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)
```