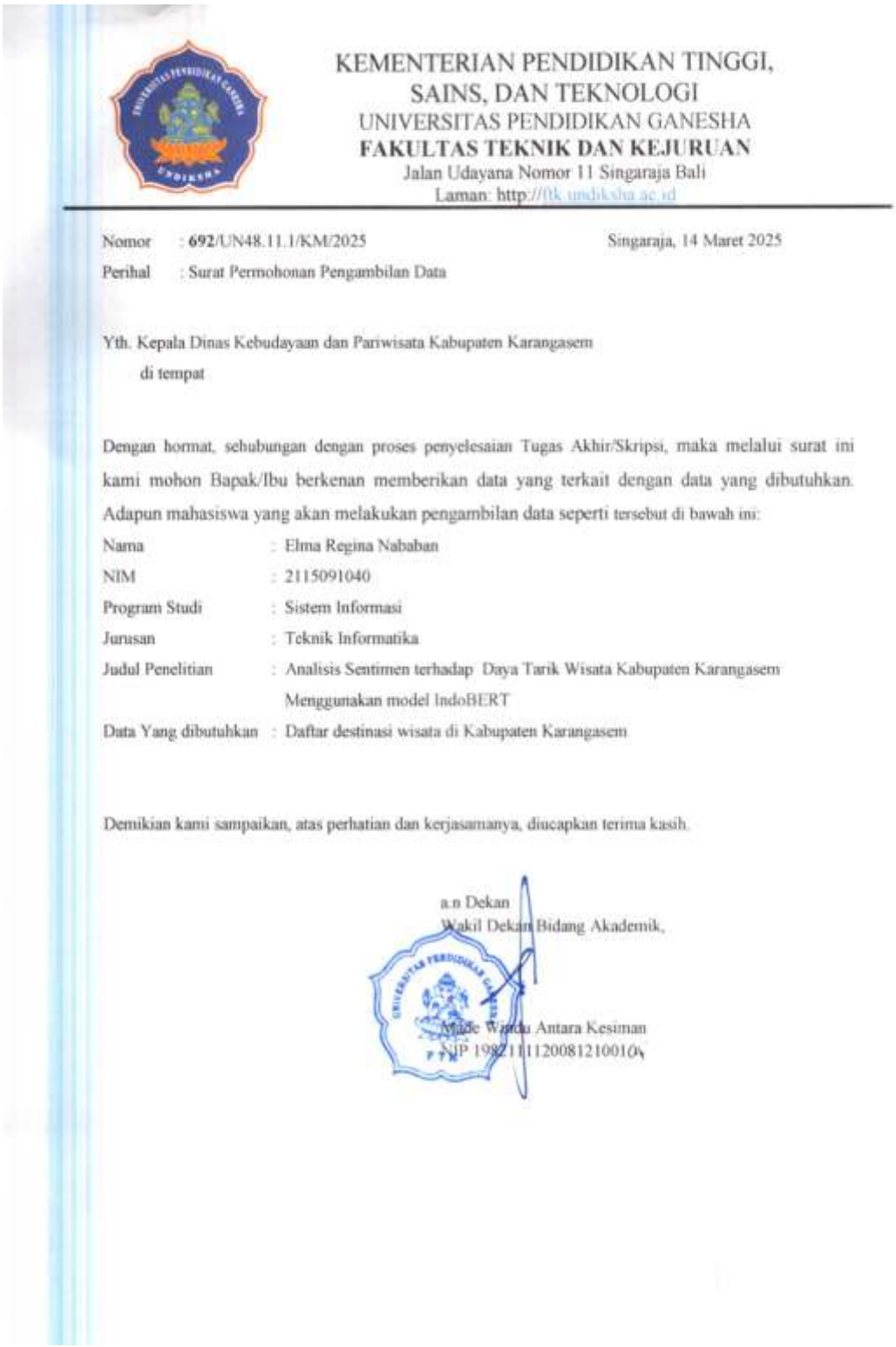
Lampiran 1. Riwayat Hidup

RIWAYAT HIDUP

Elma Regina Nababan lahir di Burau, Luwu Timur pada tanggal 7 April 2003. Penulis lahir dari pasangan suami istri Bapak Humuntal Nababan dan Ibu Nurita Sinaga. Penulis berkebangsaan Indonesia dan beragama Kristen. Penulis beralamat di Desa Bungapati, Kecamatan Tana Lili, Kabupaten Luwu Utara, Provinsi Sulawesi Selatan. Penulis menyelesaikan pendidikan Sekolah Dasar di SD Negeri 107 Lagego dan lulus pada tahun 2015. Kemudian penulis melanjutkan Sekolah Menengah Pertama di SMP Negeri 2 Bone-Bone dan lulus pada tahun 2018. Pada tahun 2021, penulis lulus dari Sekolah Menengah Atas di SMA Negeri 4 Luwu Utara dan melanjutkan ke studi (S1) di Universitas Pendidikan Ganesha dengan mengambil Program Studi Sistem Informasi, Jurusan Teknik Informatika.

Lampiran 2. Permohonan izin penggunaan data Dinas Pariwisata



KEMENTERIAN PENDIDIKAN TINGGI,
SAINS, DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
FAKULTAS TEKNIK DAN KEJURUAN
Jalan Udayana Nomor 11 Singaraja Bali
Laman: http://ftk.undiksha.ac.id

Nomor : **692**/UN48.11.1/KM/2025 Singaraja, 14 Maret 2025
Perihal : Surat Permohonan Pengambilan Data

Yth. Kepala Dinas Kebudayaan dan Pariwisata Kabupaten Karangasem
di tempat

Dengan hormat, sehubungan dengan proses penyelesaian Tugas Akhir/Skripsi, maka melalui surat ini kami mohon Bapak/Ibu berkenan memberikan data yang terkait dengan data yang dibutuhkan. Adapun mahasiswa yang akan melakukan pengambilan data seperti tersebut di bawah ini:

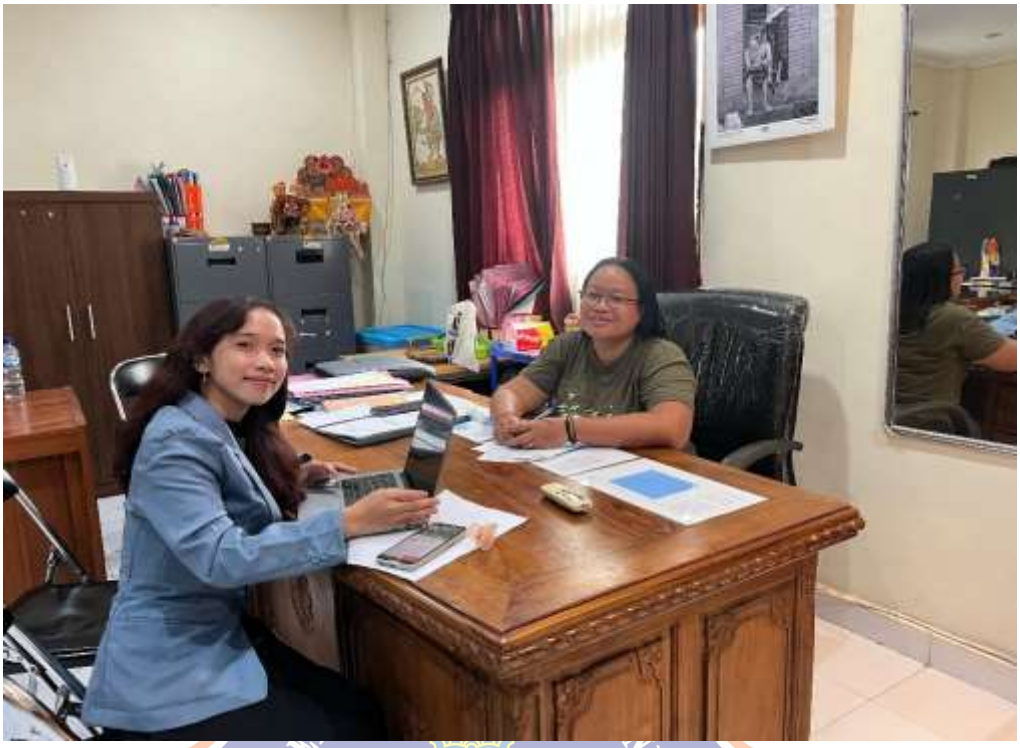
Nama	:	Elma Regina Nababan
NIM	:	2115091040
Program Studi	:	Sistem Informasi
Jurusan	:	Teknik Informatika
Judul Penelitian	:	Analisis Sentimen terhadap Daya Tarik Wisata Kabupaten Karangasem Menggunakan model IndoBERT
Data Yang dibutuhkan	:	Daftar destinasi wisata di Kabupaten Karangasem

Demikian kami sampaikan, atas perhatian dan kerjasamanya, diucapkan terima kasih.

a.n Dekan
Wakil Dekan Bidang Akademik,

I Made Windu Antara Kesiman
NIP 198211112008121001

Lampiran 3. Wawancara Bidang Destinasi Dinas Pariwisata Karangasem



Keterangan : Wawancara dilakukan dengan Ibu Anna Christiana, SS.,MAP selaku Kepala Bidang Destinasi dan Pemasaran Pariwisata. Dokumentasi diambil pada tanggal 17 Maret 2025 yang beertempat di Ruang Bidang Destinasi dan Pemasaran Pariwisata Kabupaten Karangasem

Lampiran 4. List Daftar Pariwisata Kab. Karangasem pada *keyword*

NO	NAMA DAYA TARIK WISATA	JENIS	LOKASI
1	DTW Taman Soekasada Ujung	Wisata Budaya	Kecamatan Karangasem
2	DTW Pesona Bukit Asah Bali	Wisata Alam	Kecamatan Karangasem
3	DTW Pantai Pasir Putih/ Virgin Beach	Wisata Alam	Kecamatan Karangasem
4	DTW Pantai Jasi	Wisata Alam	Kecamatan Karangasem
5	DTW Candidasa	Wisata Alam	Kecamatan Karangasem
6	DTW Museum Lontar Dukuh Penaban	Wisata Budaya	Kecamatan Karangasem
7	DTW Tenganan Pegringsingan	Wisata Budaya	Kecamatan Manggis
8	DTW Padangbai	Wisata Alam	Kecamatan Manggis
9	DTW Tenganan Dauh Tukad	Wisata Budaya	Kecamatan Manggis
10	DTW Pantai Blue Lagoon	Wisata Alam	Kecamatan Manggis
11	DTW Pantai Wates Yeh Malet	Wisata Alam	Kecamatan Manggis
12	DTW Jemeluk	Wisata Alam	Kecamatan Abang
13	DTW Pesona Bukit Lempuyang	Wisata Alam	Kecamatan Abang
14	DTW Taman Tirta Gangga	Wisata Budaya	Kecamatan Abang
15	DTW Maha Gangga Valley	Wisata Alam	Kecamatan Abang
16	DTW Pesona Alam Kastala	Wisata Alam	Kacamatan Bebandem
17	DTW Agrowisata Salak Sibetan	Wisata Alam	Kacamatan Bebandem
18	DTW Bukit Surga	Wisata Alam	Kacamatan Bebandem
19	DTW Putung	Wisata Alam	Kecamatan Selat
20	DTW Pesona Alam Jaga Satru	Wisata Alam	Kecamatan Selat

NO	NAMA DAYA TARIK WISATA	JENIS	LOKASI
21	DTW Lingkungan Pura Agung Besakih	Wisata Budaya	Kecamatan Rendang
22	DTW Taman Edelweis Bali	Wisata Alam	Kecamatan Rendang
23	DTW Telaga Surya	Wisata Alam	Kecamatan Rendang
24	DTW Pesona Alam Sangkan Gunung	Wisata Alam	Kecamatan Sidemen
25	DTW Air Terjun Gembleng	Wisata Alam	Kecamatan Sidemen
26	DTW Pesona Alam Munti Gunung	Wisata Alam	Kecamatan Kubu
27	DTW Pantai Amed	Wisata Alam	Kecamatan Amed Jemeluk
28	DTW Pantai Bias Tugel	Wisata Alam	Kecamatan Manggis

Lampiran 5. Surat Ketersediaan Pelabelan Data

KEMENTERIAN PENDIDIKAN TINGGI, SAINS, DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
FAKULTAS TEKNIK DAN KEJURUAN
Jalan Udayana Nomor 11 Singaraja - Bali Kode Pos 81116
Telepon (0362) 22570 Email: ftk@undiksha.ac.id Laman: http://ftk.undiksha.ac.id

Nomor : **2489**/UN48.11.1/DL03.00/2025 Singaraja, 4 September 2025
Perihal : Surat Permohonan Pengambilan Data

Yth. Kepala SMK Negeri 1 Sukasada
di tempat

Dengan hormat, sehubungan dengan proses penyelesaian Tugas Akhir/Skripsi, maka melalui surat ini kami mohon Bapak/Ibu berkenan memberikan data yang terkait dengan data yang dibutuhkan. Adapun mahasiswa yang akan melakukan pengambilan data seperti tersebut di bawah ini:

Nama	:	Elma Regina Nababan
NIM	:	2115091040
Program Studi	:	Sistem Informasi
Jurusan	:	Teknik Informatika
Data yang dibutuhkan	:	Mengisi label sentimen (positif, negatif dan netral) pada Komentar yang ada di formulir yang telah diberikan
Judul Penelitian	:	Analisis Sentimen Terhadap Daya Tarik Wisata Kabupaten Karangasem Menggunakan Model IndoBERT

Demikian kami sampaikan, atas perhatian dan kerjasamanya, diucapkan terima kasih.

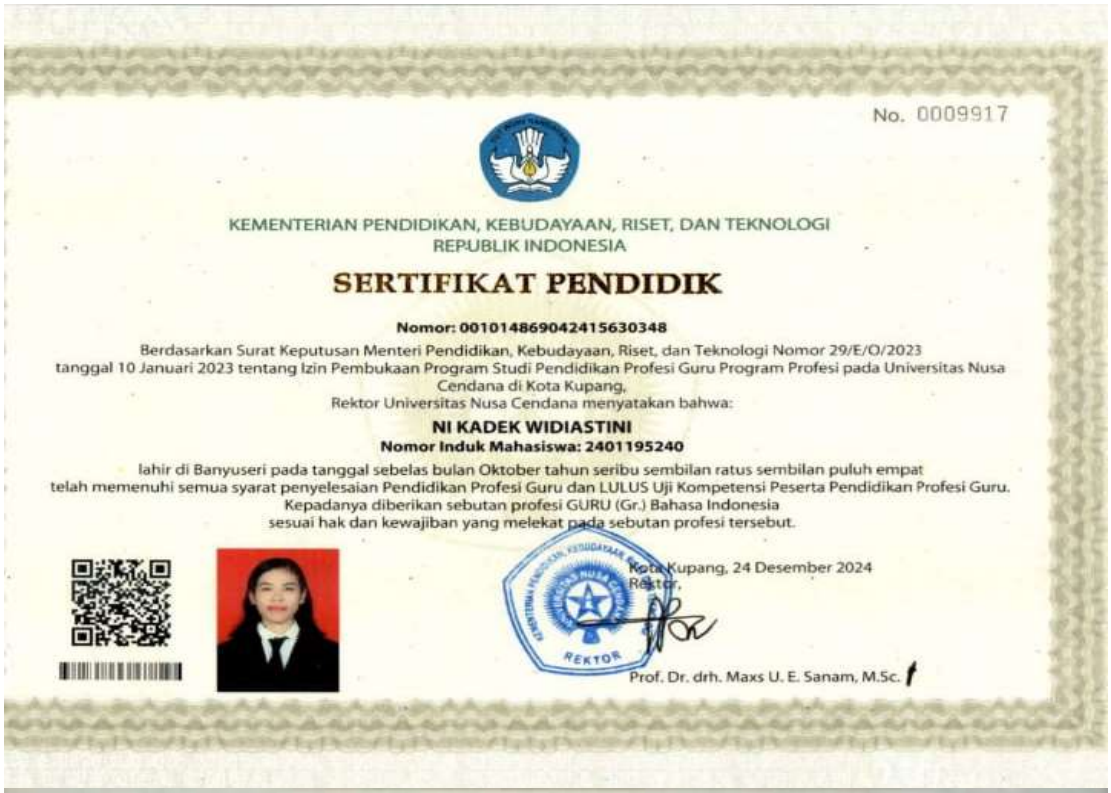
a.n Dekan
Wakil Dekan Bidang Akademik,

Made Windu Antara Kesiman
NIP 198211112008121001

Balai Sertifikasi Elektronik

Catatan :
- UU ITE No. 11 Tahun 2008 Pasal 5 ayat 1 "Informasi Elektronik dan/atau Dokumen Elektronik dan/atau hasil cetaknya merupakan alat bukti hukum yang sah"
- Dokumen ini tertanda ditandatangani secara elektronik menggunakan sertifikat elektronik yang diterbitkan BsrE
- Surat ini dapat dibuktikan keasliannya dengan menggunakan *qr code* yang telah tersedia

Lampiran 6. Lampiran 6. Sertifikat Pendidik



No. 0009917

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI
REPUBLIK INDONESIA

SERTIFIKAT PENDIDIK

Nomor: 00101486904241563034 8

Berdasarkan Surat Keputusan Menteri Pendidikan, Kebudayaan, Riset, dan Teknologi Nomor 29/E/O/2023 tanggal 10 Januari 2023 tentang Izin Pembukaan Program Studi Pendidikan Profesi Guru Program Profesi pada Universitas Nusa Cendana di Kota Kupang,
Rektor Universitas Nusa Cendana menyatakan bahwa:

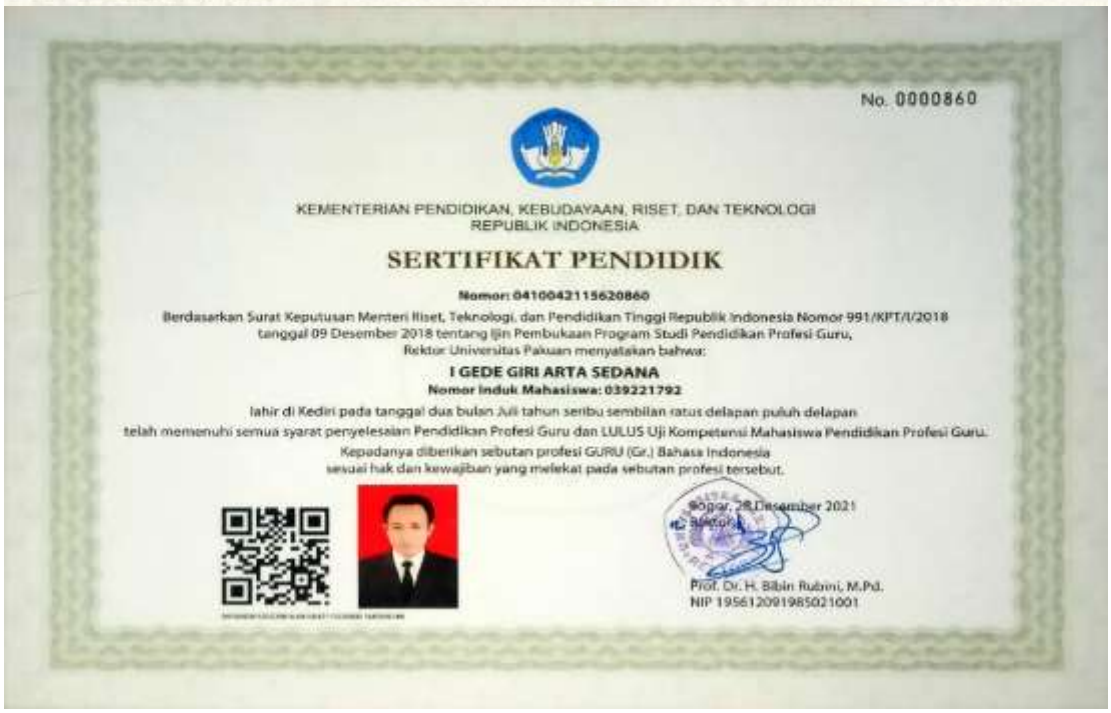
NI KADEK WIDIASTINI
Nomor Induk Mahasiswa: 2401195240

lahir di Banyuseri pada tanggal sebelas bulan Oktober tahun seribu sembilan ratus sembilan puluh empat telah memenuhi semua syarat penyelesaian Pendidikan Profesi Guru dan LULUS Uji Kompetensi Peserta Pendidikan Profesi Guru.
Kepadanya diberikan sebutan profesi GURU (Gr.) Bahasa Indonesia
sesuai hak dan kewajiban yang melekat pada sebutan profesi tersebut.

Kota Kupang, 24 Desember 2024
Rektor,

REKTOR

Prof. Dr. drh. Maxs U. E. Sanam, M.Sc.



No. 0000860

KEMENTERIAN PENDIDIKAN, KEBUDAYAAN, RISET, DAN TEKNOLOGI
REPUBLIK INDONESIA

SERTIFIKAT PENDIDIK

Nomor: 0410042115620860

Berdasarkan Surat Keputusan Menteri Riset, Teknologi, dan Pendidikan Tinggi Republik Indonesia Nomor 991/KPT/I/2018 tanggal 09 Desember 2018 tentang Ijin Pembukaan Program Studi Pendidikan Profesi Guru,
Rektor Universitas Pakuan menyatakan bahwa:

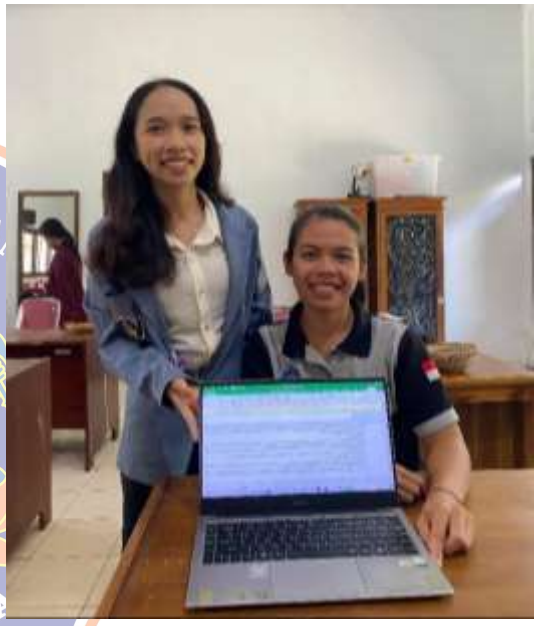
I GEDE GIRI ARTA SEDANA
Nomor Induk Mahasiswa: 039221792

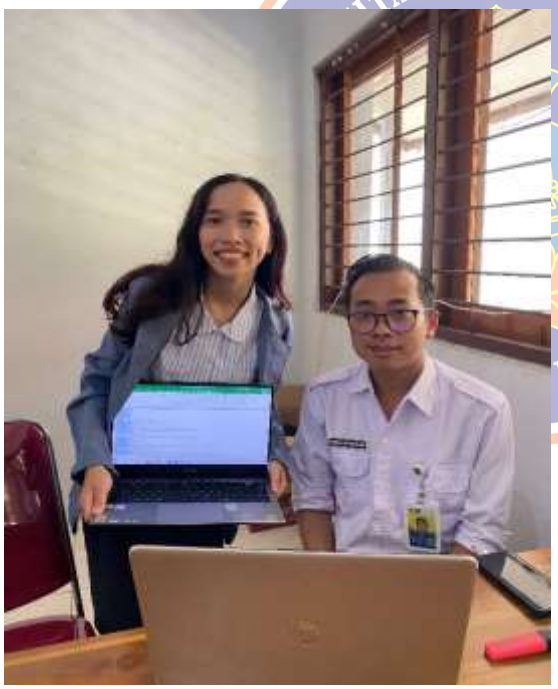
lahir di Kediri pada tanggal dua bulan Juli tahun seribu sembilan ratus delapan puluh delapan
telah memenuhi semua syarat penyelesaian Pendidikan Profesi Guru dan LULUS Uji Kompetensi Mahasiswa Pendidikan Profesi Guru.
Kepadanya diberikan sebutan profesi GURU (Gr.) Bahasa Indonesia
sesuai hak dan kewajiban yang melekat pada sebutan profesi tersebut.

Bogor, 28 Desember 2021
Rektor,

Prof. Dr. H. Bibin Rubini, M.Pd.
NIP 195612091985021001

Lampiran 7. Dokumentasi bersama Validator





Lampiran 8. Kode *Pre-Processing* Data

1. ***Cleaning***

```
import pandas as pd
import re
import unicodedata
import emoji
# Load data
df = pd.read_csv('/content/data.csv')
# Fungsi cleaning
def clean_text(text):
    text = str(text).lower()
    text = unicodedata.normalize("NFKC", text)
    # hapus URL, mention, emoji
    text = re.sub(r'http\S+|www\.\S+', ' ', text)
    text = re.sub(r'@\w+|#\w+', ' ', text)
    text = emoji.replace_emoji(text, replace=' ')
    # hapus angka & tanda baca
    text = re.sub(r'\d+', ' ', text)
    text = re.sub(r'[^\w\s]', ' ', text)
    # hapus spasi berlebih
    text = re.sub(r'\s+', ' ', text).strip()
    return text
# Terapkan cleaning
df = df.dropna(subset=['text'])
df['cleaning_data'] =
df['text'].apply(clean_text)
df.to_csv('hasil_cleaning.csv', index=False)
```

2. ***Case Folding***

```
# 1. Import library pandas
# PROSES CASE FOLDING
# 2. Membaca dataset final_cleaning_data.csv
df = pd.read_csv('/content/final_cleaning.csv')
def case_folding(text):
    return str(text).lower()
df['case_folding'] =
df['cleaning_data'].apply(case_folding)
df.to_csv('data_casefolding.csv', index=False)
from IPython.display import display
print("\n=== TABEL HASIL CASEFOLDING (SEBELUM &
SESUDAH) ===")
display(df[['cleaning_data','case_folding']].head(10))
```

3. *Stopword Removal*

```
import pandas as pd
from nltk.corpus import stopwords
import nltk
nltk.download('stopwords')
# Load data
df = pd.read_csv('/content/data_casefolding.csv')
# Stopword + pengecualian
stop_words = set(stopwords.words('indonesian')) - {
    "tidak","bukan","kurang","lebih",
    "sangat","cukup","terlalu",
    "belum","masih","jangan"
}
# Fungsi stopword removal
def remove_stopwords(text):
    return " ".join([w for w in
str(text).lower().split() if w not in stop_words])
# Terapkan & simpan
df['stopword_removal'] =
df['case_folding'].apply(remove_stopwords)
df.to_csv('data_stopword_removal.csv', index=False)
```

4. *Tokenize*

```
import pandas as pd
from IPython.display import display
df = pd.read_csv('/content/data_stopword_removal.csv')
df['stopword_removal'] =
df['stopword_removal'].fillna('')
def tokenize(text):
    if not isinstance(text, str):
        return ""
    tokens = text.split()
    return " ".join(tokens)
df['tokenizing'] =
df['stopword_removal'].apply(tokenize)
df.to_csv('hasil_tokenizing.csv', index=False)
print("✅ Tokenizing selesai")
display(df[['stopword_removal',
'tokenizing']].head(10))
```

5. *Normalize*

```
import pandas as pd
# Load data hasil tokenizing
df = pd.read_csv('/content/hasil_tokenizing.csv')
df['tokenizing'] = df['tokenizing'].fillna('')
# Kamus normalisasi
norm_dict = {
"gk":"tidak","ga":"tidak","nggak":"tidak","tdk":"tidak
",
"bgt":"banget","bgtt":"banget","dgn":"dengan","krn":"k
arena","dr":"dari","yg":"yang","utk":"untuk","sm":"sam
a","aja":"saja","dll":"","dsb":""
}
# Fungsi normalisasi
def normalize(text):
    return " ".join([norm_dict.get(w, w) for w in
text.split()])
# Terapkan & simpan
df['normalization'] =
df['tokenizing'].apply(normalize)
df.to_csv('hasil_normalization.csv', index=False)
```

6. *Stemming*

```
import pandas as pd
from Sastrawi.Stemmer.StemmerFactory import
StemmerFactory
# Inisialisasi stemmer
stemmer = StemmerFactory().create_stemmer()
# Load data
df = pd.read_csv('/content/hasil_normalization.csv')
df['normalization'] = df['normalization'].fillna('')
# Fungsi stemming
def stem_text(text):
    return " ".join([
        word if word in exception_words else
stemmer.stem(word)
        for word in str(text).split()
    ])
# Terapkan & simpan
df['stemming'] = df['normalization'].apply(stem_text)
df.to_csv('hasil_stemming.csv', index=False)
```

Lampiran 9. Kode *Downsampling* Data

```python
import pandas as pd
from sklearn.utils import resample

# Load data
df = pd.read_csv("/content/hasil_stemming.csv")

# Pisahkan kelas
df_neg = df[df["Label"] == "N"]
df_net = df[df["Label"] == "O"]
df_pos = df[df["Label"] == "P"]

# Tentukan jumlah minimum
min_size = min(len(df_neg), len(df_net), len(df_pos))

# Downsampling
df_neg = resample(df_neg, replace=False,
n_samples=min_size, random_state=42)
df_net = resample(df_net, replace=False,
n_samples=min_size, random_state=42)
df_pos = resample(df_pos, replace=False,
n_samples=min_size, random_state=42)

# Gabungkan & acak
df_down = pd.concat([df_neg, df_net, df_pos]) \
            .sample(frac=1, random_state=42) \
            .reset_index(drop=True)

# Simpan
df_down.to_csv("/content/data_downsampling_only.csv",
index=False)
```

Lampiran 10. Kode *Easy Data Augmentation* (EDA)

```
import pandas as pd
import random
import numpy as np
from collections import Counter

# KONFIGURASI
INPUT_PATH  = "/content/hasil_stemming.csv"
OUTPUT_PATH = "/content/hasil_stemming_eda.csv"

TEXT_COL  = "cleaning_data"
LABEL_COL = "Label"

AUGMENT_RATIO = 0.5
SEED = 42

random.seed(SEED)
np.random.seed(SEED)

# LOAD DATA
df = pd.read_csv(INPUT_PATH)
df = df[[TEXT_COL, LABEL_COL]].dropna()
df = df[df[LABEL_COL].isin(["P", "N",
"O"])].reset_index(drop=True)
df.columns = ["text", "label"]

# KAMUS SINONIM
SYN_DICT = {
    "indah": ["cantik", "menawan"],
    "bagus": ["mantap", "oke"],
    "buruk": ["jelek", "kurang baik"],
    "mahal": ["cukup mahal", "harga tinggi"],
    "murah": ["terjangkau", "ramah kantong"],
    "ramai": ["padat", "banyak orang"],
    "tenang": ["damai", "hening"],
    "kotor": ["jorok", "kurang bersih"],
}

# FUNGSI EDA
def synonym_replacement(words, p=0.3):
    return [
        random.choice(SYN_DICT[w]) if w in SYN_DICT
and random.random() < p else w
        for w in words
    ]

def random_deletion(words, p=0.1):
    if len(words) <= 3:
        return words
    new_words = [w for w in words if random.random() >
p]
    return new_words if new_words else words

def random_swap(words):
    if len(words) < 2:
        return words
    i, j = random.sample(range(len(words)), 2)
    words[i], words[j] = words[j], words[i]
    return words

def eda_transform(text):
    words = text.split()
    if not words:
        return text
```

```
    op = random.choice([
        lambda x: synonym_replacement(x),
        lambda x: random_deletion(x),
        lambda x: random_swap(x)
    ])
    return " ".join(op(words))

# AUGMENTASI (kelas minoritas)
augmented_rows = []

for label in ["N", "O"]:
    subset = df[df["label"] == label]
    n_aug = int(len(subset) * AUGMENT_RATIO)
    for _ in range(n_aug):
        row = subset.sample(1).iloc[0]
        new_text = eda_transform(row["text"])

        if new_text != row["text"]:
            augmented_rows.append({
                "text": new_text,
                "label": label
            })
# GABUNG & SIMPAN
df_aug = pd.concat([df, pd.DataFrame(augmented_rows)],
ignore_index=True)

df_aug = df_aug.rename(columns={
    "text": TEXT_COL,
    "label": LABEL_COL
})

df_aug.to_csv(OUTPUT_PATH, index=False)
```

Lampiran 11. Kode *Downsampling* dan EDA

```
import pandas as pd
import random
import numpy as np
from collections import Counter

# KONFIGURASI
INPUT_PATH  = "/content/hasil_stemming.csv"
OUTPUT_PATH =
"/content/hasil_stemming_eda_downsampling.csv"

TEXT_COL  = "stemming"
LABEL_COL = "Label"

EDA_RATIO = 0.5
MAX_RATIO_P = 1.5
SEED = 42

random.seed(SEED)

np.random.seed(SEED)

# LOAD DATA
df = pd.read_csv(INPUT_PATH)
df = df[[TEXT_COL, LABEL_COL]].dropna()
df = df[df[LABEL_COL].isin(["P", "N",
"O"])].reset_index(drop=True)
df.columns = ["text", "label"]

# Penanda data asli
df["data_type"] = "real"

# KAMUS SINONIM
SYN_DICT = {
    "indah": ["cantik", "menawan"], "bagus":
["mantap", "baik"], "buruk": ["jelek", "kurang"],
"mahal": ["harga tinggi"], "murah": ["terjangkau"],
"ramai": ["padat"],  "tenang": ["damai"], "kotor":
["jorok"],
}

# FUNGSI EDA
def synonym_replacement(words, p=0.3):
    return [
        random.choice(SYN_DICT[w]) if w in SYN_DICT
and random.random() < p else w
        for w in words
    ]

def random_deletion(words, p=0.1):
    if len(words) <= 3:
        return words
    new_words = [w for w in words if random.random() >
p]
    return new_words if new_words else words

def eda_transform(text):
    words = text.split()
    if not words:
        return text
    op = random.choice([synonym_replacement,
random_deletion])
    return " ".join(op(words.copy()))
```

```python
# EDA (kelas minoritas)
augmented_rows = []

for label in ["N", "O"]:
    subset = df[df["label"] == label]
    n_aug = int(len(subset) * EDA_RATIO)

    for _ in range(n_aug):
        row = subset.sample(1).iloc[0]
        new_text = eda_transform(row["text"])

        if new_text != row["text"]:
            augmented_rows.append({
                "text": new_text,
                "label": label,
                "data_type": "eda"
            })
df_eda = pd.concat([df, pd.DataFrame(augmented_rows)],
ignore_index=True)
df_eda = df_eda.drop_duplicates()

# DOWNSAMPLING kelas positif
counts = Counter(df_eda["label"])
max_minor = max(counts["N"], counts["O"])
max_p = int(max_minor * MAX_RATIO_P)
df_p = df_eda[df_eda["label"] == "P"].sample(
    n=min(len(df_eda[df_eda["label"] == "P"]), max_p),
    random_state=SEED
)
df_n = df_eda[df_eda["label"] == "N"]
df_o = df_eda[df_eda["label"] == "O"]

df_final = pd.concat([df_p, df_n, df_o])
df_final = df_final.sample(frac=1,
random_state=SEED).reset_index(drop=True)

# SIMPAN

df_final = df_final.rename(columns={
    "text": TEXT_COL,
    "label": LABEL_COL
})

df_final.to_csv(OUTPUT_PATH, index=False)
```

Lampiran 12. Kode *Fine Tuning* IndoBERT

```
    import os
os.environ["TOKENIZERS_PARALLELISM"] = "false"
os.environ["WANDB_DISABLED"] = "true"

import pandas as pd
import numpy as np
import torch
from torch import nn
from datasets import Dataset
from sklearn.model_selection import StratifiedKFold
from sklearn.metrics import classification_report,
confusion_matrix
from sklearn.utils.class_weight import
compute_class_weight

from transformers import (
    AutoTokenizer,
    AutoModelForSequenceClassification,
    Trainer,
    TrainingArguments
)

import evaluate
import matplotlib.pyplot as plt
import seaborn as sns

# LOAD DATA
df =
pd.read_csv("/content/hasil_stemming_eda_downsampling.
csv")

label2id = {"N":0, "O":1, "P":2}
id2label = {0:"N", 1:"O", 2:"P"}

df["Label"] = df["Label"].map(label2id)
df.dropna(subset=['Label'], inplace=True)
df["Label"] = df["Label"].astype(int)

# TOKENIZER
tokenizer =
AutoTokenizer.from_pretrained("indobenchmark/indobert-
base-p1", use_fast=True)

def tokenize(batch):
    return tokenizer(
        batch["stemming_data"],
        truncation=True,
        padding="max_length",
```

```
            max_length=384
    )

# CUSTOM TRAINER
class WeightedTrainer(Trainer):
    def __init__(self, class_weights=None, *args,
**kwargs):
        super().__init__(*args, **kwargs)
        self.class_weights =
class_weights.to(self.model.device)

    def compute_loss(self, model, inputs,
return_outputs=False, num_items_in_batch=0):
        labels = inputs.get("labels")
        outputs = model(**inputs)
        logits = outputs.get("logits")

        loss =
nn.CrossEntropyLoss(weight=self.class_weights)(logits,
labels)
        return (loss, outputs) if return_outputs else
loss

# K-FOLD
kfold = StratifiedKFold(n_splits=10, shuffle=True,
random_state=42)

metric_accuracy = evaluate.load("accuracy")
metric_f1 = evaluate.load("f1")

fold_results = []

X = df["cleaning_data"]
y = df["Label"]

all_true_labels = []
all_predicted_labels = []

# TRAINING LOOP
for fold, (train_idx, test_idx) in
enumerate(kfold.split(X, y), 1):

    train_df =
df.iloc[train_idx].reset_index(drop=True)
    test_df  =
df.iloc[test_idx].reset_index(drop=True)

    train_ds = Dataset.from_pandas(train_df)
    test_ds  = Dataset.from_pandas(test_df)

    train_ds = train_ds.map(tokenize, batched=True)
```

```
    test_ds  = test_ds.map(tokenize, batched=True)

    train_ds = train_ds.rename_column("Label",
"labels")
    test_ds  = test_ds.rename_column("Label",
"labels")

    train_ds.set_format(type="torch", columns=
["input_ids","attention_mask","labels"])
    test_ds.set_format(type="torch", columns=
["input_ids","attention_mask","labels"])

    class_weights = compute_class_weight(
        class_weight='balanced',
        classes=np.unique(train_df["Label"]),
        y=train_df["Label"]
    )
    class_weights = torch.tensor(class_weights,
dtype=torch.float)

    model =
AutoModelForSequenceClassification.from_pretrained(
        "indobenchmark/indobert-base-p1",
        num_labels=3,
        id2label=id2label,
        label2id=label2id
    )

    args = TrainingArguments(
        output_dir=f"./tmp_fold_{fold}",
        learning_rate=3e-5,
        per_device_train_batch_size=8,
        per_device_eval_batch_size=16,
        num_train_epochs=3,
        weight_decay=0.01,
        warmup_ratio=0.1,
        save_strategy="no",
        eval_strategy="no",
        report_to="none",
        fp16=torch.cuda.is_available()
    )

    trainer = WeightedTrainer(
        model=model,
        args=args,
        train_dataset=train_ds,
        tokenizer=tokenizer,
        class_weights=class_weights
    )
```

```
    trainer.train()

    preds = trainer.predict(test_ds)
    y_true = test_df["Label"]
    y_pred = preds.predictions.argmax(axis=1)

    all_true_labels.extend(y_true.tolist())
    all_predicted_labels.extend(y_pred.tolist())

    acc =
metric_accuracy.compute(predictions=y_pred,
references=y_true)["accuracy"]
    f1 = metric_f1.compute(predictions=y_pred,
references=y_true, average="macro")["f1"]
    fold_results.append((acc, f1))
# HASIL AKHIR
acc_mean = np.mean([r[0] for r in fold_results])
f1_mean  = np.mean([r[1] for r in fold_results])

print(f"Mean Accuracy : {acc_mean:.4f}")
print(f"Mean F1 Macro : {f1_mean:.4f}")
```

Lampiran 13. Distribusi Data 28 DWT Kabupaten Karangasem

NO	NAMA DAYA TARIK WISATA	Data	P	N	O
1	DTW Taman Soekasada Ujung	1594	1376	140	78
2	DTW Pesona Bukit Asah Bali	217	189	23	5
3	DTW Pantai Pasir Putih/ Virgin Beach	659	585	44	30
4	DTW Pantai Jasri	123	107	13	3
5	DTW Candidasa	151	136	11	4
6	DTW Museum Lontar Dukuh Penaban	101	101	0	0
7	DTW Tenganan Pegringsingan	554	459	49	46
8	DTW Padangbai	529	131	319	79
9	DTW Tenganan Dauh Tukad	15	12	2	1
10	DTW Pantai Blue Lagoon	1228	912	244	72
11	DTW Pantai Wates Yeh Malet	38	22	9	7
12	DTW Jemeluk	173	147	13	13
13	DTW Pesona Bukit Lempuyang	4149	1445	1517	1187
14	DTW Taman Tirta Gangga	4222	3800	231	191
15	DTW Maha Gangga Valley	420	386	24	10
16	DTW Pesona Alam Kastala	22	20	1	1
17	DTW Agrowisata Salak Sibetan	87	86	0	1
18	DTW Bukit Surga	32	22	7	3
19	DTW Bukit Putung	22	15	4	3
20	DTW Pesona Alam Jaga Satru	283	266	10	7
21	DTW Lingkungan Pura Agung Besakih	4301	2763	958	580
22	DTW Taman Edelweis Bali	945	854	64	27
23	DTW Telaga Surya	444	374	42	28
24	DTW Pesona Alam Sangkan Gunung	15	9	3	3
25	DTW Air Terjun Gembleng	1170	1117	23	30
26	DTW Savana Tianyar	457	346	84	27
27	DTW Pantai Amed	503	433	41	29
28	DTW Pantai Bias Tugel	381	331	38	12