1. Membuat requirement.txt

requirements.txt
```
flask
Jinja2
pandas
scikit-learn==1.0.2
```

2. Membuat virtual environment dan run requirements.txt

```
PS C:\Users\User\OneDrive\Documents\Digital Skola\Project 7> .venv\Scripts\activate
(.venv) PS C:\Users\User\OneDrive\Documents\Digital Skola\Project 7> pip install -r requirements.txt
```

3. Membuat file app.py

app.py > ...
```python
from flask import Flask
from flask import request
from flask import jsonify
import pandas as pd
from modules.insurance_model import InsuranceModel

app = Flask(__name__)

@app.route('/')
def home():
    return "Welcome to API Modeling Health Insurance"

@app.route('/predict', methods=['POST'])
def predict():
    data = request.get_json()
    df = pd.DataFrame(data)
    result_predict = InsuranceModel().runModel(df, typed='single')
    return jsonify({
        "status" : "predicted",
        "predicted_result": result_predict
    })

if __name__ == "__main__":
    app.run(port=8000)
```

4. Membuat folder modules dan di dalamnya file insurance_model.py dan insurance_pre.py

Insurance_model.py

modules > insurance_model.py > InsuranceModel

```python
from modules.insurance_pre import InsurancePre

import os
import pickle
import time

import pandas as pd
import numpy as np

import warnings
warnings.filterwarnings('ignore')

class InsuranceModel():
    def __init__(self):
        pass

    def runModel(self, data, typed='multi'):
        path = os.getcwd()+"/"+"packages"+"/"
        # path = os.getcwd()+"/modules/packages/"
        model = pickle.load(open(path + 'model_InsuranceRecommendation.pkl', 'rb'))
        col_p = pickle.load(open(path + 'columnPreparation.pkl', 'rb'))
        col_m = pickle.load(open(path + 'columnModelling.pkl', 'rb'))

        X = data[col_p]
        colEncoder, colpOneHotEncoder, colStandarScaler = InsurancePre().colPreparation()
        for col in X.columns:
            prep = pickle.load(open(path + 'prep' + col + '.pkl', 'rb'))
            if col in colpOneHotEncoder:
                dfTemp = pd.DataFrame(prep.transform(X[[col]]).toarray())
                X = pd.concat([X.drop(col, axis=1), dfTemp], axis=1)
            else:
                dfTemp = pd.DataFrame(prep.transform(X[[col]]))
                X = pd.concat([X.drop(col, axis=1), dfTemp], axis=1)
        X.columns = col_m

        if typed == 'multi':
            y = model.predict(X)
            return y

        elif typed == 'single':
            y = model.predict(X)[0]
            if y == 0:
                return 0
            else:
                return 1
        else:
            return False
```

Insurance_pre.py

```
modules > insurance_pre.py > InsurancePre
1   class InsurancePre():
2       def __init__(self):
3           pass
4
5       def colPreparation(self):
6           labelEncoder = ['Gender','Driving_License','Previously_Insured','Vehicle_Damage']
7           oneHotEncoder = ['Vehicle_Age','Region_Code','Policy_Sales_Channel']
8           scallingStandar = ['Age','Annual_Premium','Vintage']
9
10          return labelEncoder, oneHotEncoder, scallingStandar
```

5. Run app

```
(.venv) PS C:\Users\User\OneDrive\Documents\Digital Skola\Project 7> flask run --port 8000
 * Serving Flask app 'app.py'
 * Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:8000
Press CTRL+C to quit
127.0.0.1 - - [30/Jan/2024 22:37:39] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [30/Jan/2024 22:38:23] "POST /predict HTTP/1.1" 200 -
```