Application Building:

Build The Python Flask App

#Importing required libraries

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

#Load the model and initialize Flask app

app=Flask(\_\_name\_\_) filename=’resale\_model.sav’ model\_rand=pickle.load(open(filename,’rb’))

#Configure app.py to fetch the parameter values from the ui,and return the prediction

@app.route(‘/’) def index():

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('modeltype') brand= request.form.get('brand')

fuelType = request.form.get('fuel') vehicletype= request.form.get('vehicletype')

new\_row("yearOfRegistration':regyear, 'powerPS':powerps, 'kilometer':kms,

monthofRegistration': regmonth, gearbox gearbox, 'notRepairedDamage': damage,

'model':model, 'brand':brand, 'fuelType': fuelType,

'vehicleType': vehicletype)

print(new row)

new\_df = pd.DataFrame(columns =['vehicleType', 'yearOfRegistration', 'gearbox", 'powerPS', 'model', 'kilometer', 'monthofRegistration', 'fuelTypek, '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'))

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(‘resalespredict.html’,ypred = ‘The resale value predicted is {:.2f}$’.format(y\_prediction[0]))

Run the app

If \_\_name\_\_ == ‘\_\_main’:

app.run(host=’localhost’,debug = True, threaded = False)