layer in base_model.layers:
    layer.trainable = False

x = base_model.output
x = GlobalAveragePooling2D()(x)
x = BatchNormalization()(x)
x = Dense(128, activation='relu')(x)
x = Dropout(0.5)(x)
output = Dense(num_classes, activation='softmax')(x)

model = Model(inputs=base_model.input, outputs=output)

loss_fn =
tf.keras.losses.CategoricalCrossentropy(label_smoothing=0.
05)

model.compile(
    optimizer=tf.keras.optimizers.Adam(1e-4),
    loss=loss_fn,
    metrics=['accuracy']
)

model.summary()
```

Lampiran 7. *Source Code* Pelatihan Model *MobileNetV2*

```
history = model.fit(
    train_generator,
    validation_data=val_generator,
    epochs=20,
    callbacks=callbacks,
    class_weight=class_weights
)


for layer in base_model.layers[-120:]:
    layer.trainable = True

model.compile(
    optimizer=tf.keras.optimizers.Adam(3e-5),
    loss=loss_fn,
    metrics=['accuracy']
)

history_ft = model.fit(
    train_generator,
    validation_data=val_generator,
    epochs=25,
    callbacks=callbacks,
    class_weight=class_weights
)
```

Lampiran 8. *Source Code* Evaluasi Model

```
import pandas as pd
from sklearn.metrics import classification_report,
accuracy_score

y_pred_prob = model.predict(test_generator)
y_pred = y_pred_prob.argmax(axis=1)
y_true = test_generator.classes

acc = accuracy_score(y_true, y_pred)

report_dict = classification_report(
    y_true,
    y_pred,
    target_names=class_names,
    output_dict=True
)

precision = report_dict['weighted avg']['precision']
recall = report_dict['weighted avg']['recall']
f1 = report_dict['weighted avg']['f1-score']

hasil = pd.DataFrame({
    "Metric": ["Accuracy", "Precision", "Recall", "F1-
Score"],
    "Score": [
        round(acc, 4),
        round(precision, 4),
        round(recall, 4),
        round(f1, 4)
    ]
})

print("HASIL EVALUASI MODEL")
print("=" * 50)
print(hasil.to_string(index=False))
```

Lampiran 9. *Source Code* Implementasi Model (*app.py*)

```
from flask import Flask, render_template, request
import numpy as np
import json
from tensorflow.keras.models import load_model
from PIL import Image

app = Flask(__name__)
model = load_model("model.h5")
with open("labels.json", "r") as f:
    labels = json.load(f)

def preprocess_image(image):
    image = image.resize((224, 224))
    image = np.array(image) / 255.0
    return np.expand_dims(image, axis=0)

@app.route("/", methods=["GET", "POST"])
def index():
    result, confidence, predicted_image = None, 0, None
    all_results = []
    if request.method == "POST":
        try:
            file = request.files["image"]
            image = Image.open(file).convert("RGB")
            img = preprocess_image(image)
            prediction = model.predict(img)[0]
            index_pred = np.argmax(prediction)
```

```
            result = labels[index_pred]
            confidence = float(prediction[index_pred] *
100)
            for i, prob in enumerate(prediction):
                all_results.append({
                    "class": labels[i],
                    "confidence": round(float(prob * 100),
2),
                    "image":
f"/static/reference/{labels[i]}.jpg"
                })
            all_results = sorted(
                all_results, key=lambda x:
x["confidence"], reverse=True
            )
            predicted_image =
f"/static/reference/{result}.jpg"
        except Exception as e:
            result, confidence = f"Error: {str(e)}", 0
    return render_template(
        "index.html", result=result,
        confidence_percent=round(confidence, 2),
        all_results=all_results,
predicted_image=predicted_image
    )


if __name__ == "__main__":
    app.run(debug=True)
```