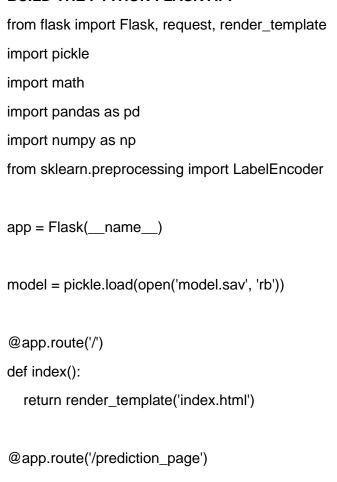
## **BUILD THE PYTHON FLASK APP**

Team ID	PNT2022TMID12591
Project Name	Car Resale Value Prediction

## **BUILD THE PYTHON FLASK APP**



```
def prediction_page():
  return render_template('form.html')
@app.route('/prediction', methods=['POST'])
def prediction():
  # regyear = request.form.get('registrationYear')
  # powerps = request.form.get('powerOfCarInPS')
  # kms = request.form.get('KmTheCarAsDriven')
  # regmonth = request.form.get('registrationMonth')
  # gearbox = request.form.get('gearbox')
  # damage = request.form.get('damage')
  # model_type = request.form.get('modelType')
  # brand = request.form.get('brandOfTheCar')
  # fuel_type = request.form.get('fuefuelTypeOfTheCar')
  # vehicle_type = request.form.get('vehicleType')
  # print(regyear, powerps, kms, regmonth, gearbox, damage, model_type, brand, fuel_type, vehicle_type)
  regyear = int(request.form.get('registrationYear'))
  powerps = float(request.form.get('powerOfCarInPS'))
  kms = float(request.form.get('KmTheCarAsDriven'))
  regmonth = int(request.form.get('registrationMonth'))
```

```
gearbox = request.form.get('gearbox')
damage = request.form.get('damage')
model_type = request.form.get('modelType')
brand = request.form.get('brandOfTheCar')
fuel_type = request.form.get('fuelTypeOfTheCar')
vehicle_type = request.form.get('vehicleType')
print(regyear, powerps, kms, regmonth, gearbox, damage, model_type, brand, fuel_type, vehicle_type)
new_row = {
  'vehicleType' : vehicle_type,
  'yearOfRegistration': regyear,
  'gearbox' : gearbox,
  'powerPS': powerps,
  'model' : model_type,
  'kilometer': kms,
  'monthOfRegistration': regmonth,
  'fuelType' : fuel_type,
  'brand': brand,
  'notRepairedDamage' : damage
print(new_row)
```

```
new_df = pd.DataFrame(columns=['vehicleType', 'yearOfRegistration', 'gearbox', 'powerPS', 'model', 'kilometer', 'monthOfRegistration', 'fuelType',
'brand', 'notRepairedDamage'])
  new_df = new_df.append(new_row, ignore_index=True)
  labels = ['vehicleType', 'gearbox', 'model', 'fuelType', 'brand', 'notRepairedDamage']
  mapper = \{\}
  for i in labels:
     mapper[i] = LabelEncoder()
     mapper[i].classes_ = np.load(str('numpy_classes/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',
     'kilometer',
     'monthOfRegistration',
     'powerPS'
  + [x + '_labels' for x in labels]]
  x = labeled.values
  print(x)
```

```
result = model.predict(x)[0]
result = math.ceil(result)
result = '$' + str(result)
print('The predicted result: ', result)

return render_template('form.html', pred_result=result)

if __name__ == '__main__':
    app.run(debug=True)
```