LAMPIRAN

Lampiran 1. Wawancara Bersama Pihak AKPK

Lampiran 2. Wawancara Bersama Pihak LP3M

Lampiran 3. Field Data Histori Mahaisiswa

Field	***Data Type***
nim	int64
nama_mahasiswa	object
kode_program_studi	int64
program_pendidikan	object
program_studi	object
ips	float64
sks_semester	int64
status_mahasiswa_semester	int64
id_periode	object

Lampiran 4. Field Data List Mahasiswa

Field	***Data Type***
nim	int64
nama_mahasiswa	object
jenis_kelamin	object
jalur_penerimaan	object
tahun_angkatan	int64
semester_masuk	float64
tanggal_mulai_masuk	object
kode_program_studi	int64
nama_program_studi	object
propinsi	object
kota_kab	object
kecamatan	object
ipk	float64
status_mahasiswa_terakhir	object
penerima_beasiswa_kipk	object
penerima_beasiswa_kjmu	object
kelompok_ukt	object
nominal_ukt	float64

Lampiran 5. Atribut Data Model dan Prediksi

Atribut Model	***Data Type***
ipk	float64
sks	int64
masa_studi	float64

Lampiran 6. Atribut Data Visualisasi Prediksi Mahasiswa

Atribut Prediksi	***Data Type***
nim	int64
nama_mahasiswa	object
nama_program_studi	object
tahun_angkatan	int64
sks	int64
ipk	float64
masa_studi	float64
status_mahasiswa	object

Lampiran 7. Source Code Model

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
import requests
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score
from sklearn.neighbors import KNeighborsClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.metrics import confusion_matrix
from sklearn.model_selection import cross_val_score
from math import sqrt
from sklearn.metrics import f1_score
from sklearn.metrics import recall_score
from sklearn.metrics import precision_score
from sklearn.metrics import precision_recall_fscore_support
from sklearn.naive_bayes import GaussianNB
from itertools import cycle
from scipy import interp
from sklearn import metrics

import requests
import pandas as pd
from datetime import datetime

# Nonaktifkan peringatan SSL jika diperlukan
requests.packages.urllib3.disable_warnings()

# API endpoint URL
url = ' '

# API key, secret key, and Basic Auth credentials
api_key_name = ' '
```

```
api_secret_key = ' '
authorization = ‘ '

headers = {
    'API_KEY_NAME': api_key_name,
    'API_SECRET_KEY': api_secret_key,
    'Authorization': authorization
}

# Mendapatkan tahun sekarang
current_year = datetime.now().year

# Membuat list untuk menyimpan DataFrame
dfs = []

# Looping untuk berbagai nilai id_periode dari tahun sekarang ke tahun 2015
for year in range(current_year, 2014, -1):
    for periode in [2, 1]:
        id_periode = f'{year}{periode}'
        parameter_list = {'id_periode': id_periode}

        # Kirim permintaan POST
        response = requests.post(url, headers=headers, data=parameter_list, verify=False)

        if response.status_code == 200:
            data = response.json()["data"]
            df = pd.DataFrame(data)
            dfs.append(df)
        else:
            print("Error:", response.status_code)

# Menggabungkan semua DataFrame menjadi satu DataFrame
data_mhs = pd.concat(dfs, ignore_index=True)
data_mhs
```

```python
import requests
import pandas as pd
from datetime import datetime

requests.packages.urllib3.disable_warnings()

url = ‘ ‘

# API key, secret key, and Basic Auth credentials
api_key_name = ‘ ‘
api_secret_key = ' '
authorization = ‘ '

headers = {
    'API_KEY_NAME': api_key_name,
    'API_SECRET_KEY': api_secret_key,
    'Authorization': authorization
}

# Mendapatkan tahun sekarang
current_year = datetime.now().year

# Membuat list untuk menyimpan DataFrame
dfs = []

for year in range(current_year, 2014, -1):
    angkatan = f'{year}'
    parameter_list = {'angkatan': angkatan}

    response = requests.post(url, headers=headers, data=parameter_list, verify=False)
    if response.status_code == 200:
        data = response.json()["data"]
        df = pd.DataFrame(data)
        dfs.append(df)
    else:
```

```
        print("Error:", response.status_code)

# Menggabungkan semua DataFrame menjadi satu DataFrame
data_ipk = pd.concat(dfs, ignore_index=True)
data_ipk

# Filter program pendidikan S1
data_s1_only = data_mhs[data_mhs['program_pendidikan'] == 'S1']
data_s1_only

# Label Encoding
label_encoder = LabelEncoder()

for c in data_s1_only.columns:
    if data_s1_only[c].dtype == 'object':
        data_s1_only.loc[:, c] = label_encoder.fit_transform(data_s1_only[c].values)

data_s1_only

# Merge Dataframe df_mhs & df_ipk
data_mahasiswa = pd.merge(data_s1_only.drop(columns=['ips']),
                          data_ipk[['nim', 'ipk']], on='nim', how='inner')
data_mahasiswa

# Create DataFrame sks
sks = data_s1_only.groupby('nim')['sks_semester'].sum().reset_index()
sks.columns = ['nim', 'sks']
sks

# Merge DataFrame sks & ipk
sks_with_ipk = pd.merge(sks, data_ipk[['nim', 'ipk']], on='nim', how='inner')
sks_with_ipk

# Create Masa Studi Attribute
```

```
import datetime

current_year = datetime.datetime.now().year

# Initial digits
first_nim = data_s1_only.groupby('nim')['id_periode'].min().reset_index()
first_nim.columns = ['nim', 'first_id_periode']

# Occurrences of ID periods
id_periode_count =
data_s1_only.groupby(['nim']).size().reset_index(name='id_periode_count')
merged_df = pd.merge(data_s1_only, first_nim, on='nim', how='left')
merged_df['tahun_masuk'] = merged_df['first_id_periode'].astype(str).str[:4].astype(int)

# Counting cuti & non-aktif
inactive_status = merged_df[merged_df['status_mahasiswa_semester'].isin(['cuti', 'non-
aktif'])].groupby('nim').size().reset_index(name='inactive_count')

merged_df = pd.merge(merged_df, inactive_status, on='nim', how='left')
merged_df['inactive_count'].fillna(0, inplace=True)

merged_df = pd.merge(merged_df, id_periode_count, on='nim', how='left')
merged_df['id_periode_count'].fillna(0, inplace=True)

merged_df['masa_studi'] = merged_df['id_periode_count'] - merged_df['inactive_count']
merged_df[['nim', 'masa_studi']]

# Merge DataFrame merged_df & sks_with_ipk
merged_df_all = pd.merge(merged_df[['nim', 'masa_studi']], sks_with_ipk, on='nim',
how='inner')

merged_df_all

# Removing duplicates based on the 'nim' column value
df_mahasiswa = merged_df_all.drop_duplicates(subset=['nim'], ignore_index=True)
```

```
df_mahasiswa

# Inisialisasi kolom 'status_mahasiswa' dengan nilai 'prediksi dropout' secara default
df_mahasiswa['status_mahasiswa'] = 'terprediksi dropout'

# Cek kriteria untuk 'mahasiswa aktif' dan set nilai 'status_mahasiswa' berdasarkan kriteria
for index, row in df_mahasiswa.iterrows():
    if row['masa_studi'] == 2.0 and row['sks'] >= 19:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 3.0 and row['sks'] >= 40:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 4.0 and row['sks'] >= 40:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 5.0 and row['sks'] >= 60:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 6.0 and row['sks'] >= 60:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 7.0 and row['sks'] >= 80:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 8.0 and row['sks'] >= 80:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 9.0 and row['sks'] >= 100:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 10.0 and row['sks'] >= 100:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'
```

```
    elif row['masa_studi'] == 11.0 and row['sks'] >= 120:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 12.0 and row['sks'] >= 120:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 13.0 and row['sks'] >= 144:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

    elif row['masa_studi'] == 14.0 and row['sks'] >= 144:
        df_mahasiswa.at[index, 'status_mahasiswa'] = 'tidak terprediksi dropout'

# Set nilai 'status_mahasiswa' berdasarkan kriteria dropout
criteria_dropout = (
    (df_mahasiswa['masa_studi'] >= 14.0) |
    (df_mahasiswa['ipk'] < 2.00) |
    (df_mahasiswa['masa_studi'] == 0.0) |
    (df_mahasiswa['sks'] == 0.0) |
    (df_mahasiswa['ipk'] == 0.00)
)

df_mahasiswa.loc[criteria_dropout, 'status_mahasiswa'] = 'terprediksi dropout'
df_mahasiswa[['nim', 'masa_studi', 'sks', 'ipk', 'status_mahasiswa']]

X = df_mahasiswa[['masa_studi', 'sks', 'ipk']]
y = df_mahasiswa['status_mahasiswa']

#tuning
knn = KNeighborsClassifier(n_neighbors=5)
scores = cross_val_score(knn, X, y, cv=5)

print("Accuracy:", scores)
print("Mean Accuracy:", scores.mean())

#splitting data
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state = 420)

#scalling data
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)

#training model
classifier = KNeighborsClassifier(n_neighbors = 5, metric = 'minkowski', p = 2)#The default metric is minkowski, and with p=2 is equivalent to the standard Euclidean metric.
classifier.fit(X_train, y_train)

y_train_pred = classifier.predict(X_train)
y_test_pred = classifier.predict(X_test)

cm = confusion_matrix(y_train, y_train_pred)
print(cm)
accuracy_score(y_train, y_train_pred)

cm = confusion_matrix(y_test, y_test_pred)
print(cm)
accuracy_score(y_test, y_test_pred)

from sklearn.metrics import classification_report
print(classification_report(y_test, y_test_pred))
```

Lampiran 8. Source Code Website dan Prediksi Model

1. *HTML <dropout mahasiswa>*

```
<!DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="UTF-8" />
   <meta name="viewport" content="width=device-width, initial-scale=1.0" />

   <!-- TITLE -->
   <title>Dashboard UPN "Veteran" Jakarta</title>

   <!-- CONNECT CSS -->
   <link rel="stylesheet" href="./static/css/dropoutmahasiswa.css" />

   <!-- ICON ACTIVATION -->
   <link
     rel="stylesheet"
     href="https://cdnjs.cloudflare.com/ajax/libs/font-awesome/6.5.1/css/all.min.css"
   />

   <!-- BOOTSTRAP CSS -->
   <link
     href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/css/bootstrap.min.css"
     rel="stylesheet"
     integrity="sha384-
QWTKZyjpPEjISv5WaRU9OFeRpok6YctnYmDr5pNlyT2bRjXh0JMhjY6hW+ALEwI
H"
     crossorigin="anonymous"
   />
 </head>

 <body>
   <!-- BOOTSTRAP BUNDLE -->
   <script
```

```
    src="https://cdn.jsdelivr.net/npm/bootstrap@5.3.3/dist/js/bootstrap.bundle.min.js"
    integrity="sha384-
YvpcrYf0tY3lHB60NNkmXc5s9fDVZLESaAA55NDzOxhy9GkcIdslK1eN7N6jIeHz"
    crossorigin="anonymous"
  ></script>

  <!-- HEADER -->
  {% include 'header.html' %}

  <!-- MAIN PAGE -->
  <div class="container-main">
    {% include 'sidemenu.html' %}

    <!-- CONTENT - RIGHT -->
    <!-- ========================================== CHANGE
CODE START FROM HERE
========================================== -->
    <main id="main-page">
      <!-- <div class="dashboard-menu fs-5 px-4 py-2 d-flex">
        <i class="bar-expand fa-solid fa-bars"></i>
        <p class="subtitle fs-6 fw-bold">Dashboard Kelulusan Mahasiswa</p>
      </div> -->
      <div class="dashboard-menu fs-5 px-4 d-flex">
        <i class="bar-expand fa-solid fa-bars"></i>
        <p class="subtitle fs-6 fw-bold">Dashboard Dropout Mahasiswa</p>
      </div>


      <!-- CHART -->
      <br>
      <div id="cards-container" class="container-fluid py-3">
        <div id="status-title" style="font-family: 'Montserrat', sans-serif; text-align:
center; font-weight: bold;">Status Mahasiswa S1 UPNVJ (2015 - Saat Ini)</div>
        <div class="boxes d-flex">
          <div id="non-aktif" class="card">
```

```
            <div class="card-body" data-status="NON-AKTIF">NON-AKTIF</div>
          </div>
          <div id="drop-out" class="card">
            <div class="card-body" data-status="DROP-OUT/PUTUS STUDI">DROP-OUT/PUTUS STUDI</div>
          </div>
          <div id="aktif" class="card">
            <div class="card-body" data-status="AKTIF">AKTIF</div>
          </div>
          <div id="menunggu-isi-krs" class="card">
            <div class="card-body" data-status="MENUNGGU ISI KRS">MENUNGGU ISI KRS</div>
          </div>
          <div id="keluar" class="card">
            <div class="card-body" data-status="KELUAR">KELUAR</div>
          </div>
          <div id="cuti" class="card">
            <div class="card-body" data-status="CUTI">CUTI</div>
          </div>
          <div id="kampus-merdeka" class="card">
            <div class="card-body" data-status="KAMPUS MERDEKA">KAMPUS MERDEKA</div>
          </div>
          <div id="lulus" class="card">
            <div class="card-body" data-status="LULUS">LULUS</div>
          </div>
          <div id="tanpa-keterangan" class="card">
            <div class="card-body" data-status="TANPA KETERANGAN">TANPA KETERANGAN</div>
          </div>
        </div>
        <br>
        <div id="status_caption" style="font-family: 'Montserrat', sans-serif; text-align: center;">
          Terdapat sembilan status mahasiswa S1 Universitas Pembangunan Nasional
```

```
Veteran Jakarta, diantaranya adalah
            NON-AKTIF, DROP-OUT/PUTUS STUDI, AKTIF, MENUNGGU ISI KRS,
KELUAR, CUTI, KAMPUS MERDEKA, LULUS, DAN TANPA KETERANGAN.
          </div>
      </div>
<br>
      <div id="chart"></div>
      <div id="chart_caption" style="font-family: 'Montserrat', sans-serif; text-align:
center;">
        Bar Chart diatas menggambarkan jumlah dropout setiap program studi pada
mahasiswa S1 Universitas Pembangunan Nasional
        Veteran Jakarta pada tahun 2015 hingga saat ini.
      </div>

<br>
      <div id="line-chart"></div>
      <div id="linechart_caption" style="font-family: 'Montserrat', sans-serif; text-align:
center;">
        Line chart diatas menggambarkan jumlah Dropout mahasiswa S1 Universitas
Pembangunan Nasional Veteran Jakarta Per-tahun.
      </div>
<br>
      <div id="bar-chart"></div>
      <div id="barchart_caption" style="font-family: 'Montserrat', sans-serif; text-align:
center;">
        Chart diatas menggambarkan jumlah dan persentase dari prediksi mahasiswa S1
Universitas Pembangunan Nasional Veteran Jakarta
        yang teridentifikasi dropout dan yang tidak teridentifikasi dropout.
      </div>
<br>
<br>
          <!-- FILTER -->
        <div class="filter d-flex px-2">
          <div class="dropdown px-2">
            <button
```

```
      id = "tahun_akademik"
      class="btn btn-secondary dropdown-toggle"
      type="button"
      data-bs-toggle="dropdown"
      aria-expanded="false"
    >
      Tahun
    </button>
    <ul id="filter" class="dropdown-menu">
      <!-- <li><a class="dropdown-item" href="#">All</a></li> -->
      <li><a class="dropdown-item" href="#">2017</a></li>
      <li><a class="dropdown-item" href="#">2018</a></li>
      <li><a class="dropdown-item" href="#">2019</a></li>
      <li><a class="dropdown-item" href="#">2020</a></li>
      <li><a class="dropdown-item" href="#">2021</a></li>
      <li><a class="dropdown-item" href="#">2022</a></li>
      <li><a class="dropdown-item" href="#">2023</a></li>
    </ul>
  </div>


  <div class="dropdown px-2">
    <button
      id = "prodi"
      class="btn btn-secondary dropdown-toggle"
      type="button"
      data-bs-toggle="dropdown"
      aria-expanded="false"
    >
      Program Studi
    </button>
    <ul id="filter_prodi" class="dropdown-menu">
      <!-- <li>
        <a class="dropdown-item" href="#">All</a>
      </li> -->
```

```
<li>
  <a class="dropdown-item" href="#">S1 Mananjemen</a>
</li>
<li><a class="dropdown-item" href="#">S1 Akuntansi</a></li>
<li><a class="dropdown-item" href="#">S1 Kedokteran</a></li>
<li>
  <a class="dropdown-item" href="#">S1 Teknik Mesin</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Teknik Industri</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Teknik Perkapalan</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Ilmu Komunikasi</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Hubungan Internasional</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Informatika</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Sistem Informasi</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Hukum</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Keperawatan</a>
</li>
<li>
  <a class="dropdown-item" href="#">S1 Kesehatan Masyarakat</a>
</li>
```

```
            <li>
              <a class="dropdown-item" href="#">S1 Gizi</a>
            </li>
            <li>
              <a class="dropdown-item" href="#">S1 Ekonomi Pembangunan</a>
            </li>
            <li>
              <a class="dropdown-item" href="#">S1 Ekonomi Syariah</a>
            </li>
            <li>
              <a class="dropdown-item" href="#">S1 Farmasi</a>
            </li>
            <li>
              <a class="dropdown-item" href="#">S1 Teknik Elektro</a>
            </li>
            <li>
              <a class="dropdown-item" href="#">S1 Ilmu Politik</a>
            </li>
            <li>
              <a class="dropdown-item" href="#">S1 Sains Informasi</a>
            </li>
          </ul>
        </div>
      </div>

      <br />
      <h1 style="color: white;">Data Prediksi Mahasiswa Teridentifikasi DROP-
OUT</h1>
      <div class="table-container">
        <table id="prediction-table" class="table">
          <thead>
            <tr>
              <th>Nim</th>
              <th>Nama Mahasiswa</th>
              <th>Program Studi</th>
```

```
            <th>Tahun Angkatan</th>
            <th>SKS</th>
            <th>IPK</th>
            <th>Masa Studi</th>
            <th>Status Mahasiswa</th>
          </tr>
        </thead>
        <tbody>
          <!-- Data tabel -->
        </tbody>
      </table>
    </div>

  </div>
 </main>
</div>

<!-- FOOTER -->
{% include 'footer.html' %}

<!-- APEXCHART -->
<script src="https://cdn.jsdelivr.net/npm/apexcharts"></script>

<!-- CONNECT JS -->
<script src="./static/js/dropoutmahasiswa.js"></script>
</body>
    </html>
```

2. *HTML <header>*

```
<!-- HEADER -->
<header class="navbar">
 <div class="container-fluid">
  <div class="logo d-flex px-2 align-self-center">
```

```
    <img src="../static/images/logoupnvj.png" alt="LOGO-UPNVJ" />

    <div class="fs-6 fw-bold px-3 logo-name align-self-center">
      DASHBOARD <br />
      UPN "VETERAN" JAKARTA
    </div>
   </div>
  </div>
</header>
```

3. *HTML <footer>*

```
<!-- FOOTER -->
<footer class="fw-bold pt-2">
 Copyright © 2024 ASA Group All rights reserved
</footer>
```

4. *CSS*

```
@import url("https://fonts.googleapis.com/css2?family=Montserrat&display=swap");

/* COLORS VARIABLE */
:root {
  --darkgreen: #116611;
  --lightgreen: #e7efe7;
  --lightgrey: #cccccc;
}

/* ALL STYLE */
* {
  margin: 0;
  padding: 0;
  font-family: "Montserrat", sans-serif;
  box-sizing: border-box;
}
```

```css
html,
body {
  height: 100%;
  width: 100%;
}

/* HEADER */
header {
  background-color: var(--darkgreen);
  color: white;
  height: 80px;
}

.logo {
  height: 55px;
}

.logo-name {
  letter-spacing: 0.5px;
}

/* MAIN PAGE */
.container-main {
  display: flex;
  height: calc(100% - 120px);
}

/* SIDEBAR - LEFT */
nav {
  background-color: var(--lightgreen);
  width: 355px;
  height: 100%;
  overflow: auto;
  transition: width 0.3s ease;
}
```

```css
.menu-navigationbar {
  text-decoration: none;
  color: black;
  display: flex;
}

.menu-navigationbar:hover {
  color: var(--darkgreen);
  background-color: var(--lightgrey);
}

.submenu {
  font-size: 15px;
  letter-spacing: 0.5px;
}

/* BAR EXPAND */
.bar-expand {
  cursor: pointer;
}

.bar-expand:hover {
  color: var(--darkgreen);
}

.dashboard-menu {
  padding-top: 12px;
  justify-content: space-between;
}

.dashboard-menu p {
  letter-spacing: 0.5px;
}
```

```css
/* BAR EXPAND - HIDE */
#sidebar.collapsed {
  width: 0;
  overflow: hidden;
}

#main-page.expanded {
  margin-left: 0;
}

/* CONTENT */
#main-page {
  flex-grow: 1;
  overflow-y: auto;
  position: relative;
  transition: width 0.3s ease;
  width: 100vh;
}

.filter {
  justify-content: end;
}

.btn-secondary {
  background-color: var(--darkgreen) !important;
}

.boxes {
  justify-content: center;
}

.cards-container {
  padding-left: 10px;
}
```

```css
.card {
  width: 140px;
  margin: 10px;
  border: 1px solid #ddd;
  border-radius: 8px;
  box-shadow: 5px 5px 10px rgba(0, 100, 0, 0.5);
  transition: box-shadow 0.2s;
  padding: 10px;
  box-sizing: border-box;
}

.card:hover {
  box-shadow: 8px 8px 15px rgba(0, 100, 0, 0.7);
}

.card-body .title {
  font-size: 11px;
  font-weight: bold;
  margin-bottom: 10px;
}

.card-body {
  text-align: center;
  background-color: var --darkgreen;
  padding: 10px;
}

.card-body .content {
  font-size: 12px;
  margin-bottom: 5px;
}

#status-title {
  font-family: "Montserrat", sans-serif;
  text-align: center;
```

```
  font-size: 20px;
  font-weight: bold;
  padding: 10px 0;
  margin-bottom: 20px;
}

/* Responsive design */
@media (max-width: 600px) {
  .card {
    width: 100%;
    margin: 5px 0;
  }
}

.chart-container {
  width: 80%;
  max-width: 800px;
  margin-bottom: 20px;
}

.pie {
  justify-content: center;
}

#pie-chart,
#pie-chart2,
#pie-chart3 {
  width: 300px;
}

h1 {
  text-align: center;
  color: white;
  background-color: var(--darkgreen);
  padding: 10px 0;
```

```
}

.table-container {
  max-height: 500px;
  overflow-y: auto;
}

#prediction-table th {
  background-color: #f2f2f2;
  position: sticky;
  top: 0;
}

.table {
  border-collapse: collapse;
  width: 100%;
}

.table th,
.table td {
  border: 1px solid #ddd;
  padding: 8px;
  text-align: left;
}

.table th {
  background-color: #f2f2f2;
}

.filter {
  justify-content: end;
}

.btn-secondary {
  width: 300px !important;
```

```css
}

.dropdown-menu {
  width: 300px !important;
  max-height: 250px;
  overflow-y: auto;
}

.dropdown-item {
  text-align: center !important;
}

.dropdown-item.active,
.btn-secondary,
.btn-primary {
  background-color: var(--darkgreen) !important;
}

#filter {
  position: sticky;
  top: 50px;
  z-index: 999;
}

/* FOOTER */
footer {
  height: 40px;
  font-size: 15px;
  background-color: var(--lightgrey);
  text-align: center;
}

table {
  width: 100%;
  border-collapse: collapse;
```

```
}
/* th,
```

5. *JavaScript*

```
document.addEventListener("DOMContentLoaded", function () {
  var barExpand = document.querySelector(".bar-expand");
  var sidebar = document.getElementById("sidebar");
  var mainPage = document.getElementById("main-page");

  barExpand.addEventListener("click", function () {
    sidebar.classList.toggle("collapsed");
    mainPage.classList.toggle("expanded");
  });
});

const tableBody = document.querySelector("#prediction-table tbody");
const fetchButton = document.getElementById("fetch-button");
const year = "2015";

function all() {
  function fetchData(year) {
    fetch(`/dropout/predict/${year}`)
      .then((response) => {
        if (!response.ok) {
          throw new Error(`HTTP error! status: ${response.status}`);
        }
        return response.json();
      })
      .then((data) => {
        const jsonObject = JSON.parse(data.data_prediction_json);
        displayData(jsonObject);
      })
      .catch((error) =>
        console.error("Error fetching prediction data:", error)
```

```
    );
}

function displayData(data) {
  tableBody.innerHTML = "";

  data.forEach((item) => {
    const row = document.createElement("tr");
    row.innerHTML = `
      <td>${item.nim}</td>
      <td>${item.nama_mahasiswa}</td>
      <td>${item.nama_program_studi}</td>
      <td>${item.tahun_angkatan}</td>
      <td>${item.sks}</td>
      <td>${item.ipk}</td>
      <td>${item.masa_studi}</td>
      <td>${item.status_mahasiswa}</td>
    `;
    tableBody.appendChild(row);
  });
}

fetchData(year);

// FOR CARDS
document.addEventListener("DOMContentLoaded", (event) => {
  function fetchData(year) {
    fetch(`/dropout/persentase_status_mahasiswa/${year}`)
      .then((response) => {
        if (!response.ok) {
          throw new Error(`HTTP error! status: ${response.status}`);
        }
        return response.json();
      })
      .then((data) => {
```

```
      updateCards(data.persentase);
    })
    .catch((error) => console.error("Error fetching data:", error));
}

// Update cards with the fetched data
function updateCards(data) {
  const statusToId = {
    "NON-AKTIF": "non-aktif",
    "DROP-OUT/PUTUS STUDI": "drop-out",
    AKTIF: "aktif",
    "MENUNGGU ISI KRS": "menunggu-isi-krs",
    KELUAR: "keluar",
    CUTI: "cuti",
    "KAMPUS MERDEKA": "kampus-merdeka",
    LULUS: "lulus",
    "TANPA KETERANGAN": "tanpa-keterangan",
  };

  data.forEach((item) => {
    const cardId = statusToId[item.status];
    if (cardId) {
      const card = document.getElementById(cardId);
      if (card) {
        card.querySelector(".card-body").innerHTML = `
        <div class="title">${item.status}</div><br>
        <div class="content">Jumlah: ${item.jumlah}</div>
        <div class="content">Persentase: ${item.persentase}%</div>
      `;
      }
    }
  });
}

fetchData(year);
```

```
});
// FOR LINE AND BAR CHART
// bar chart2
document.addEventListener("DOMContentLoaded", () => {
  function fetchData(year) {
    return fetch(`/dropout/prodi_chart/${year}`).then((response) => {
      if (!response.ok) {
        throw new Error(`HTTP error! status: ${response.status}`);
      }
      return response.json();
    });
  }
  const predefinedColors = ["#1f77b4"];

  // Update chart with the fetched data
  function updateChart(year) {
    fetchData(year)
      .then((data) => {
        var chartData = {
          series: [
            {
              name: "Jumlah Dropout",
              data: data.jumlah_dropout,
            },
          ],
          chart: {
            height: 400,
            type: "bar",
            fontFamily: '"Montserrat", sans-serif',
          },
          colors: predefinedColors,
          plotOptions: {
            bar: {
              horizontal: false,
              columnWidth: "55%",
```

```
    endingShape: "rounded",
    dataLabels: {
      position: "top",
    },
  },
},
dataLabels: {
  enabled: true,
  formatter: function (val) {
    return val;
  },
  style: {
    fontSize: "12px",
    colors: ["#000"],
  },
  offsetY: -20,
  position: "top",
},
title: {
  text: "Jumlah Dropout Per-Program Studi S1 UPNVJ (2015 - Saat Ini)",
  align: "center",
  style: {
    fontSize: "20px",
    fontWeight: "bold",
  },
},
xaxis: {
  categories: data.program_studi,
  title: {
    text: "Program Studi",
    style: {
      fontWeight: "bold",
    },
  },
  labels: {
```

```
      style: {
        fontSize: "12px",
        fontWeight: "bold",
      },
    },
    axisBorder: {
      show: true,
      color: "#cccccc",
      height: 1,
      width: "100%",
      offsetX: 0,
      offsetY: 0,
    },
  },
  yaxis: {
    title: {
      text: "Jumlah Dropout",
      style: {
        fontWeight: "bold",
      },
    },
    labels: {
      style: {
        fontSize: "12px",
        fontWeight: "bold",
      },
    },
    axisBorder: {
      show: true,
      color: "#cccccc",
      width: 1,
      offsetX: 0,
      offsetY: 0,
    },
  },
```

```
        grid: {
          borderColor: "#e0e0e0",
          strokeDashArray: 0,
          xaxis: {
            lines: {
              show: false,
            },
          },
          yaxis: {
            lines: {
              show: true,
            },
          },
        },
        tooltip: {
          y: {
            formatter: function (val) {
              return val;
            },
          },
        },
      };

      var chart = new ApexCharts(
        document.querySelector("#chart"),
        chartData
      );
      chart.render();
    })
    .catch((error) => console.error("Error fetching data:", error));
  }
  updateChart(year);
});

document.addEventListener("DOMContentLoaded", () => {
```

```
// Fetch and render line chart
function fetchLineChartData(year) {
  console.log(`Fetching line chart data for year: ${year}`);
  fetch(`/dropout/jumlah_dropout_yearly/${year}`)
    .then((response) => {
      if (!response.ok) {
        throw new Error(`HTTP error! status: ${response.status}`);
      }
      return response.json();
    })
    .then((data) => {
      console.log("Line chart data received:", data);
      const categories = Object.keys(data.jumlah_dropout_per_tahun);
      const seriesData = Object.values(data.jumlah_dropout_per_tahun);

      var lineChartOptions = {
        series: [
          {
            name: "Dropout Count",
            data: seriesData,
          },
        ],
        chart: {
          height: 350,
          type: "line",
          zoom: {
            enabled: false,
          },
          fontFamily: '"Montserrat", sans-serif',
        },
        dataLabels: {
          enabled: true,
          formatter: function (val) {
            return val;
          },
```

```
    style: {
      fontSize: "12px",
      colors: ["#000"],
    },
    background: {
      enabled: false,
    },
    dropShadow: {
      enabled: false,
    },
    offsetY: -10,
  },
  stroke: {
    curve: "straight",
    colors: ["#FF0000"],
    width: 3,
  },
  markers: {
    size: 5,
    colors: ["#FF0000"],
    strokeColors: "#fff",
    strokeWidth: 2,
    hover: {
      size: 7,
    },
  },
  title: {
    text: "Jumlah Dropout Per-tahun S1 UPNVJ (2015 - Saat Ini)",
    align: "center",
    style: {
      fontSize: "20px",
      color: "#333",
    },
  },
  grid: {
```

```
    row: {
      colors: ["rgba(255, 0, 0, 0.2)", "rgba(255, 0, 0, 0)"],
      opacity: 0.5,
    },
    borderColor: "#ffffff",
  },
  xaxis: {
    categories: categories,
    title: {
      text: "Year",
      style: {
        fontSize: "12px",
        fontWeight: "bold",
        color: "#333",
      },
    },
    labels: {
      style: {
        colors: "#333",
        fontSize: "12px",
        fontWeight: "bold",
      },
    },
  },
  yaxis: {
    title: {
      text: "Dropout Count",
      style: {
        fontSize: "12px",
        fontWeight: "bold",
        color: "#333",
      },
    },
    labels: {
      style: {
```

```
            colors: "#333",
            fontSize: "12px",
            fontWeight: "bold",
          },
        },
      },
      tooltip: {
        y: {
          formatter: function (val) {
            return `Jumlah Dropout: ${val}`;
          },
          style: {
            fontSize: "12px",
            color: "#000",
          },
        },
        theme: "light",
        marker: {
          show: true,
          fillColors: ["#FF0000"],
        },
      },
    };

    var lineChart = new ApexCharts(
      document.querySelector("#line-chart"),
      lineChartOptions
    );
    lineChart.render();
  })
  .catch((error) => {
    console.error("Error fetching line chart data:", error);
  });
}
// Fetch and render bar chart
```

```
function fetchBarChartData(year) {
  console.log(`Fetching bar chart data for year: ${year}`);
  fetch(`/dropout/presentase_predict/${year}`)
    .then((response) => {
      if (!response.ok) {
        throw new Error(`HTTP error! status: ${response.status}`);
      }
      return response.json();
    })
    .then((data) => {
      console.log("Bar chart data received:", data);
      const categories = [
        "Terprediksi Dropout",
        "Tidak Terprediksi Dropout",
      ];
      const seriesData = [
        data.terprediksi_dropout.jumlah,
        data.tidak_terprediksi_dropout.jumlah,
      ];
      const percentages = [
        data.terprediksi_dropout.persentase,
        data.tidak_terprediksi_dropout.persentase,
      ];

      var barChartOptions = {
        series: [
          {
            name: "Jumlah",
            data: seriesData,
          },
        ],
        chart: {
          type: "bar",
          height: 350,
          fontFamily: '"Montserrat", sans-serif',
```

```
    },
    plotOptions: {
      bar: {
        borderRadius: 4,
        horizontal: true,
        colors: {
          ranges: [
            {
              from: 0,
              to: seriesData[0],
              fontWeight: "bold",
              color: "#ff4d4d",
            },
            {
              from: seriesData[0] + 1,
              to: seriesData[1] + seriesData[0],
              fontWeight: "bold",
              color: "#4caf50",
            },
          ],
        },
      },
    },
    dataLabels: {
      enabled: false,
    },
    xaxis: {
      categories: categories,
      title: {
        text: "Jumlah",
        style: {
          fontWeight: "bold",
        },
      },
      labels: {
```

```
      style: {
        colors: "#000000",
        fontSize: "12px",
        fontWeight: "bold",
      },
    },
  },
  yaxis: {
    title: {
      text: "Status",
      style: {
        fontWeight: "bold",
      },
    },
    labels: {
      style: {
        colors: "#000000",
        fontSize: "12px",
        fontWeight: "bold",
      },
    },
  },
  title: {
    text: "Persentase Prediksi Dropout Mahasiswa Aktif S1 UPNVJ",
    align: "center",
    style: {
      fontSize: "20px",
      fontWeight: "bold",
    },
  },
  tooltip: {
    enabled: true,
    shared: false,
    y: {
      formatter: function (val, { dataPointIndex }) {
```

```
            const percentage = percentages[dataPointIndex].toFixed(2);
            return `Jumlah: ${val}, Persentase: ${percentage}%`;
          },
        },
      },
      grid: {
        borderColor: "#e7e7e7",
        strokeDashArray: 4,
      },
    };

    var barChart = new ApexCharts(
      document.querySelector("#bar-chart"),
      barChartOptions
    );
    barChart.render();
  })
  .catch((error) => {
    console.error("Error fetching bar chart data:", error);
  });
 }

 fetchLineChartData(year);
 fetchBarChartData(year);
});
}
all();

// FILTER TABLE
function updatedChart(tahun_akademik, prodi) {
 fetch(`/dropout/predict/${year}`)
  .then((response) => {
   if (!response.ok) {
    throw new Error(`HTTP error! status: ${response.status}`);
   }
```

```
    return response.json();
  })
  .then((data) => {
    console.log("Received data:", data);
    const jsonObject = JSON.parse(data.data_prediction_json);

    if (Array.isArray(jsonObject)) {
      const filteredData = jsonObject.filter(
        (item) =>
          (tahun_akademik === "All" ||
            item.tahun_angkatan === tahun_akademik) &&
          (prodi === "All" || item.nama_program_studi === prodi)
      );
      updateTable(filteredData);
    } else {
      console.error("Expected an array, but received:", jsonObject);
    }
  })
  .catch((error) => console.error("Error fetching/parsing data:", error));
}

function updateTable(data) {
 const tableBody = document.querySelector("#prediction-table tbody");
 tableBody.innerHTML = "";

 if (Array.isArray(data)) {
  data.forEach((item) => {
    const row = document.createElement("tr");
    row.innerHTML = `
      <td>${item.nim}</td>
      <td>${item.nama_mahasiswa}</td>
      <td>${item.nama_program_studi}</td>
      <td>${item.tahun_angkatan}</td>
      <td>${item.sks}</td>
      <td>${item.ipk}</td>
```

```
        <td>${item.masa_studi}</td>
        <td>${item.status_mahasiswa}</td>
      `;

      tableBody.appendChild(row);
    });
  } else {
    console.error("Data is not an array:", data);
  }
}

document.addEventListener("DOMContentLoaded", function () {
  var filter_tahun = document.getElementById("filter");
  var filter_prodi = document.getElementById("filter_prodi");

  var selectedTahun = "All";
  var selectedProdi = "All";

  filter_tahun.addEventListener("click", function (e) {
    var target = e.target;

    if (target.tagName === "A") {
      var newSelectedTahun = target.textContent;

      if (newSelectedTahun === selectedTahun) {
        selectedTahun = "All";
        document.getElementById("tahun_akademik").textContent = "Tahun";
        filter_tahun
          .querySelectorAll("a")
          .forEach((a) => a.classList.remove("active"));
        filter_tahun.querySelector("a").classList.add("active");
      } else {
        selectedTahun = newSelectedTahun;
        document.getElementById("tahun_akademik").textContent = selectedTahun;
        filter_tahun
```

```
        .querySelectorAll("a")
        .forEach((a) => a.classList.remove("active"));
      target.classList.add("active");
    }

    switch (selectedTahun) {
      case "All":
        all();
        break;
      case "2017":
      case "2018":
      case "2019":
      case "2020":
      case "2021":
      case "2022":
      case "2023":
        updatedChart(selectedTahun, selectedProdi);
        break;
      default:
        console.error("Tahun akademik tidak dikenal:", selectedTahun);
    }
  }
});

filter_prodi.addEventListener("click", function (e) {
  var target = e.target;

  if (target.tagName === "A") {
    var newSelectedProdi = target.textContent;

    if (newSelectedProdi === selectedProdi) {
      selectedProdi = "All";
      document.getElementById("prodi").textContent = "Program Studi";
      filter_prodi
        .querySelectorAll("a")
```

```
    .forEach((a) => a.classList.remove("active"));
  filter_prodi.querySelector("a").classList.add("active");
 } else {
  selectedProdi = newSelectedProdi;
  document.getElementById("prodi").textContent = selectedProdi;
  filter_prodi
    .querySelectorAll("a")
    .forEach((a) => a.classList.remove("active"));
  target.classList.add("active");
 }

 switch (selectedProdi) {
  case "All":
   all();
   break;
  case "S1 Mananjemen":
  case "S1 Akuntansi":
  case "S1 Kedokteran":
  case "S1 Teknik Mesin":
  case "S1 Teknik Industri":
  case "S1 Teknik Perkapalan":
  case "S1 Ilmu Komunikasi":
  case "S1 Hubungan Internasional":
  case "S1 Informatika":
  case "S1 Sistem Informasi":
  case "S1 Hukum":
  case "S1 Keperawatan":
  case "S1 Kesehatan Masyarakat":
  case "S1 Gizi":
  case "S1 Ekonomi Pembangunan":
  case "S1 Ekonomi Syariah":
  case "S1 Farmasi":
  case "S1 Teknik Elektro":
  case "S1 Ilmu Politik":
  case "S1 Sains Informasi":
```

```
        updatedChart(selectedTahun, selectedProdi);
        break;
      default:
        console.error("Program studi tidak dikenal:", selectedProdi);
    }
  }
});
    });
```

6. *Python*

```
from flask import Blueprint, render_template, json, jsonify
import os
import pickle
import requests
import numpy as np
import pandas as pd
from datetime import datetime
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import StandardScaler

# Defining a blueprint
dropout_bp = Blueprint(
    'dropout_bp', __name__,
    template_folder='templates',
    static_folder='static',
    url_prefix='/dropout'
)

# Initialize API key, secret key, and Basic Auth credentials
api_key_name = ' '
api_secret_key = ' '
authorization = ' '
# Headers
headers = {
```

```
'API_KEY_NAME': api_key_name,
'API_SECRET_KEY': api_secret_key,
'Authorization': authorization,
}


@dropout_bp.route('/')
def main_dropout():
    current_year = datetime.now().year
    for year in range(current_year, 2014, -1):
        year = f'{year}'

    if check_file_predict(year):
        return render_template(f'dropoutmahasiswa.html')
    else:
        return get_data_endpoint(year)

@dropout_bp.route('/')
def visualization():
    current_year = datetime.now().year
    for year in range(current_year, 2014, -1):
        year = f'{year}'

    if check_file_data_mahasiswa(year):
        return render_template(f'dropoutmahasiswa.html')
    else:
        return get_histori_mahasiswa(year)

@dropout_bp.route('/get_data/<year>/', methods=['GET'])
def get_data_endpoint(year):
    histori_mahasiswa = get_histori_mahasiswa(‘ ‘)
    list_mahasiswa = get_list_mahasiswa(‘ ‘)

    data_s1_only = histori_mahasiswa[histori_mahasiswa['program_pendidikan'] == 'S1']
```

```
    data_mahasiswa = pd.merge(data_s1_only.drop(columns=['ips']),
                              list_mahasiswa[['nim', 'ipk', 'status_mahasiswa_terakhir',
'tahun_angkatan',
                                              'nama_program_studi']], on='nim', how='inner')
    df_pred = preprocess_data(data_mahasiswa)

    data_mahasiswa_json = df_pred.to_json(orient='records')

    file_path_predict = f"routes/dropout/data/predict_{year}.json"
    with open(file_path_predict, 'w') as f:
        f.write(data_mahasiswa_json)

    return jsonify(data_mahasiswa_json = data_mahasiswa_json)


# GET DATA FROM API
def get_histori_mahasiswa(url):
    current_year = datetime.now().year
    dfs = []
    for year in range(current_year, 2014, -1):
        for periode in [2, 1]:
            id_periode = f'{year}{periode}'
            parameter_list = {'id_periode': id_periode}
            response = requests.post(url, headers=headers, data=parameter_list,
verify=False)
            if response.status_code == 200:
                data = response.json()["data"]
                df = pd.DataFrame(data)
                dfs.append(df)
            else:
                print("Error:", response.status_code)

    data_mhs = pd.concat(dfs, ignore_index=True)
    return data_mhs
```

```python
def get_list_mahasiswa(url):
    current_year = datetime.now().year
    dfs = []

    for year in range(current_year, 2014, -1):
        angkatan = f'{year}'
        parameter_list = {'angkatan': angkatan}
        response = requests.post(url, headers=headers, data=parameter_list, verify=False)
        if response.status_code == 200:
            data = response.json()["data"]
            df = pd.DataFrame(data)
            dfs.append(df)
        else:
            print("Error:", response.status_code)

    data_ipk = pd.concat(dfs, ignore_index=True)
    return data_ipk

#PREDICT
# MODEL PREDICT
def predict_model(KNN_model_fix):
    with open(KNN_model_fix, 'rb') as f:
        load_model = pickle.load(f)
    return load_model

def check_file_predict(year):
    file_predict = f"routes/dropout/data/predict_{year}.json"
    return os.path.exists(file_predict)

@dropout_bp.route('/predict/<year>')
def model_predict(year):
    file_predict = f"routes/dropout/data/predict_{year}.json"
    with open(file_predict, 'r') as file:
        df = json.load(file)
```

```
    data_predict= pd.DataFrame(df)
    data_prediction = data_predict[['nim','nama_mahasiswa', 'status_mahasiswa_terakhir',
'masa_studi', 'sks', 'ipk', 'tahun_angkatan', 'nama_program_studi']]
    data_prediction['ipk'] = data_prediction['ipk'].astype(float)

    model = predict_model(f"routes/dropout/data/KNN_model_fix.pkl")
    sc = StandardScaler()
    data_predict = data_prediction[['masa_studi', 'sks', 'ipk']]
    sc.fit(data_predict)
    new_data_scaled = sc.transform(data_predict)
    predicted_class = model.predict(new_data_scaled)
    data_prediction['status_mahasiswa'] = predicted_class

    data_prediction['status_mahasiswa'] =
data_prediction['status_mahasiswa'].apply(chart_predict)
    data_prediction =
data_prediction.loc[data_prediction['status_mahasiswa_terakhir']=='AKTIF']

    data_prediction_json = data_prediction.to_json(orient='records')
    file_path_predict = f"routes/dropout/data/prediction_{year}.json"
    with open(file_path_predict, 'w') as f:
        f.write(data_prediction_json)

    return jsonify(data_prediction_json = data_prediction_json)

def chart_predict(status):
    if status == 1:
        return "Tidak Terprediksi Dropout"
    else:
        return "Terprediksi Dropout"

def preprocess_data(data_mahasiswa):
    # Create DataFrame sks
    data_mahasiswa['sks_semester'] = data_mahasiswa['sks_semester'].astype(np.int64)
    sks = data_mahasiswa.groupby(['nim'])['sks_semester'].sum().reset_index()
```

```
    sks.columns = ['nim','sks']
    sks
    # Merge DataFrame sks & ipk
    sks_with_ipk = pd.merge(sks, data_mahasiswa[['nim', 'nama_mahasiswa',
                                                 'status_mahasiswa_terakhir', 'ipk', 'tahun_angkatan',
                                                 'nama_program_studi']], on='nim', how='inner')
    sks_with_ipk
    # Create Masa Studi Attribute
    # Initial digits
    first_nim = data_mahasiswa.groupby('nim')['id_periode'].min().reset_index()
    first_nim.columns = ['nim', 'first_id_periode']
    # Occurrences of ID periods
    id_periode_count =
data_mahasiswa.groupby(['nim']).size().reset_index(name='id_periode_count')
    merged_df = pd.merge(data_mahasiswa, first_nim, on='nim', how='left')
    merged_df['tahun_masuk'] =
merged_df['first_id_periode'].astype(str).str[:4].astype(int)
    # Counting cuti & non-aktif
    inactive_status = merged_df[merged_df[
        'status_mahasiswa_semester'].isin(['cuti','non-aktif'])].groupby(
            'nim').size().reset_index(name='inactive_count')

    merged_df = pd.merge(merged_df, inactive_status, on='nim', how='left')
    merged_df['inactive_count'].fillna(0, inplace=True)

    merged_df = pd.merge(merged_df, id_periode_count, on='nim', how='left')
    merged_df['id_periode_count'].fillna(0, inplace=True)

    merged_df['masa_studi'] = merged_df['id_periode_count'] -
merged_df['inactive_count']
    merged_df[['nim', 'masa_studi']]

    # Merge DataFrame sks_with_ipk & merged_df by nim
    merged_df_all = pd.merge(sks_with_ipk, merged_df[['nim', 'masa_studi']], on='nim',
how='inner')
```

```
    df_mahasiswa = merged_df_all.drop_duplicates(subset=['nim'], ignore_index=True)

    return df_mahasiswa

# Presentase predict
@dropout_bp.route('/presentase_predict/<year>')
def presentase_predict(year):
    file_predict = f"routes/dropout/data/prediction_{year}.json"
    with open(file_predict, 'r') as file:
        df = pd.read_json(file)

    dropout_count = (df['status_mahasiswa'] == "Terprediksi Dropout").sum()
    non_dropout_count = (df['status_mahasiswa'] == "Tidak Terprediksi Dropout").sum()
    total_count = len(df)

    dropout_percentage = (dropout_count / total_count) * 100
    non_dropout_percentage = (non_dropout_count / total_count) * 100

    dropout_count = int(dropout_count)
    non_dropout_count = int(non_dropout_count)
    total_count = int(total_count)

    output = {
        "terprediksi_dropout": {
            "jumlah": dropout_count,
            "persentase": dropout_percentage
        },
        "tidak_terprediksi_dropout": {
            "jumlah": non_dropout_count,
            "persentase": non_dropout_percentage
        }
    }
    file_presentase_predict_mahasiswa =
f"routes/dropout/data/file_presentase_predict_mahasiswa_{year}.json"
    with open(file_presentase_predict_mahasiswa, 'w') as f:
```

```
        json.dump(output, f, indent=4)

    return jsonify(output)


#OTHERS VISUALIZATION
@dropout_bp.route('/get_histori_mahasiswa/<year>/', methods=['GET'])
def get_data_histori_mahasiswa(year):
    histori_mahasiswa = get_histori_mahasiswa(‘ ‘)

    data_s1_only = histori_mahasiswa[histori_mahasiswa['program_pendidikan'] == 'S1']

    data_s1_only_json = data_s1_only.to_json(orient='records')
    file_path_data_mahasiswa = f"routes/dropout/data/file_data_mahasiswa_{year}.json"
    with open(file_path_data_mahasiswa, 'w') as f:
        f.write(data_s1_only_json)

    return jsonify(data_s1_only_json = data_s1_only_json)

@dropout_bp.route('get_list_mahasiswa/<year>', methods=['GET'])
def get_data_list_mahasiswa(year):
    list_mahasiswa = get_list_mahasiswa(‘ ‘)

    list_mahasiswa_json = list_mahasiswa.to_json(orient='records')
    file_path_list_data_mahasiswa =
f"routes/dropout/data/file_list_data_mahasiswa_{year}.json"
    with open(file_path_list_data_mahasiswa, 'w') as f:
        f.write(list_mahasiswa_json)

    return jsonify(list_mahasiswa_json=list_mahasiswa_json)

@dropout_bp.route('get_merged_mahasiswa/<year>', methods=['GET'])
def get_merged_mahasiswa(year):
    histori_mahasiswa = get_histori_mahasiswa(‘ ‘)
    list_mahasiswa = get_list_mahasiswa(‘ ‘)
```

```
    data_s1_only = histori_mahasiswa[histori_mahasiswa['program_pendidikan'] == 'S1']

    merged_data = pd.merge(data_s1_only.drop(columns=['ips']), list_mahasiswa,
on='nim', how='inner')

    merged_data_json = merged_data.to_json(orient='records')

    file_path_merged = f"routes/dropout/data/merged_data_{year}.json"
    with open(file_path_merged, 'w') as f:
        f.write(merged_data_json)

    return jsonify(merged_data_json = merged_data_json)

def check_file_data_mahasiswa(year):
    file_data_mahasiswa = f"routes/dropout/data/file_data_mahasiswa_{year}.json"
    return os.path.exists(file_data_mahasiswa)

def check_file_list_data_mahasiswa(year):
    file_list_data_mahasiswa =
f"routes/dropout/data/file_list_data_mahasiswa_{year}.json"
    return os.path.exists(file_list_data_mahasiswa)

def check_file_merged_data(year):
    file_merged_data_mahasiswa = f"routes/dropout/data/merged_data_{year}.json"
    return os.path.exists(file_merged_data_mahasiswa)

# PRESENTASE STATUS MAHASISWA
@dropout_bp.route('/persentase_status_mahasiswa/<year>')
def persentase_mahasiswa(year):
    file_data_mahasiswa = f"routes/dropout/data/file_data_mahasiswa_{year}.json"
    with open(file_data_mahasiswa, 'r') as file:
        data = json.load(file)

    total_mahasiswa = len(data)
```

```
    status_counts = {}

    for mahasiswa in data:
        status = mahasiswa['status_mahasiswa_semester'].strip().upper()
        if status in status_counts:
            status_counts[status] += 1
        else:
            status_counts[status] = 1

    persentase_status_mahasiswa = [
        {
            "status": status,
            "jumlah": count,
            "persentase": round((count / total_mahasiswa) * 100, 2)
        }
        for status, count in status_counts.items()
    ]

    file_path_persentase_mahasiswa =
f"routes/dropout/data/file_presentase_mahasiswa_{year}.json"
    with open(file_path_persentase_mahasiswa, 'w') as f:
        json.dump(persentase_status_mahasiswa, f)
    return jsonify(persentase=persentase_status_mahasiswa)

# DROPOUT YEARLY
@dropout_bp.route('/jumlah_dropout_yearly/<year>', methods=['GET'])
def jumlah_dropout_yearly(year):
        file_merged_data_mahasiswa = f"routes/dropout/data/merged_data_{year}.json"
        with open(file_merged_data_mahasiswa, 'r') as file:
            data = json.load(file)

        jumlah_dropout_per_tahun = {}

        for mahasiswa in data:
            tahun_angkatan = mahasiswa.get('tahun_angkatan', None)
```

```
            if not tahun_angkatan:
                continue

            status = mahasiswa.get('status_mahasiswa_semester', '').strip().upper()
            if status != "DROP-OUT/PUTUS STUDI":
                continue

            if tahun_angkatan in jumlah_dropout_per_tahun:
                jumlah_dropout_per_tahun[tahun_angkatan] += 1
            else:
                jumlah_dropout_per_tahun[tahun_angkatan] = 1

        output_file_path = f"routes/dropout/data/jumlah_dropout_per_tahun_{year}.json"
        with open(output_file_path, 'w') as output_file:
            json.dump(jumlah_dropout_per_tahun, output_file, indent=4)
        return jsonify(jumlah_dropout_per_tahun=jumlah_dropout_per_tahun)



@dropout_bp.route('/prodi_chart/<year>')
def prodi_dropout_count(year):
    file_merged_data_mahasiswa = f"routes/dropout/data/merged_data_{year}.json"
    with open(file_merged_data_mahasiswa, 'r') as file:
        data = json.load(file)

    prodi_dropout_count = {}
    for entry in data:
        prodi = entry['program_studi']
        status = entry['status_mahasiswa_semester']
        if status == 'DROP-OUT/PUTUS STUDI':
            if prodi in prodi_dropout_count:
                prodi_dropout_count[prodi] += 1
            else:
                prodi_dropout_count[prodi] = 1

    chart_data = {
```

```
        'program_studi': list(prodi_dropout_count.keys()),
        'jumlah_dropout': list(prodi_dropout_count.values())
    }
    file_chart_json = f"routes/dropout/data/prodi_dropout_count_{year}.json"
    with open(file_chart_json, 'w') as output_file:
        json.dump(chart_data, output_file, indent=4)
    return jsonify(chart_data)
```

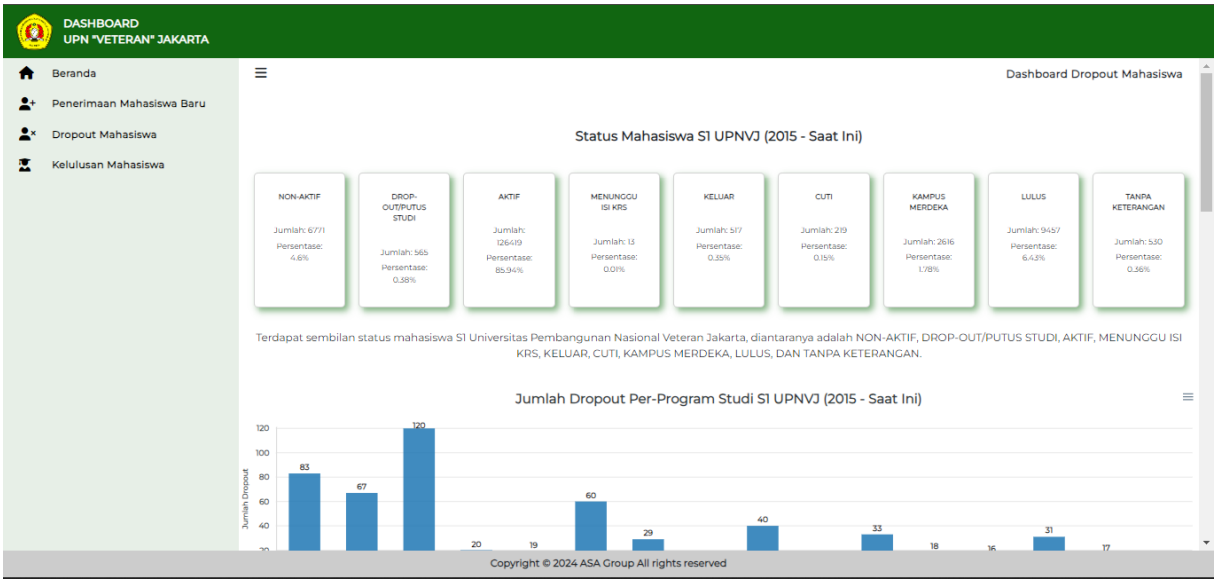
7. *Flask*

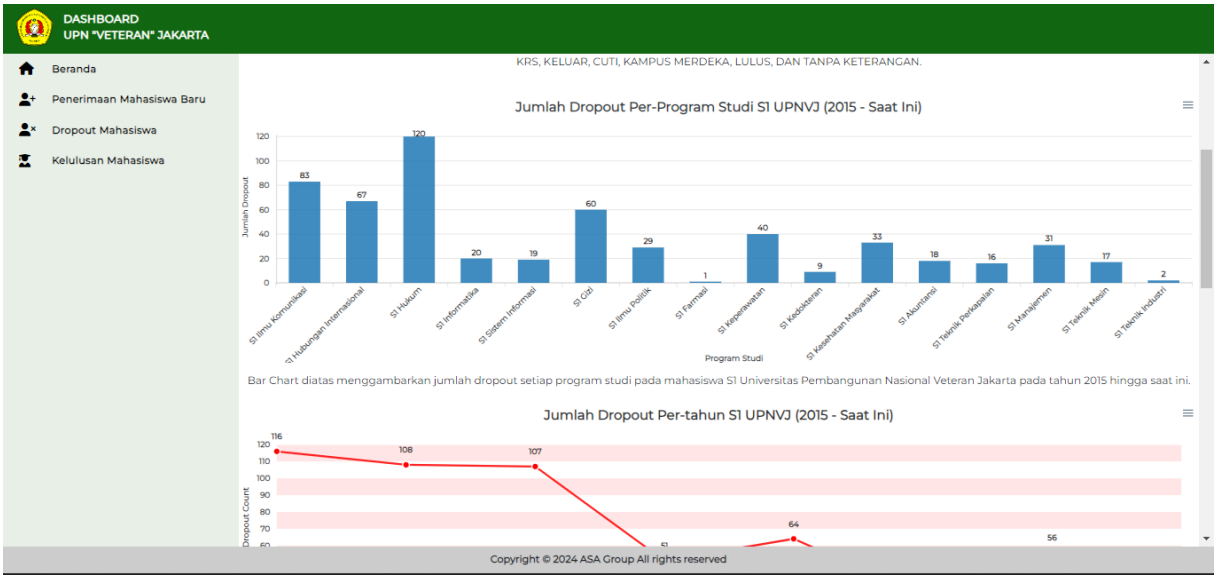
```
import pickle
import numpy as np
import pandas as pd
from flask import Flask, render_template
from routes.penerimaan.routes import penerimaan_bp
from routes.dropout.routes import dropout_bp
from routes.kelulusan.routes import kelulusan_bp

# model = pickle.load(open('model_decisiontree.pkl', 'rb'))


# == == == == == == == == == = FLASK == == == == == == == == == #
app = Flask(__name__)
app.static_folder = 'static'
app.register_blueprint(penerimaan_bp)
app.register_blueprint(dropout_bp)
app.register_blueprint(kelulusan_bp)

@app.route('/')
def index():
    return render_template(f'index.html')

# endpoint penerimaanmahasiswa
if __name__ == '__main__':
    app.run(port=9090, debug=True)
```

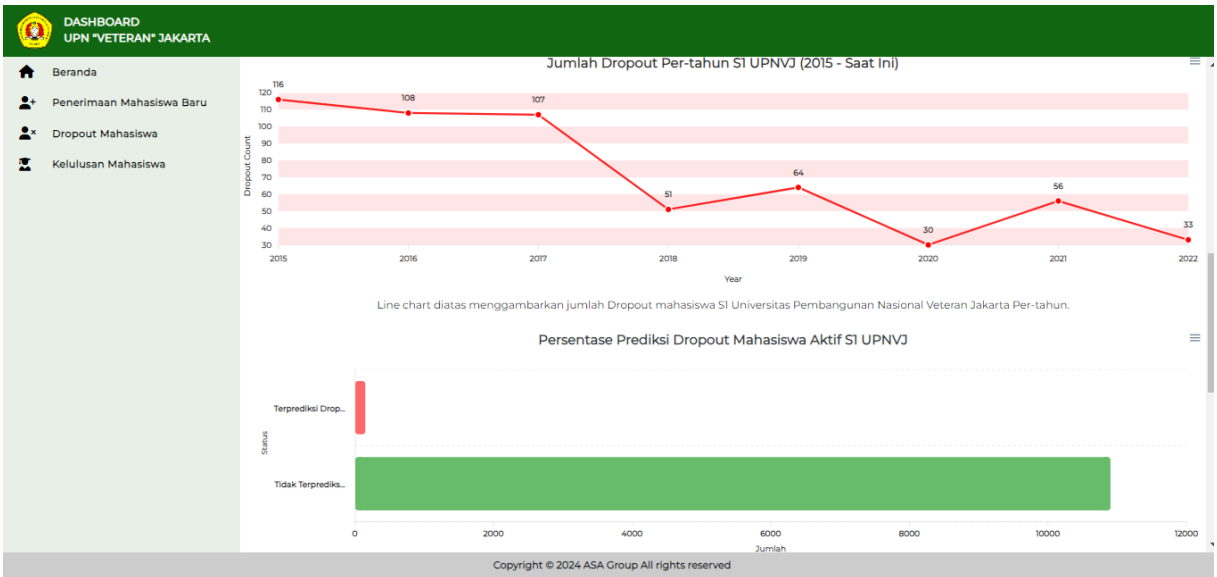
Lampiran 9. Hasil Pemodelan

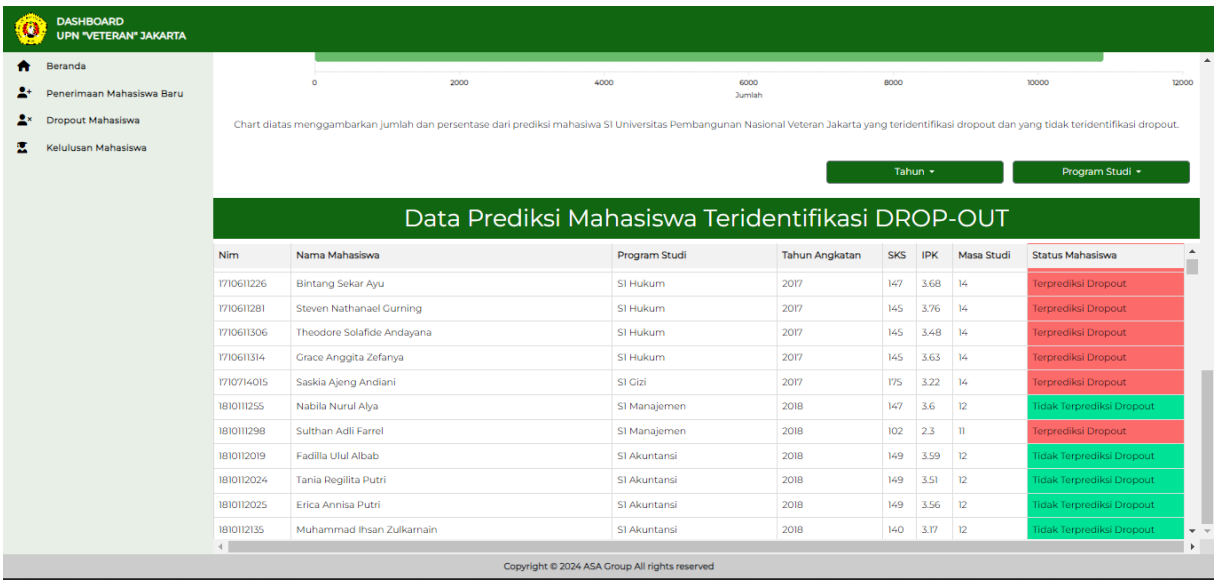
Algoritma	Data	Evaluasi Model
KNN	*Training*	*Accuracy*: 0.9978
		Precision: 0.9980
		Recall: 0.9995
		F1-Score: 0.9988
	Testing	*Accuracy*: 0.9978
		Precision: 0.9985
		Recall: 0.9990
		F1-Score: 0.9988
Naïve Bayes	*Training*	*Accuracy*: 0.9658
		Precision: 0.9734
		Recall: 0.9840
		F1-Score: 0.9787
	Testing	*Accuracy*: 0.9633
		Precision: 0.9750
		Recall: 0.9849
		F1-Score: 0.9799

Lampiran 10. Sistem Dashboard Analytic








DASHBOARD
UPN "VETERAN" JAKARTA
Beranda
Penerimaan Mahasiswa Baru
Dropout Mahasiswa
Kelulusan Mahasiswa
0
2000
4000
6000
8000
10000
12000
Jumlah
Chart diatas menggambarkan jumlah dan persentase dari prediksi mahasiswa SI Universitas Pembangunan Nasional Veteran Jakarta yang teridentifikasi dropout dan yang tidak teridentifikasi dropout.
Tahun
Program Studi
Data Prediksi Mahasiswa Teridentifikasi DROP-OUT
Nim
Nama Mahasiswa
Program Studi
Tahun Angkatan
SKS
IPK
Masa Studi
Status Mahasiswa
1710611226 Bintang Sekar Ayu S1 Hukum 2017 147 3.68 14 Terprediksi Dropout
1710611281 Steven Nathanael Gurning S1 Hukum 2017 145 3.76 14 Terprediksi Dropout
1710611306 Theodore Solafide Andayana S1 Hukum 2017 145 3.48 14 Terprediksi Dropout
1710611314 Grace Anggita Zefanya S1 Hukum 2017 145 3.63 14 Terprediksi Dropout
1710714015 Saskia Ajeng Andiani S1 Gizi 2017 175 3.22 14 Terprediksi Dropout
1810111255 Nabila Nurul Alya S1 Manajemen 2018 147 3.6 12 Tidak Terprediksi Dropout
1810111298 Sulthan Adli Farrel S1 Manajemen 2018 102 2.3 11 Terprediksi Dropout
1810112019 Fadilla Ulul Albab S1 Akuntansi 2018 149 3.59 12 Tidak Terprediksi Dropout
1810112024 Tania Regilita Putri S1 Akuntansi 2018 149 3.51 12 Tidak Terprediksi Dropout
1810112025 Erica Annisa Putri S1 Akuntansi 2018 149 3.56 12 Tidak Terprediksi Dropout
1810112135 Muhammad Ihsan Zulkarnain S1 Akuntansi 2018 140 3.17 12 Tidak Terprediksi Dropout
Copyright © 2024 ASA Group All rights reserved