LAMPIRAN

Lampiran 1. Surat balasan permohonan izin PKL

PT Angkasa Pura I
Juanda - Surabaya :
Bandar Udara Internasional Juanda Jalan Ir. Haji Juanda Surabaya 61253
tel : (+6231) 2986200 fax : (+6231) 2986200
web : www.ap1.co.id

Nomor : AP.I.1787/DL.13/2024/SUB.AD-B
Lampiran : 1
Perihal : KONFIRMASI PERMOHONAN PKL - SAINS DATA UPN VETERAN JAWA TIMUR (17 JUNI - 18 AGUSTUS 2024)

Kepada Yth,
KOORDINATOR PROGRAM STUDI SAINS DATA
UNIVERSITAS PEMBANGUNAN NASIONAL "VETERAN" JAWA TIMUR
di -
Tempat

Menunjuk Surat Koordinator Program Studi Sains Data UPN "Veteran" Jawa Timur Nomor : 23/UN63.7/SD/III/2024 tanggal 5 Maret 2024 perihal Permohonan Izin Praktik Kerja Lapangan (PKL), bersama ini dapat disampaikan bahwa pada prinsipnya manajemen dapat menyetujui permohonan pelaksanaan Praktek Kerja Lapangan 1 (satu) orang mahasiswa program studi Sains Data UPN Veteran Jawa Timur atas nama Muhimmatul Arofah di Airport Technology Section TMT 17 Juni - 18 Agustus 2024.

Adapun ketentuan selama pelaksanaan Praktek Kerja Lapangan sebagai berikut :

1. Melaksanakan Kerja Praktek sesuai dengan jam kerja yang berlaku di perusahaan;
2. Menaati Tata Tertib Kerja Praktek;
3. Menggunakan kemeja, jas almamater dan Tanda Pengenal yang dikeluarkan oleh Perusahaan;
4. Wajib mengembalikan Tanda Pengenal setelah Kerja Praktek selesai;
5. Menjaga semua kerahasiaan Perusahaan;
6. Mengisi daftar hadir di unit kerja terkait pada waktu datang dan pulang;
7. Wajib melaksanakan Protokol Kesehatan;
8. Tidak melaksanakan kegiatan yang melanggar ketentuan yang berlaku (misalnya : foto didaerah terlarang, penggunaan obat/minuman terlarang, dll);
9. Menyampaikan 1 (satu) set laporan Praktek Kerja Lapangan kepada Human Capital Business Partner Section;
10. Menyampaikan permintaan melalui laman PPID (https://ppid.ap1.co.id/prosedur-permohonan-informasi) jika ada permohonan data / informasi yang dibutuhkan, baik untuk keperluan penulisan laporan maupun untuk keperluan akademis lainnya

Terkait hal tersebut di atas, jika membutuhkan konfirmasi lebih lanjut dapat menghubungi Human Capital Business Partner Section. Demikian disampaikan, atas perhatiannya diucapkan terima kasih.

Surabaya, 07 Mei 2024
a.n. GENERAL MANAGER
PGS. AIRPORT ADMINISTRATION SENIOR MANAGER,

Ditandatangani secara elektronik
MAHENDRA

Tembusan Yth. :

1. GENERAL MANAGER BANDARA JUANDA (SUB)
2. CO. GENERAL MANAGER BANDARA JUANDA SURABAYA (SUB)
3. AIRPORT TECHNOLOGY MANAGER (SUB)

SUB/SUB.AD/1491079-G/G.RIZQITA

Lampiran 2. Code script tahap pengumpulan data

```
# proses pengumpulan data yang diambil dari ulasan google maps
import time
import csv
import re
from bs4 import BeautifulSoup
from selenium import webdriver
from webdriver manager.chrome import ChromeDriverManager as CM

def main(url maps):
    options = webdriver.ChromeOptions()
    # options.add argument("--headless")
    options.add argument('--no-sandbox')
    options.add_argument("--log-level=3")
    mobile_emulation = {
        "userAgent": "Mozilla/5.0 (Linux; Android 4.2.1; en-us; Nexus 5
Build/JOP40D) AppleWebKit/535.19 (KHTML, like Gecko) Chrome/90.0.1025.166 Mobile
Safari/535.19"}
    options.add_experimental_option("mobileEmulation", mobile_emulation)

    driver = webdriver.Chrome(executable_path=CM().install(), options=options)
    driver.set_window_size(600, 700)

    alamatURL = url_maps
    driver.get(alamatURL)
    time.sleep(3)
    try:
        count = 0
        print(f"load all data..")
        while True:
            print(f"run infinite scrolling...{count}")
            count += 1
            last_review =
driver.find_elements_by_css_selector('div[jsinstance="0"]')
            driver.execute_script('arguments[0].scrollIntoView(true);',
last review[count])
            time.sleep(0.8)
    except IndexError:
        print('finish load data')

    print('end scroll')
    page source = driver.page source
    soup = BeautifulSoup(page_source, 'lxml')
    section = soup.find all('div', {'jsinstance' : re.compile(r".*")})
    result = []
```

```
    re_nama_waktu = r'id="ml-reviews-page-user-review-
name.\w*".jsan=".\w*.\w*,\d.\w*,\d.\w*".jstcache="\d*">\w* \w*<.div>
<\w*.\w*="\w*" \w*="\d.\w*" jstcache="\d*">\d \w* \w*'
    re_rating = r'div aria-label="\w* \w*.\d'
    re_ulasan = f'<span class="\w*" jsan="\w.\w*" jstcache="\d*">[^<]*'
    for item in section:
        matches = re.search(re_nama_waktu, str(item), re.MULTILINE)
        matches_rating = re.search(re_rating, str(item), re.MULTILINE)
        matches_ulasan = re.search(re_ulasan, str(item), re.MULTILINE)
        if matches != None:
            raw = matches.group()
            raw_rating = matches_rating.group()
            raw_ulasan = matches_ulasan.group()

            nama = raw.split('>')[1].replace('</div', '')
            waktu = raw.split('>')[-1]
            rating = raw_rating.replace('div aria-label="Nilai dari ', '')
            ulasan = re.split(r'\d*">', raw ulasan)[1]
            temp = {
                'nama': nama,
                'waktu': waktu,
                'rating': rating,
                'ulasan': ulasan
            }
            result.append(temp)
    # save to csv
    keys = result[0].keys()
    with open(f'output.csv', 'w', newline='') as output file:
        dict_writer = csv.DictWriter(output_file, keys)
        dict writer.writeheader()
        dict writer.writerows(result)
    print('finish')
    time.sleep(2)


if __name__ == '__main__':
    url_map =
"https://www.google.com/maps/place/Bandar+Udara+Internasional+Juanda/@-
7.3788663,112.7847141,17z/data=!3m1!4b1!4m6!3m5!1s0x2dd7e50b3bf959b9:0xc0ff7c587
86318e8!8m2!3d-
7.3788716!4d112.787289!16zL20vMGdocnAy?entry=ttu&g_ep=EgoyMDI0MTEyNC4xIKXMDSoASA
FQAw%3D%3D"
    main(url_map)
```

```
# proses penggabungan dan pembersihan data
from google.colab import drive
drive.mount('/content/drive')
import os
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
# Path ke direktori tempat file-file Excel berada
search_path = '/content/drive/My Drive/file pkl/dataset'

# List untuk menampung DataFrame
df_list = []

# Loop melalui direktori dan subdirektori untuk mencari file Excel
for root, dirs, files in os.walk(search_path):
    for file in files:
        if file.endswith(".csv"):
            file path = os.path.join(root, file)
            df = pd.read csv(file path)
            df list.append(df)
combined_df = pd.concat(df_list, ignore_index=True)
olumn to delete = ['url', 'profile', 'fontLabelMedium', 'photo', 'dSlJg', 'dSlJg
2', 'znYl0 2']

# Menghapus kolom yang ditentukan
df = combined df.drop(column to delete, axis=1)
df = df.dropna(how='all')

# Definisikan urutan waktu
time order = [
    '3 hari lalu', '4 hari lalu', '5 hari lalu', '6 hari lalu', 'seminggu lalu',
    '2 minggu lalu', '3 minggu lalu', '4 minggu lalu', 'sebulan lalu', '2 bulan
lalu',
    '3 bulan lalu', '4 bulan lalu', '5 bulan lalu', '6 bulan lalu', '7 bulan
lalu',
    '8 bulan lalu', '9 bulan lalu', '10 bulan lalu', '11 bulan lalu', 'setahun
lalu',
    '2 tahun lalu', '3 tahun lalu', '4 tahun lalu', '5 tahun lalu', '6 tahun
lalu',
    '7 tahun lalu', '8 tahun lalu', '9 tahun lalu', '10 tahun lalu', '11 tahun
lalu',
    '12 tahun lalu', '13 tahun lalu'
]

# Buat dictionary untuk mengonversi waktu menjadi nilai numerik
time_dict = {time: index for index, time in enumerate(time_order)}
```

```
# Tambahkan kolom baru berdasarkan urutan waktu
df['time_order'] = df['time'].map(time_dict)

desired_time_range = [
    '3 hari lalu', '4 hari lalu', '5 hari lalu', '6 hari lalu', 'seminggu lalu',
    '2 minggu lalu', '3 minggu lalu', '4 minggu lalu', 'sebulan lalu', '2 bulan
lalu',
    '3 bulan lalu',  '4 bulan lalu', '5 bulan lalu', '6 bulan lalu', '7 bulan
lalu',
    '8 bulan lalu', '9 bulan lalu', '10 bulan lalu', '11 bulan lalu', 'setahun
lalu',
    '2 tahun lalu', '3 tahun lalu', '4 tahun lalu', '5 tahun lalu'
]
df filtered = df[df['time'].isin(desired time range)]

# Urutkan DataFrame berdasarkan kolom numerik
df_sorted = df_filtered.sort_values('time_order')
df1 = df sorted.dropna(ignore index=True)
df_cleaned = df1.dropna()
```

Lampiran 3. Code script tahap pra-proses data

```
pip install tweet-preprocessor
pip install Sastrawi
pip install deep-translator
import pandas as pd
import re
import Sastrawi
from Sastrawi.StopWordRemover.StopWordRemoverFactory import
StopWordRemoverFactory, StopWordRemover,ArrayDictionary
from Sastrawi.Stemmer.StemmerFactory import StemmerFactory
from deep_translator import GoogleTranslator

def clean text(text):
    if isinstance(text, str):
        text = re.sub(r'\n', ' ', text)
        text = re.sub(r'RT[\s]+', ' ', text)
        text = re.sub(r'https?://\S+', ' ', text)
        text = re.sub(r'[^A-Za-z0-9 ]', '', text)
        text = re.sub(r'\s+', ' ', text).strip()
        text = re.sub(r'[=+{}[\]<>@#$%^&*()_~`|\\/:;\"\'?!.,-]', ' ', text)
    return text

# Verify the correct column name and apply the function
# Replace 'actual_column_name' with the correct name of the column containing
the tweet text
df['review'] = df['review'].apply(clean_text)
df['review'] = df['review'].str.lower()
norm = {}
# Populate the dictionary with words from the CSV
for _, row in nrml.iterrows():
    norm[f" {row['Word']} "] = f" {row['Correction']} "
def normalisasi(str_text):
  for i in norm:
    str_text = str_text.replace(i, norm[i])
  return str_text
df['review'] = df['review'].apply(lambda x: normalisasi(x))
stop_words = StopWordRemoverFactory().get_stop_words()
#stop_words.extend(more_stop_words)

new array = ArrayDictionary(stop words)
stop_words_remover_new = StopWordRemover(new_array)

def stopword(str_text):
  str text = stop words remover new.remove(str text)
  return str text
```

```
df['review'] = df['review'].apply(lambda x: stopword(x))
tokenized = df['review'].apply(lambda x:x.split())
def stemming(text_cleaning):
  factory = StemmerFactory()
  stemmer = factory.create_stemmer()
  do = []
  for w in text_cleaning:
    dt = stemmer.stem(w)
    do.append(dt)
  d clean = []
  d_clean = " ".join(do)
  print(d_clean)
  return d_clean

tokenized = tokenized.apply(stemming)
tokenized.to csv('/content/drive/My Drive/file pkl/Preprocessing.csv',
index=False)
translator = GoogleTranslator(source='id', target='en')

# Fungsi untuk membagi teks menjadi beberapa bagian
def split_text(text, length):
    return [text[i:i+length] for i in range(0, len(text), length)]

# Fungsi untuk menerjemahkan teks
def translate_text(text):
    try:
        if pd.isna(text) or not isinstance(text, str) or len(text) == 0:
            return ""
        elif len(text) > 5000:
            parts = split text(text, 5000)
            translated parts = [translator.translate(part) for part in parts]
            return ' '.join(translated parts)
        else:
            return translator.translate(text)
    except Exception as e:
        print(f"Error translating text: {e}")
        return text

# Terapkan fungsi terjemahan pada kolom teks
data['review_english'] = data['review'].apply(translate_text)

# Simpan kembali DataFrame yang telah diterjemahkan ke file CSV baru
data.to_csv('/content/drive/My Drive/file pkl/Terjemahan.csv', index=False)
```

Lampiran 4. Code script tahap pelabelan data

```
!pip install textblob
!pip install nltk
pip install wordcloud
import nltk
from textblob import TextBlob
from textblob.classifiers import NaiveBayesClassifier
from nltk.stem import PorterStemmer
nltk.download('punkt')

data = pd.read_csv("Terjemahan_clean1.csv")
data['review english'] = data['review english'].str.lower()
data gmaps = list(data['review english'])
polaritas = 0

status = []
total_positif = total_negatif = total_netral = total = 0

for i,tweet in enumerate(data_gmaps):
  analysis = TextBlob(tweet)
  polaritas += analysis.polarity

  if analysis.sentiment.polarity > 0:
    total_positif += 1
    status.append('Positif')
  elif analysis.sentiment.polarity == 0:
    total_netral += 1
    status.append('Netral')
  else:
    total_negatif += 1
    status.append('Negatif')
  total += 1
data['gmaps_klasifikasi'] = status
data.to_csv('/content/drive/My Drive/file pkl/data_klasifikasi.csv',
index=False)

import seaborn as sns
import matplotlib.pyplot as plt
from wordcloud import WordCloud, STOPWORDS

def plot_cloud(wordcloud):
  plt.figure(figsize=(10,8))
  plt.imshow(wordcloud, interpolation='bilinear')
  plt.axis('off')
  plt.show()
```

```
all_words = ' '.join([tweets for tweets in data['review']])

wordcloud = WordCloud(
    width=3000,
    height=2000,
    random_state=3,
    background_color='black',
    colormap='Blues_r',
    collocations=False,
    stopwords=STOPWORDS
).generate(all_words)

plot_cloud(wordcloud)

sns.set_theme()

labels = ['Positif', 'Netral', 'Negatif']
counts = [total positif, total netral, total negatif]

def show bar chart(labels, counts, title):
  fig, ax = plt.subplots(figsize=(8, 6))
  bars = ax.bar(labels, counts, color=['#2394f7', '#fac343', '#f72323'])

  for bar, count in zip(bars, counts):
    height = bar.get_height()
    ax.annotate(f'{count}', xy=(bar.get x() + bar.get width() / 2, height),
                xytext=(0, 3),
                textcoords="offset points",
                ha='center', va='bottom')
  ax.grid(axis='y', linestyle='--', alpha=0.7)

  ax.set xlabel('Sentimen')
  ax.set_ylabel('Jumlah')
  ax.set title(title)

  plt.show()

show bar chart(labels, counts, "Distribusi Sentimen dari Review Google Maps
Menggunakan TextBlob")
```

Lampiran 5. Code script tahap klasifikasi dan pengujian dengan Naive Bayes

```
import random

dataset = data.drop(['review'], axis=1, inplace=False)
dataset = [tuple(x) for x in dataset.to records(index=False)]

set_positif = []
set netral = []
set negatif = []

for n in dataset:
  if(n[1] == 'Positif'):
    set positif.append(n)
  elif(n[1] == 'Negatif'):
    set_negatif.append(n)
  else:
    set_netral.append(n)

set_positif = random.sample(set_positif, k=int(len(set_positif)/2))
set_negatif = random.sample(set_negatif, k=int(len(set_negatif)/2))
set_netral = random.sample(set_netral, k=int(len(set_netral)/2))

train = set_positif + set_negatif + set_netral
train_set = []
for n in train:
  train_set.append(n)

data_tweet = list(data['review_english'])
polaritas = 0
status = []
total_positif = total_negatif = total_netral = total = 0
for i, tweet in enumerate(data_tweet):
  analysis = TextBlob(tweet, classifier=cl)
  if analysis.classify() == 'Positif':
    total_positif += 1
  elif analysis.classify() == 'Netral':
    total_netral += 1
  else:
    total_negatif += 1

  status.append(analysis.classify())
  total += 1

print(f'\nHasil Analisis Data: \nPositif = {total positif}\nNetral=
{total netral}\nNegatif = {total negatif}')
print(f'\nTotal Data: {total}')
```

```
status = pd.DataFrame({"Klasifikasi Bayes": status})
data['gmaps_klasifikasi_bayes'] = status
# set tema seaborn untuk tampilan yang lebih profesional
sns.set_theme()

labels = ['Positif', 'Netral', 'Negatif']
counts = [total_positif, total_netral, total_negatif]

def show_bar_chart(labels, counts, title):
  fig, ax = plt.subplots(figsize=(8, 6))
  bars = ax.bar(labels, counts, color=['#2394f7', '#fac343', '#f72323'])

  # menambahkan keterangan persentase
  for bar, count in zip(bars, counts):
    height = bar.get_height()
    ax.annotate(f'{count}', xy=(bar.get x() + bar.get width() / 2, height),
                xytext=(0, 3),
                textcoords="offset points",
                ha='center', va='bottom')
  # menambahkan grid
  ax.grid(axis='y', linestyle='--', alpha=0.7)

  # menambahkan label dari sumbu dan judul
  ax.set xlabel('Sentimen')
  ax.set_ylabel('Jumlah')
  ax.set title(title)
  plt.show()

show_bar_chart(labels, counts, "Distribusi Sentimen dari Review Google Maps
Menggunakan Naive Bayes")

cl = NaiveBayesClassifier(train set)
gmaps = cl.accuracy(dataset)
print("Akurasi Test: ", gmaps)

data_gmaps = data
data_gmaps.to_csv("Labeling_NB.csv", index=False)
```

Lampiran 6. Code script tahap analisis data eksploratif

```
# eksplorasi top 10 kata positif
from collections import Counter

data = pd.read csv('Labeling NB.csv')
positive reviews = data[data['gmaps klasifikasi bayes'] == 'Positif']
all_positive_words = ' '.join(positive_reviews['review'])

# Generate word cloud
wordcloud = WordCloud(
    width=3000,
    height=2000,
    random state=3,
    background_color='black',
    colormap='Blues_r',
    collocations=False,
    stopwords=STOPWORDS
).generate(all_positive_words)

# Plot word cloud
def plot_cloud(wordcloud):
    plt.figure(figsize=(10,8))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.show()

plot_cloud(wordcloud)

# Get word frequency
words = all_positive_words.split()
word_freq = Counter(words)

# Get top 10 words
top_10_words = word_freq.most_common(10)
print("Top 10 words in positive reviews:")
for word, freq in top 10 words:
    print(f"{word}: {freq}")

# Plot top 10 words
top 10 df = pd.DataFrame(top 10 words, columns=['Word', 'Frequency'])
sns.barplot(x='Frequency', y='Word', data=top_10_df)
plt.title('Top 10 Words in Positive Reviews')
plt.show()
```

```
# eksplorasi top 10 kata negatif
from collections import Counter

data = pd.read_csv('Labeling_NB.csv')
negative_reviews = data[data['gmaps_klasifikasi_bayes'] == 'Negatif']
all_negative_words = ' '.join(negative_reviews['review'])

# Generate word cloud
wordcloud = WordCloud(
    width=3000,
    height=2000,
    random_state=3,
    background_color='black',
    colormap='OrRd r',
    collocations=False,
    stopwords=STOPWORDS
).generate(all_negative_words)

# Plot word cloud
def plot cloud(wordcloud):
    plt.figure(figsize=(10,8))
    plt.imshow(wordcloud, interpolation='bilinear')
    plt.axis('off')
    plt.show()

plot cloud(wordcloud)

# Get word frequency
words = all_negative_words.split()
word freq = Counter(words)

# Get top 10 words
top_10_words = word_freq.most_common(10)
print("Top 10 words in negative reviews:")
for word, freq in top_10_words:
    print(f"{word}: {freq}")

# Plot top 10 words
top_10_df = pd.DataFrame(top_10_words, columns=['Word', 'Frequency'])
sns.barplot(x='Frequency', y='Word', data=top_10_df)
plt.title('Top 10 Words in Negative Reviews')
plt.show()
```