BUILD THE PYTHON FLASK APP

|  |  |
| --- | --- |
| **Team ID** | **PNT2022TMID42057** |
| **Project Name** | **Car Resale value Prediction** |

**BUILD THE PYTHON FLASK APP**

import pandas as pd import numpy as np

from flask import Flask,render\_template,Response,request import pickle

from sklearn.preprocessing import LabelEncoder import pickle

app=Flask( name ,template\_folder='../IBM') filename = 'resale\_model.sav'

model\_rand = pickle.load(open(filename,'rb'))

@app.route('/') def index():

return render\_template('index.html')

@app.route('/resaleintro.html') def p():

return render\_template('resaleintro.html')

@app.route('/predict') def predict():

return render\_template('resalepredict.html')

@app.route('/y\_predict',methods=['GET','POST']) def y\_predict():

regyear = int(request.form['regyear']) powerps = float(request.form['powerps']) kms = float(request.form['kms'])

regmonth = int(request.form.get('regmonth')) gearbox = request.form['gearbox']

damage = request.form['dam']

model = request.form.get('model\_type') brand = request.form.get('brand') fuelType = request.form.get('fuel')

vehicletype= request.form.get('vehicletype') new\_row =

{'yearOfRegistration':regyear,'powerPS':powerps,'kilometer':kms,'monthOfRegistration':regm onth,'gearbox':gearbox,'notRepairedDamage':damage,'model':model,'brand':brand,'fuelType':f uelType,'vehicleType':vehicletype}

print(new\_row)

new\_df = pd.DataFrame(columns=['vehicleType','yearOfRegistration','gearbox','powerPS','model','kilo meter','monthOfRegistration','fuelType','brand','notRepairedDamage'])

new\_df = new\_df.append(new\_row,ignore\_index=True)

labels = ['gearbox','notRepairedDamage','model','brand','fuelType','vehicleType'] mapper = {}

for i in labels:

mapper[i] = LabelEncoder()

mapper[i].classes\_ = np.load(str('classes'+i+'.npy'),allow\_pickle=True) tr = mapper[i].fit\_transform(new\_df[i])

new\_df.loc[:,i+'\_Labels'] = pd.Series(tr,index=new\_df.index)

labeled = new\_df[ ['yearOfRegistration','powerPS','kilometer','monthOfRegistration'] + [x+"\_Labels" for x in labels]]

X = labeled.values print(X)

y\_prediction = model\_rand.predict(X) print(y\_prediction)

return render\_template('resalepredict.html',ypred="{:.2f}".format(y\_prediction[0]))

if name == ' main ': app.run(host='Localhost',debug=True,threaded=False)

