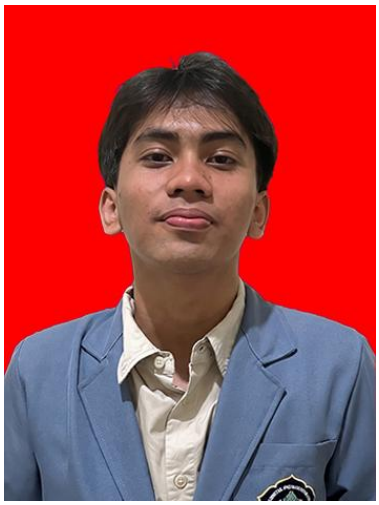
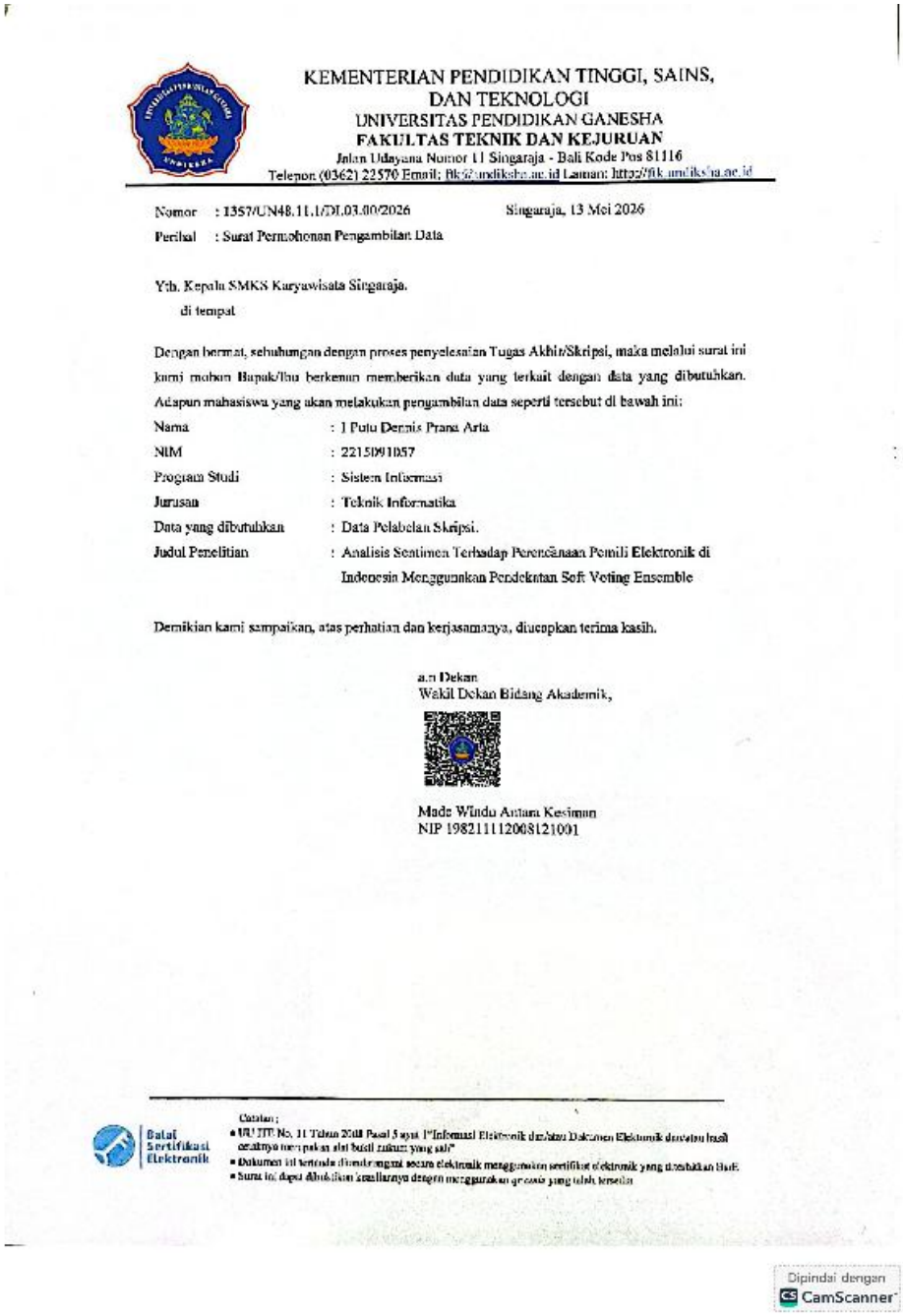
LAMPIRAN

Lampiran 01. Riwayat Hidup



I Putu Dennis Prana Arta lahir di Jembrana pada tanggal 27 Februari 2004. Penulis merupakan putra dari pasangan I Ketut Budiarta dan Ni Luh Narawati. Penulis berkewarganegaraan Indonesia dan beragama Hindu. Pendidikan dasar ditempuh di SD Negeri 5 Pendem dan diselesaikan pada tahun 2016. Selanjutnya, penulis melanjutkan pendidikan menengah pertama di SMP Negeri 1 Negara dan lulus pada tahun 2019. Pendidikan menengah atas ditempuh di SMA Negeri 1 Negara dan diselesaikan pada tahun 2022. Pada tahun yang sama, penulis melanjutkan pendidikan ke jenjang Sarjana (S1) pada Program Studi Sistem Informasi, Jurusan Teknik Informatika, Fakultas Teknik dan Kejuruan, Universitas Pendidikan Ganesha.

Lampiran 02. Surat Permohonan 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 : 1357/UN48.11.1/DL.03.00/2026 Singaraja, 13 Mei 2026
Perihal : Surat Permohonan Pengambilan Data

Yth. Kepala SMKS Karyawisata Singaraja.
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	: I Putu Dennis Prana Arta
NIM	: 2215091057
Program Studi	: Sistem Informasi
Jurusan	: Teknik Informatika
Data yang dibutuhkan	: Data Pelabelan Skripsi.
Judul Penelitian	: Analisis Sentimen Terhadap Perencanaan Pemilu Elektronik di Indonesia Menggunakan Pendekatan Soft Voting Ensemble

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 cetakannya 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

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 : 1355/UN48.11.1/DL.03.00/2026 Singaraja, 13 Mei 2026
Perihal : Surat Permohonan Pengambilan Data

Yth. Kepala SMK Pariwisata Triatmajaya,
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	: I Putu Dennis Prana Arta
NIM	: 2215091057
Program Studi	: Sistem Informasi
Jurusan	: Teknik Informatika
Data yang dibutuhkan	: Data Pelanelan Skripsi.
Judul Penelitian	: Analisis Sentimen Terhadap Perencanaan Pemilu Elektronik di Indonesia Menggunakan Pendekatan Soft Voting Ensemble

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

n.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 cetakannya merupakan alat bukti hukum yang sah"
- Dokumen ini ditandatangani secara elektronik menggunakan sertifikat elektronik yang diterbitkan BsrE
- Surat ini dapat dibuktikan keasliannya dengan menggunakan qr code yang telah tersedia

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 : 1356/UN48.11.1/DI.03.00/2026 Singaraja, 13 Mei 2026
Perihal : Surat Permohonan Pengambilan Data

Yth. Kepala SMP Negeri 3 Kubutambahan.
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	: I Putu Dennis Prana Arta
NIM	: 2215091057
Program Studi	: Sistem Informasi
Jurusan	: Teknik Informatika
Data yang dibutuhkan	: Data Pelabelan Skripsi.
Judul Penelitian	: Analisis Sentimen Terhadap Perencanaan Pemilu Elektronik di Indonesia Menggunakan Pendekatan Soft Voting Ensemble

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

n.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 cetakannya merupakan alat bukti hukum yang sah"
- Dokumen ini tertanda dibubuhkan secara elektronik menggunakan sertifikat elektronik yang diterbitkan BsrE
- Surat ini dapat dibuktikan keasliannya dengan menggunakan qr code yang telah tersedia

Lampiran 03. Surat Validasi Pelabelan Data

KEMENTERIAN PENDIDIKAN TINGGI, SAINS, DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
FAKULTAS TEKNIK DAN KEJURUAN
Jalan Udayana Singaraja-Bali Kode Pos 81116
Tlp. (0362) 22570 fax. (0362) 25735
Laman: www.undiksha.ac.id

SURAT KETERANGAN

LABELING DATA

Yang bertanda tangan dibawah ini:

Nama : Kadek Indah Kusuma Dewi, S.Pd, Gr.

Jabatan : Guru Bahasa Indonesia SMAS Karyawisata Singaraja

Menerangkan bahwa Mahasiswa Universitas Pendidikan Ganesha dibawah ini :

Nama : I Putu Dennis Prana Arta

NIM : 2215091057

Prodi : S1 Sistem Informasi

Dengan ini menerangkan bahwa proses pelabelan dataset telah dilaksakan dan diselesikan pada tanggal 14 Mei 2026.

Demikian surat keterangan ini dibuat dengan sebenarnya untuk dapat digunakan sebagaimana mestinya.

Singaraja, 14 Mei 2026

Ahli/Pakar Bahasa,

Kadek Indah Kusuma Dewi, S.Pd., Gr

KEMENTERIAN PENDIDIKAN TINGGI, SAINS, DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
FAKULTAS TEKNIK DAN KEJURUAN
Jalan Udayana Singaraja-Bali Kode Pos 81116
Tlp. (0362) 22570 fax. (0362) 25735
Laman: www.undiksha.ac.id

SURAT KETERANGAN

LABELING DATA

Yang bertanda tangan dibawah ini:

Nama : Ketut Suryaningsih, S.Pd., Gr.

Jabatan : Guru Bahasa Indonesia SMPN 3 Kubutambahan

Menerangkan bahwa Mahasiswa Universitas Pendidikan Ganesha dibawah ini :

Nama : I Putu Dennis Prana Arta

NIM : 2215091057

Prodi : S1 SIstem Informasi

Dengan ini menerangkan bahwa proses pelabelan dataset telah dilaksakan dan diselesikan pada tanggal 14 Mei 2026.

Demikian surat keterangan ini dibuat dengan sebenarnya untuk dapat digunakan sebagaimana mestinya.

Singaraja, 14 Mei 2026

Ahli/Pakar Bahasa,

Ketut Suryaningsih, S.Pd., Gr

KEMENTERIAN PENDIDIKAN TINGGI, SAINS, DAN TEKNOLOGI
UNIVERSITAS PENDIDIKAN GANESHA
FAKULTAS TEKNIK DAN KEJURUAN
Jalan Udayana Singaraja-Bali Kode Pos 81116
Tlp. (0362) 22570 fax. (0362) 25735
Laman: www.undiksha.ac.id

SURAT KETERANGAN

LABELING DATA

Yang bertanda tangan dibawah ini:

Nama : Luh Risa Santiani, S.Pd., Gr

Jabatan : Guru Bahasa Indonesia SMKS Pariwisata Triatmajaya

Menerangkan bahwa Mahasiswa Universitas Pendidikan Ganesha dibawah ini :

Nama : I Putu Dennis Prana Arta

NIM : 2215091057

Prodi : S1 Sistem Informasi

Dengan ini menerangkan bahwa proses pelabelan dataset telah dilaksakan dan diselesikan pada tanggal 14 Mei 2026.

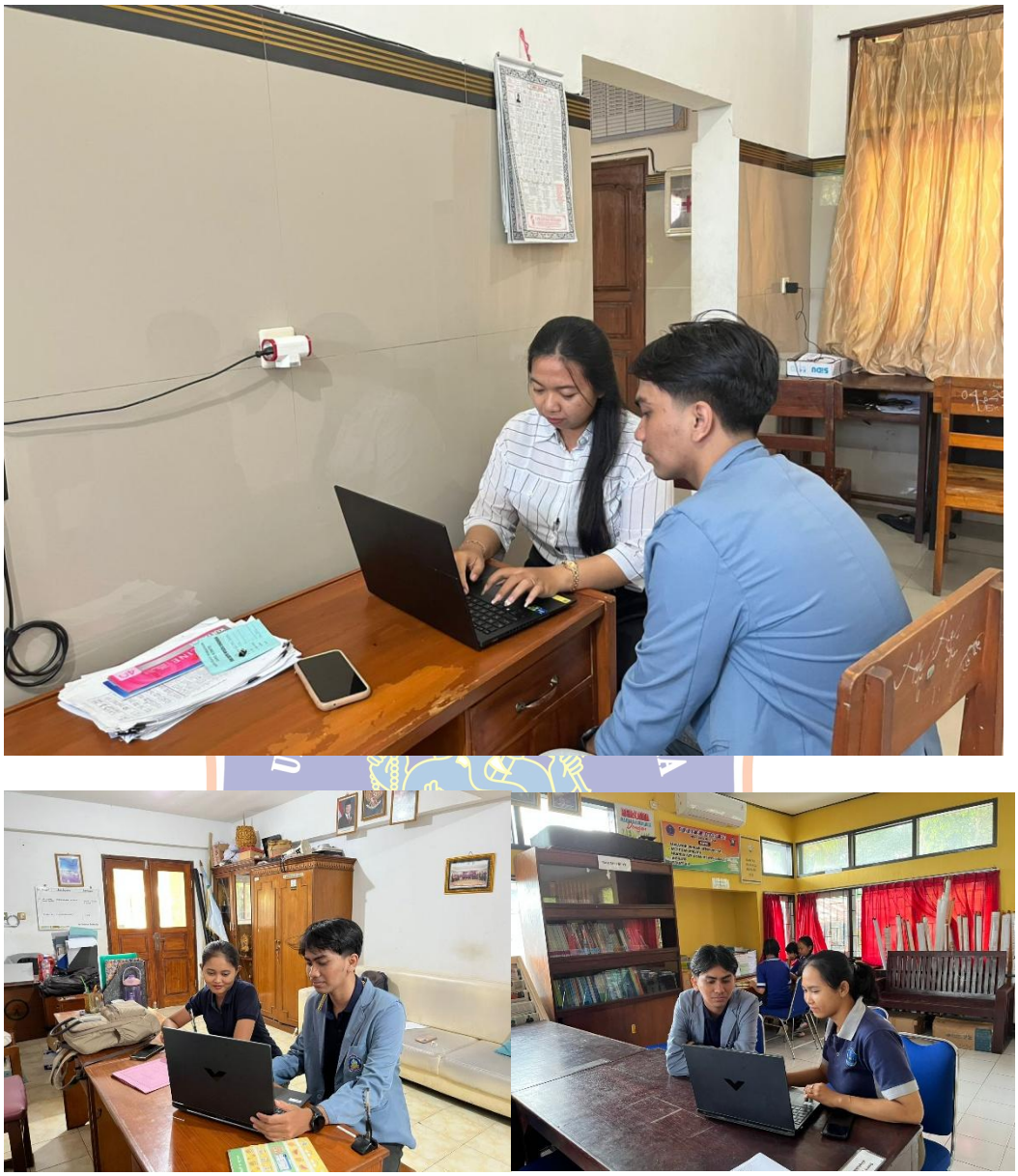
Demikian surat keterangan ini dibuat dengan sebenarnya untuk dapat digunakan sebagaimana mestinya.

Singaraja, 14 Mei 2026

Ahli/Pakar Bahasa,

Luh Risa Santiani, S.Pd., Gr

Lampiran 04. Dokumentasi Pelabelan Data



Lampiran 05. Code Crawling dan Scraping Data

a. Code Crawling X (Twitter)

```
# Import required Python package
!pip install pandas

!sudo apt-get update
!sudo apt-get install -y ca-certificates curl gnupg
!sudo mkdir -p /etc/apt/keyrings
```

```
!curl                                                -fsSL
https://deb.nodesource.com/gpgkey/nodesource-
repo.gpg.key     |     sudo     gpg     --dearmor     -o
/etc/apt/keyrings/nodesource.gpg

!NODE_MAJOR=20      &&      echo      "deb      [signed-
by=/etc/apt/keyrings/nodesource.gpg]
https://deb.nodesource.com/node_$NODE_MAJOR.x
nodistro         main"         |         sudo         tee
/etc/apt/sources.list.d/nodesource.list

!sudo apt-get update
!sudo apt-get install nodejs -y

!node -v

# Crawl Data

filename = 'CrawlData-PemiluElektronik.csv'
search_keyword = 'Pemilu Elektronik since:2019-01-01
until:2025-06-01 lang:id'
limit = 10000

!npx   -y   tweet-harvest@2.6.1   -o   "{filename}"   -s
"{search_keyword}" --tab "LATEST" -l {limit} --token
{twitter_auth_token}
```

b. Code Scraping Data TikTok

```
# Install library Apify Client
!pip install apify-client pandas

from apify_client import ApifyClient

# Initialize the ApifyClient with your API token
client = ApifyClient("API_TOKEN")

# Prepare the Actor input
run_input = {
    "commentsPerPost": 4000,
    "excludePinnedPosts": False,
    "maxRepliesPerComment": 5,
    "postURLs": [

"https://www.tiktok.com/@itdnews/video/71334654948661
85498?is_from_webapp=1&sender_device=pc&web_id=758164
5062795871765"
```

```
    ],
    "resultsPerPage": 100
}

# Run the Actor and wait for it to finish
run                                                    =
client.actor("BDecooyAmCm1QbMEI").call(run_input=run_
input)

items                                                  =
list(client.dataset(run["defaultDatasetId"]).iterate_
items())

import pandas as pd

df = pd.DataFrame(items)

nama_file = "hasil_scraping_tiktok.csv"
df.to_csv(nama_file,  index=False,  encoding="utf-8-
sig")

print(f"Data berhasil disimpan ke file: {nama_file}")
print(f"Jumlah data: {len(df)}")
```

Lampiran 06. Code Preprocessing Data

a. Code Cleaning Data

```
def cleaning_text(text):
    if not isinstance(text, str):
        return ""

    text  =  re.sub(r"http\S+|www\S+|https\S+",  "",
text)
    text = re.sub(r"@\w+", "", text)
    text = re.sub(r"#\w+", "", text)
    text = re.sub(r"\d+", "", text)
    text = re.sub(r"[^\w\s]", " ", text)
    text = re.sub(r"_[A-Za-z0-9]+", " ", text)
    text = re.sub(r"\s+", " ", text).strip()
    return text

df["cleaning"] = df["full_text"].apply(cleaning_text)

df[["full_text", "cleaning"]].head()
```

b. Code Case Folding Data

```
def case_folding(text):
    if isinstance(text, str):
        return text.lower()
    return ""

df["case_folding"]=
df["cleaning"].apply(case_folding)

df[["cleaning", "case_folding"]].head()
```

c. Code Normalization Data

```
import requests
from io import BytesIO

url =
"https://github.com/analysisdatasentiment/kamus_kata_
baku/raw/main/kamuskatabaku.xlsx"
response = requests.get(url)
file_excel = BytesIO(response.content)

kamus_df = pd.read_excel(file_excel)
kamus_df.columns = [col.strip() for col in
kamus_df.columns]

kolom_tidak_baku = kamus_df.columns[0]
kolom_baku = kamus_df.columns[1]

kamus_tidak_baku = dict(
    zip(

kamus_df[kolom_tidak_baku].astype(str).str.strip().st
r.lower(),

kamus_df[kolom_baku].astype(str).str.strip().str.lowe
r()
    )
)

print("Jumlah entri kamus:", len(kamus_tidak_baku))
kamus_df.head()

def normalize_text(text, kamus):
    if not isinstance(text, str):
        return "", [], [], []

    words = text.split()
    normalized_words = []
```

```
    kata_baku = []
    kata_tidak_baku = []
    kata_tidak_baku_hash = []

    for word in words:
        clean_word = word.strip().lower()
        if clean_word in kamus:
            normalized_word = kamus[clean_word]
            normalized_words.append(normalized_word)

            if clean_word != normalized_word:
                kata_baku.append(normalized_word)
                kata_tidak_baku.append(clean_word)

kata_tidak_baku_hash.append(hash(clean_word))
        else:
            normalized_words.append(clean_word)

    return (
        " ".join(normalized_words),
        kata_baku,
        kata_tidak_baku,
        kata_tidak_baku_hash
    )

df[["normalisasi",   "kata_baku",   "kata_tidak_baku",
"kata_tidak_baku_hash"]] = (
    df["case_folding"].apply(lambda                    x:
pd.Series(normalize_text(x, kamus_tidak_baku)))
)

df[["case_folding", "normalisasi"]].head()
```

d. Code Tokenize Data

```
def tokenize(text):
    if isinstance(text, str):
        return text.split()
    return []

df["tokenize"] = df["normalisasi"].apply(tokenize)

df[["normalisasi", "tokenize"]].head()
```

e. Code Stopword Removal

```
import nltk
from nltk.corpus import stopwords
```

```
nltk.download("stopwords")

stop_words = set(stopwords.words("indonesian"))

# Stopword tambahan
custom_stopwords = [

"ya","tok","deh","wok","we","the","wi","sih","roy","n
ot","oh","entar","mah",

"kayak","auto","nih","cs","y","abang","bang","gue","g
we","gua","guaa","wkwk",

"wkwkwk","wk","hehe","hehee","yg","aja","nya","dong",
"nih","nihh","lah",
    "jokowi","puan"
]

all_stopwords                                          =
stop_words.union(set(custom_stopwords))

def remove_stopwords(tokens):
    return [word for word in tokens if word not in
all_stopwords]


df["stopword_removal"]                                 =
df["tokenize"].apply(remove_stopwords)
df["stopword_removal"]                                 =
df["stopword_removal"].apply(lambda x: " ".join(x))
df[["tokenize", "stopword_removal"]].head()
```

f. Code Stemming Data

```
!pip install Sastrawi

from      Sastrawi.Stemmer.StemmerFactory      import
StemmerFactory

factory = StemmerFactory()
stemmer = factory.create_stemmer()

protected_words = ["pemilu", "elektronik", "evoting",
"e-voting"]

def stemming_text(text):
    if not isinstance(text, str):
```

```
        return ""

    words = text.split()
    stemmed_words = []

    for word in words:
        clean_word = word.strip().lower()

        if clean_word in protected_words:
            stemmed_words.append(clean_word)
        else:

stemmed_words.append(stemmer.stem(clean_word))

    return " ".join(stemmed_words)

df["stemming"]                                                    =
df["stopword_removal"].apply(stemming_text)

df[["stopword_removal", "stemming"]].head()
```

Lampiran 07. Code Modeling

a. Code Import Library Modeling

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.base import clone
from sklearn.model_selection import train_test_split,
StratifiedKFold, cross_val_score
from      sklearn.feature_extraction.text      import
TfidfVectorizer, CountVectorizer
from sklearn.linear_model import LogisticRegression
from  sklearn.ensemble  import  RandomForestClassifier,
VotingClassifier
from     sklearn.metrics     import     accuracy_score,
classification_report, confusion_matrix
from           sklearn.decomposition           import
LatentDirichletAllocation
from imblearn.over_sampling import BorderlineSMOTE
```

b. Code Load Dataset dan TF-IDF

```
df = pd.read_csv("Hasil_Preprocessing_Data_Rapi_Revisi.csv")

df["stemming"] = df["stemming"].fillna("").astype(str)

if df["Sentiment"].dtype == object:
    df["Sentiment"] = df["Sentiment"].map({
        "Negatif": 0,
        "Positif": 1
    })

X = df["stemming"]
y = df["Sentiment"]

print("Jumlah data:", len(df))
print("\nDistribusi label:")
print(y.value_counts())

label_counts = y.map({
    0: "Negatif",
    1: "Positif"
}).value_counts()

plt.figure(figsize=(6, 4))
sns.barplot(x=label_counts.index, y=label_counts.values)
plt.title("Distribusi Label Sentimen")
plt.xlabel("Label")
plt.ylabel("Jumlah Data")

for i, v in enumerate(label_counts.values):
    plt.text(i, v + 5, str(v), ha="center")

plt.show()

# TRAIN TEST SPLIT

X_train_text, X_test_text, y_train, y_test = train_test_split(
    X,
    y,
    test_size=0.2,
    stratify=y,
    random_state=42
)

print("\nJumlah data train:", len(X_train_text))
print("Jumlah data test:", len(X_test_text))
```

```
# TF-IDF

vectorizer = TfidfVectorizer(
    max_features=5000,
    ngram_range=(1, 2),
    min_df=2,
    max_df=0.95,
    sublinear_tf=True,
    norm="l2"
)

X_train_tfidf = vectorizer.fit_transform(X_train_text)
X_test_tfidf = vectorizer.transform(X_test_text)

print("\nShape X_train_tfidf:", X_train_tfidf.shape)
print("Shape X_test_tfidf:", X_test_tfidf.shape)
```

c. Code Fungsi Evaluasi Model tanpa SMOTE

```
def evaluate_model_no_smote(model_name, model,
X_train_data, y_train_data, X_test_data, y_test_data):
    y_train_array = np.array(y_train_data)
    y_test_array = np.array(y_test_data)

    # Evaluasi pada data test
    model_fit = clone(model)
    model_fit.fit(X_train_data, y_train_array)
    y_pred = model_fit.predict(X_test_data)

    acc = accuracy_score(y_test_array, y_pred)
    cm = confusion_matrix(y_test_array, y_pred)

    print(f"\n=== {model_name} ===")
    print("Test Accuracy:", round(acc, 4))
    print("\nClassification Report:")
    print(classification_report(y_test_array, y_pred,
target_names=["Negatif", "Positif"], digits=4,
zero_division=0))
    print("Confusion Matrix:")
    print(cm)

    plt.figure(figsize=(5, 4))
    sns.heatmap(
        cm,
        annot=True,
        fmt="d",
```

```
        cmap="Blues",
        xticklabels=["Negatif", "Positif"],
        yticklabels=["Negatif", "Positif"]
    )
    plt.title(f"Confusion Matrix - {model_name}")
    plt.xlabel("Predicted Label")
    plt.ylabel("Actual Label")
    plt.tight_layout()
    plt.show()

    # 10-Fold Cross Validation
    cv = StratifiedKFold(n_splits=10, shuffle=True,
random_state=42)

    train_scores = []
    val_scores = []

    for train_idx, valid_idx in cv.split(X_train_data,
y_train_array):
        X_fold_train = X_train_data[train_idx]
        X_fold_valid = X_train_data[valid_idx]
        y_fold_train = y_train_array[train_idx]
        y_fold_valid = y_train_array[valid_idx]

        model_fold = clone(model)
        model_fold.fit(X_fold_train, y_fold_train)

        # Training accuracy pada data training fold
asli
        y_fold_train_pred                          =
model_fold.predict(X_fold_train)
        train_acc    =    accuracy_score(y_fold_train,
y_fold_train_pred)
        train_scores.append(train_acc)

        # Validation accuracy pada data validation fold
        y_fold_valid_pred                          =
model_fold.predict(X_fold_valid)
        val_acc     =     accuracy_score(y_fold_valid,
y_fold_valid_pred)
        val_scores.append(val_acc)

    train_scores = np.array(train_scores)
    val_scores = np.array(val_scores)
    fold_numbers = np.arange(1, len(val_scores) + 1)
    cv_gap = train_scores.mean() - val_scores.mean()

    print("\n=== HASIL K-FOLD CROSS VALIDATION (10-
FOLD) ===")
```

```
    print("Training Accuracy per Fold :",
np.round(train_scores, 4))
    print("Validation Accuracy per Fold:",
np.round(val_scores, 4))
    print("Mean Training Accuracy :",
round(train_scores.mean(), 4))
    print("Mean Validation Accuracy :",
round(val_scores.mean(), 4))
    print("Train-Validation Gap :", round(cv_gap,
4))
    print("Std Validation Accuracy :",
round(val_scores.std(), 4))

    plt.figure(figsize=(8, 5))
    plt.plot(fold_numbers, train_scores, marker="o",
linewidth=2, label="Training Accuracy")
    plt.plot(fold_numbers, val_scores, marker="o",
linewidth=2, label="Validation Accuracy")

    for i, score in enumerate(train_scores):
        plt.text(fold_numbers[i], score,
f"{score:.3f}", ha="center", va="bottom", fontsize=9)

    for i, score in enumerate(val_scores):
        plt.text(fold_numbers[i], score,
f"{score:.3f}", ha="center", va="top", fontsize=9,
color="red")

    plt.title(f"K-Fold Cross Validation
Performance\nGap Train-Val = {cv_gap:.4f}")
    plt.xlabel("Fold ke-")
    plt.ylabel("Accuracy")
    plt.xticks(fold_numbers)
    plt.legend()
    plt.grid(True, linestyle="--", alpha=0.4)
    plt.tight_layout()
    plt.show()

    return {
        "Model": model_name,
        "Test Accuracy": round(acc, 4),
        "Training Scores": np.round(train_scores,
4).tolist(),
        "Validation Scores": np.round(val_scores,
4).tolist(),
        "CV Mean Training Accuracy":
round(train_scores.mean(), 4),
        "CV Mean Validation Accuracy":
round(val_scores.mean(), 4),
```

```
        "CV Gap (Train-Val)": round(cv_gap, 4),
        "CV      Std      Validation      Accuracy":
round(val_scores.std(), 4)
    }
```

d. Code Fungsi Evaluasi Model dengan SMOTE

```
def    evaluate_model_with_smote(model_name,    model,
X_train_data, y_train_data, X_test_data, y_test_data):
    y_train_array = np.array(y_train_data)
    y_test_array = np.array(y_test_data)

    # Evaluasi pada data test
    # SMOTE hanya diterapkan pada data training, bukan
test.
    smote = BorderlineSMOTE(random_state=42)

    X_train_dense = X_train_data.toarray()
    X_test_dense = X_test_data.toarray()

    X_train_res,                y_train_res                =
smote.fit_resample(X_train_dense, y_train_array)

    model_fit = clone(model)
    model_fit.fit(X_train_res, y_train_res)
    y_pred = model_fit.predict(X_test_dense)

    acc = accuracy_score(y_test_array, y_pred)
    cm = confusion_matrix(y_test_array, y_pred)

    print(f"\n=== {model_name} ===")
    print("Test Accuracy:", round(acc, 4))
    print("\nClassification Report:")
    print(classification_report(y_test_array,  y_pred,
target_names=["Negatif",     "Positif"],     digits=4,
zero_division=0))
    print("Confusion Matrix:")
    print(cm)

    plt.figure(figsize=(5, 4))
    sns.heatmap(
        cm,
        annot=True,
        fmt="d",
        cmap="Blues",
        xticklabels=["Negatif", "Positif"],
        yticklabels=["Negatif", "Positif"]
    )
```