IBM NALAIYA THIRAN

PROJECT REPORT

K.L.N COLLEGE OF ENGINEERING, POTTAPALAYAM

(An Autonomous institution, affiliated to Anna university, Chennai)

Problem statement : Car Resale value Prediction

Team ID : PNT2022TMID11620

Team Leader : T.Venkatesh

Team members : 1. E.Trinity

2. B.Sharukesh

3. T.H.Vinod

4. R.Sutharsan

Faculty mentor : G.H.Ram Ganesh

Evaluator : Dr.J.S. Kanchana

Industry mentor : Prof Swetha

INDEX

1.Introduction:

It is expected that sales of old cars and second-hand imported (reconditioned) autos will rise in tough economic times. Leasing a car rather than purchasing one entirely is typical in many affluent nations. After the lease term is up, the buyer will have the option of purchasing the vehicle for its residual value, or anticipated resale value. Therefore, being able to accurately predict the salvage value (residual value) of cars is in the best interest of sellers and financiers from a business standpoint.

We suggested an intelligent, adaptable, and efficient method that is based on applying regression algorithms to forecast the resale value of the vehicle. A regression model needs to be constructed that would provide the vehicle's closest resale value, taking into account the key variables that would impact this value. The algorithm with the highest accuracy will be chosen from among the many regression algorithms we use, and it will then be integrated into the web-based application that notifies the user of the status of his product.

1.1 Project overview:

The primary goal of this research is to create a model that can forecast the value of a used car. We will first locate and download the dataset from a variety of sites, including Kaggle, UCI, etc.

Since the majority of automobiles sold worldwide are produced by the German auto industry, we have acquired the autos.csv file, a German dataset.

Then, to make the data easier to understand and manipulate, we pre-processed the data and translated many German data into English.

Then, as we will be obtaining numerical data as the output, we employ the regression approach. We also use the Random Forest Regression method.

We used the Random Forest Regression method because

- It provides an effective way of handling missing data.
- It can produce a reasonable prediction without hyper-parameter tuning.
- It solves the issue of overfitting in decision trees.
- In every random forest tree, a subset of features is selected randomly at the node’s splitting point.

We then uploaded our machine learning model to the IBM cloud, constructed a scoring API with Flask, and built the front end using HTML, CSS, and Bootstrap..

1.2 Purpose

The major goal of this study is to determine whether it would be profitable for financiers or sellers to be able to accurately anticipate a car's residual value. We make advantage of features like fuelType, yearOfRegistration, powerPS, kilometres, and gearbox.

We communicate with users through a web-based application created with Flask, and the anticipated value is also shown on the website itself.

The user won't need to worry about their data being private because there are no login or logout choices and the user data is just saved in the browser caches.

2.Literature survey

In a literary survey, we analyse critically, and concisely earlier research and literature related to a particular research problem and utilize them for their own research purposes. It helps us in understanding the significance of new research and its connections to earlier work.

2.1 Existing problem

The prices of the car industry are fixed in now a days, so the buying a new car should be assured the worth of money.

There are various websites that provide these services but the method to do so is not the best, different models and systems may contribute to predicting the power of the actual market value of the car.

2.2 References

1. Chuyang Jin, "Price Prediction of Used Cars Using Machine Learning", 2021

This paper aims to build a model to predict used cars' reasonable prices based on multiple aspects, including vehicle mileage, year of manufacturing, fuel consumption, transmission, road tax, fuel type, and engine size. This model can benefit sellers, buyers, and car manufacturers in the used cars market. Upon completion, it can output a relatively accurate price prediction based on the information that user's input. The model building process involves machine learning and data science. The dataset used was scraped from listings of used cars. Various regression methods, including linear regression, polynomial regression, support vector regression, decision tree regression, and random forest regression, were applied in the research to achieve the highest accuracy. Before the actual start of model-building, this project visualized the data to understand the dataset better. The dataset was divided and modified to fit the regression, thus ensure the performance of the regression. To evaluate the performance of each regression, R-square was calculated. Among all regressions in this project, random forest achieved the highest R-square of 0.90416. Compared to previous research, the resulting model includes more aspects of used cars while also having a higher prediction accuracy.

2. **Feng Wang; Xusong Zhang; Qiang Wang, "Prediction of Used Car Price Based on Supervised Learning Algorithm", 2021**

In this paper, we use machine learning algorithms to predict the price of used cars with less human intervention to make the results more objective. The method used is to pre-process the dataset through Python's Pycaret package and compare the performance of each algorithm through the algorithm comparison function, in this study Extra Trees Regressor, Random Forest Regressor performs relatively well. Finally, the algorithm was optimized by using the hyperparameter function. The results show that R2 = 0.9807 obtained from extreme random numbers is the best performance. The algorithm was obtained and validated with new data to derive the final algorithm model. When new used car data flows into the used car system, used car prices will be automatically generated by this algorithm, which will make the workflow of the used car market faster and more competitive for that used car market.

3. **Janke Varshitha; K Jahnavi; C. Lakshmi, "Prediction Of Used Car Prices Using Artificial Neural Networks And Machine Learning" , 2022**

With the extensive growth in usage of cars, the newly produced cars are unable to reach the customers for various reasons like high prices, less availability, financial incapability, and so on. Hence the used car market is escalated across the globe but in India, the used car market is in a very nascent stage and mostly dominated by the unorganized sector. This gives chance for fraud while buying a used car. Hence a high precision model is required which will estimate the price of a used car with none bias towards customer or merchandiser. In this model, A Supervised learning-based Artificial Neural Network model and Random Forest Machine Learning model are developed which can learn from the car dataset provided to it. This project presents a working model for used car price prediction with a low error value. A considerable number of distinct attributes are examined for reliable and accurate predictions. The results obtained agree with theoretical predictions and have shown improvement over models which use simple linear models. An ANN (Artificial Neural Network) is built by using Keras Regression algorithm namely Keras Regressor and other Machine Learning Algorithms Namely Random Forest, Lasso, Ridge, Linear regressions are built. These algorithms are tested with the car dataset. Experimental results have shown that the Random Forest model with a Mean Absolute Error value of 1.0970472 and R2 error value of 0.772584 has given the less error among all the other algorithms. The work presented here has shown profound implications for future studies of Used Cars price Prediction using Random Forest and might one day help to solve the problem of frauds with one hundred percent accuracy.

4. **Mustapha Hankar; Marouane Birjali; Abderrahim Beni-Hssane , "Used Car Price Prediction using Machine Learning: A Case Study", 2022**

In many business fields that are related to statistics and machine learning (ML), multiple linear regression (MLR) models are often used to estimate and fit a linear relationship between a continuous response variable and other explanatory variables. In our case study, we applied several regression techniques based on supervised

machine learning to predict the resale price of used cars given many factors such as mileage, fuel type, fiscal power, mark, model, and the production year of the car. In all tested models, gradient boosting regressor showed a high R-squared score and low root mean square error.

5. **Shengqiang Han; Jianhua Qu; Jinyi Song; Zijing Liu Second-hand , "Car Price Prediction Based on a Mixed-Weighted Regression Model", 2022**

With the development of motor vehicles, the circulation demand of motor vehicles in the form of "second-hand cars" in circulation links is increasing. As a special "e-commerce commodity", second-hand cars are more complicated than ordinary e-commerce commodities. As a result, it is difficult to estimate the price of second-hand cars, which is not only influenced by the basic configuration of the car, but also by the car conditions. At present, the state has not issued a standard to judge the value of second-hand car. To solve this problem, in this paper, first making feature engineering, which includes data pre-processing and feature screening. Data pre-processing includes data cleaning and data transformation, data cleaning includes removing outliers and filling missing values, and data transformation is used to unify data format to improve data quality. The feature screening includes correlation analysis and feature extraction based on LightMBG, and the screened features provide the basis for model building, training and prediction. Then, five regression models are constructed by using the feature attributes obtained by the feature engineering for training, and evaluated. Then, Random Forest and XGBoost are weighted and mixed to get a novel regression model, and the effect of the model is better than that of the five regression models. Finally, the novel regression model is used to predict the price of second-hand cars.

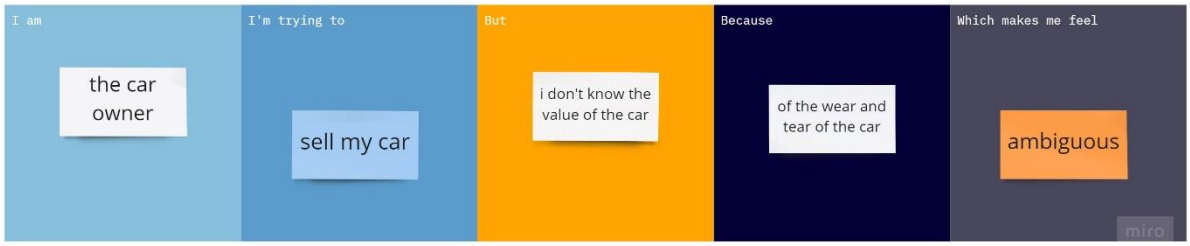
6. **Chejarla Venkat Narayana; Chinta Lakshmi Likhitha; Syed Bademiya; Karre Kusumanjali, "Machine Learning Techniques To Predict The Price Of Used Cars: Predictive Analytics in Retail Business", 2021**

It is generally known that, taking wise and challenging decisions is really a crucial task in every business. Taking improper decisions can cause huge loss and even lead to shutdown of business. To propose a novel solution for this challenge, this research work majorly focuses on one of the retail businesses i.e., used car sales business. The proposed research work shows that, the predictive analytical models will be a great add-on to business mainly for assisting the decision-making process. Predictive Analytics is a process, where the businesses use statistical methods and technologies to analyze their historical data for delivering new insights and plan the future accordingly. The major objective of our paper is to build a prediction model i.e., a fair price mechanism to predict the cars selling price based on their features like the car model, the number of years that a car is old, the type of fuel it uses, the type of seller, the type of transmission and the number of kilometres that the car has driven so far. This paper will help to get an approximation about selling price of a used car based on its features and reduces the seller and consumer risk in business. The proposed model utilizes the machine learning algorithms and regression techniques of statistics like linear, decision tree and random forest regressions to achieve this task.

7. **[Chejarla Venkata Narayana](); [Nukathoti Ooha Gnana Madhuri](); [Atmakuri NagaSindhu](); [Mulupuri Aksha](); [Chalavadi Naveen](), "Second Sale Car Price Prediction using Machine Learning Algorithm", 2022**

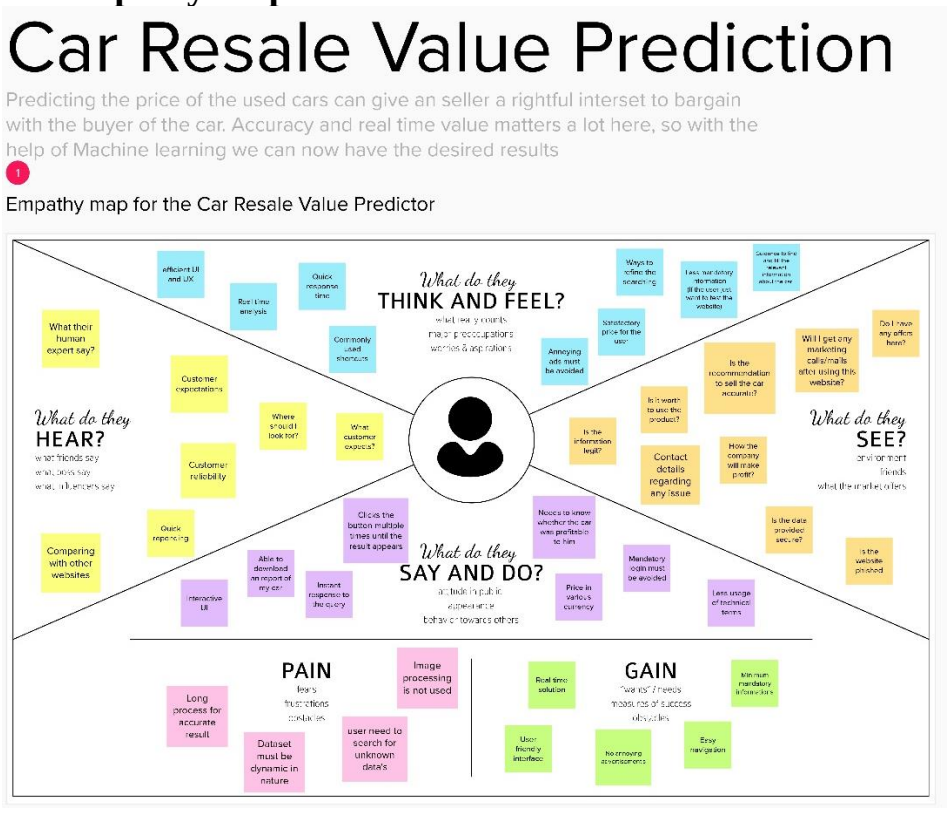
Every business firm recognizes the need of making sound and challenging decisions. Poor decisions can lead to substantial losses and even the demise of a firm. This paper is focused on one of the retail enterprises, which deals with the used car sales. The major goal is to develop a prediction model that can estimate the selling price of used cars based on key factors. Machine learning techniques such as Random Forest Regression, Feature engineering technique such as Extra Trees Regression are employed to accomplish the goal as Random Forest Regression is modelled for prediction analysis and Extra Trees Regression fits the number of decision trees. The results are so encouraging with our approach.

2.3 Problem Statement Definition

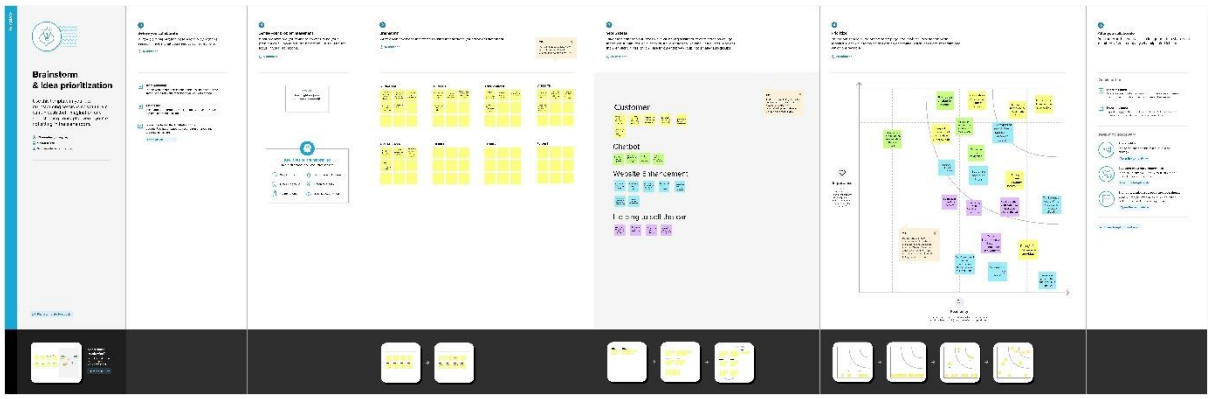


3. IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas



3.2 Ideation & Brainstorming

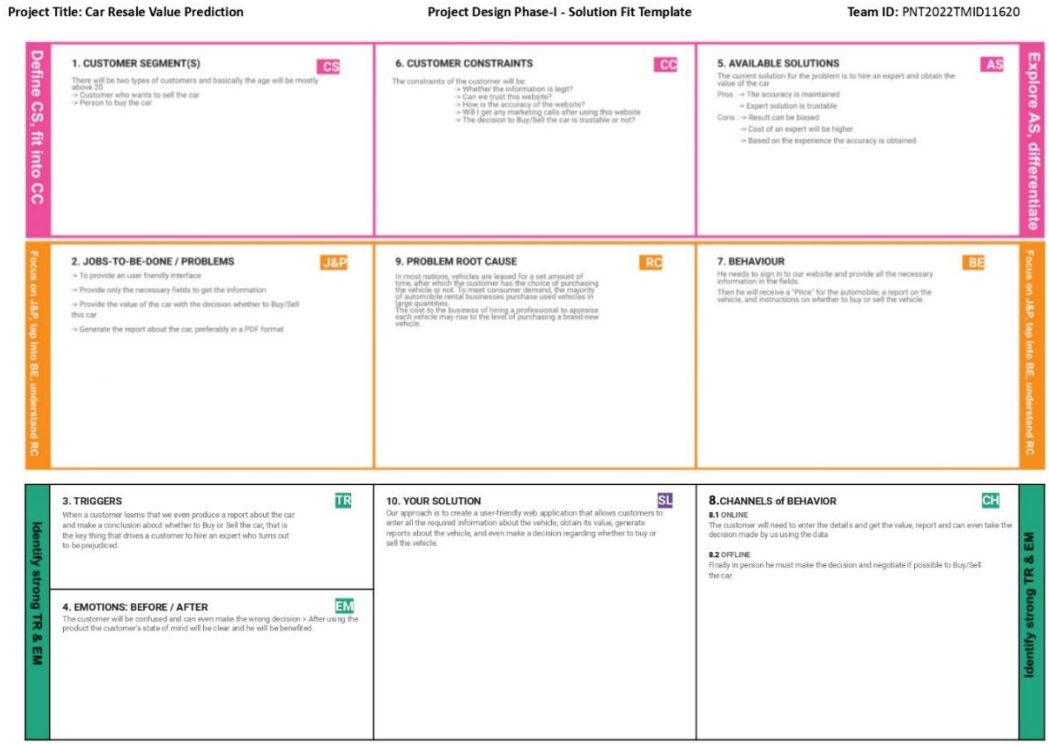


3.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	With difficult economic conditions, it is likely that sales of second-hand imported (reconditioned) cars and used cars will increase. Now a days there are many second-hand car dealers but knowing the price and generating a report about all the cars is very difficult task and can consume a lot of manpower and time. Many companies (mostly rental companies) will buy the second-hand cars in bulk, if they can predict the value and condition of the car, then they can have the upper hand in making profit In many developed countries, it is common to lease a car rather than buying it outright. After the lease period is over, the buyer has the possibility to buy the car at its residual value, i.e., its expected resale value. Thus, it is of commercial interest to sellers/financers to be able to predict the salvage value (residual value) of cars with accuracy.
2.	Idea / Solution description	In order to predict the resale value of the car, we proposed an intelligent, flexible, and effective system that is based on using regression algorithms.

		Considering the main factors which would affect the resale value of a vehicle our model is to be built that would give the nearest (most accurate) resale value of the vehicle. We will be using various regression algorithms and algorithm with the best accuracy will be taken as a solution, then it will be integrated to the web-based application where the user is notified with the status of his product.
3.	Novelty / Uniqueness	We will be generating a report about the car and will also give a suggestion whether to buy/sell the car based on the type of user asking it
4.	Social Impact / Customer Satisfaction	This can save the company from going into bankrupt, typically price of a car (even) will be in a range of 4 to 8 Lakhs, this kind of a huge investment must salvage all the profits it can have. Having a report in hand gives the user an upper hand to make profit from it Giving a suggestion to buy/sell the car can help them in making a rightful decision
5.	Business Model (Revenue Model)	We can have the google advertisements to generate the income. We can also find the cars with the same specs and features from any other dealers for a cheaper price (if available) there for we might get a commission as well.
6.	Scalability of the Solution	We can make this model as an google extension too, so that the price predication for a bulk of cars will be very fast and cost effective. As we are using the IBM cloud, it can support the user load, so that any changes made to the website or any abnormal usage can be managed without the website to fall down.

3.4 Problem Solution fit



4. REQUIREMENT ANALYSIS

4.1 Functional requirement

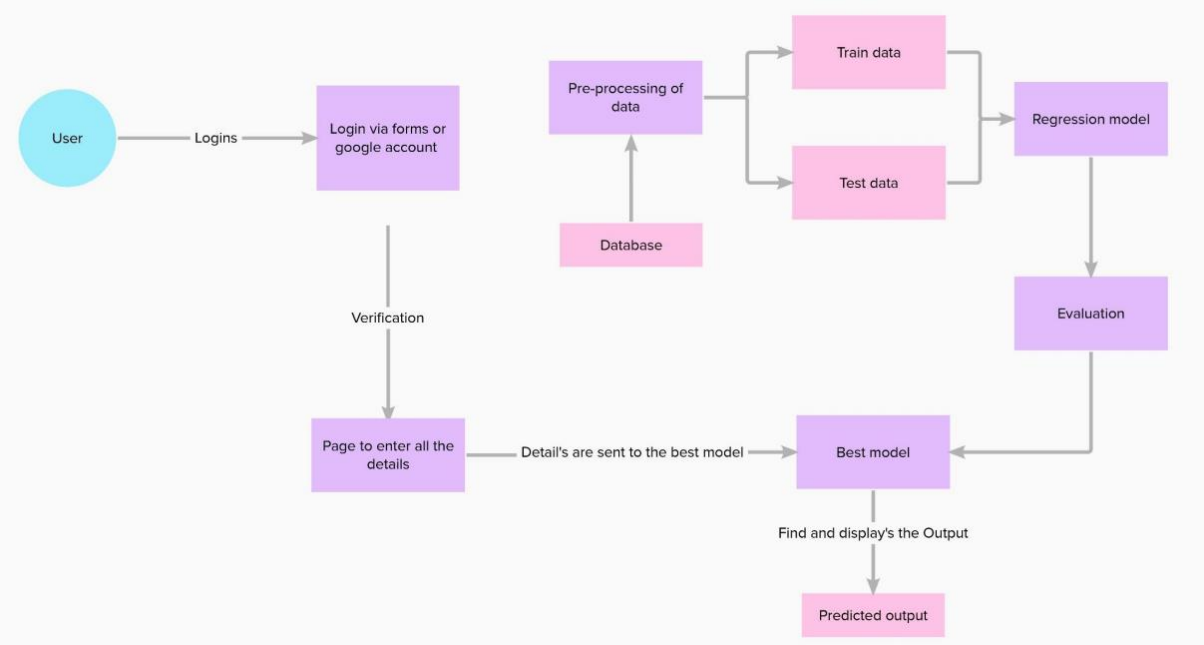
FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail
FR-2	User Confirmation	Confirmation via Email
FR-3	User Vehicle details	• Vehicle Type • Year of manufacture • Fuel Type • Brand • Model • Gearbox (Auto/Manual) • Kilometre driven • Any Damage?
FR-4	Classification whether to buy the vehicle or not? (Optional)	Using the multiple existing features, loading it into an algorithm to find whether the vehicle is worth to buy or not?

4.2 Non-Functional requirements

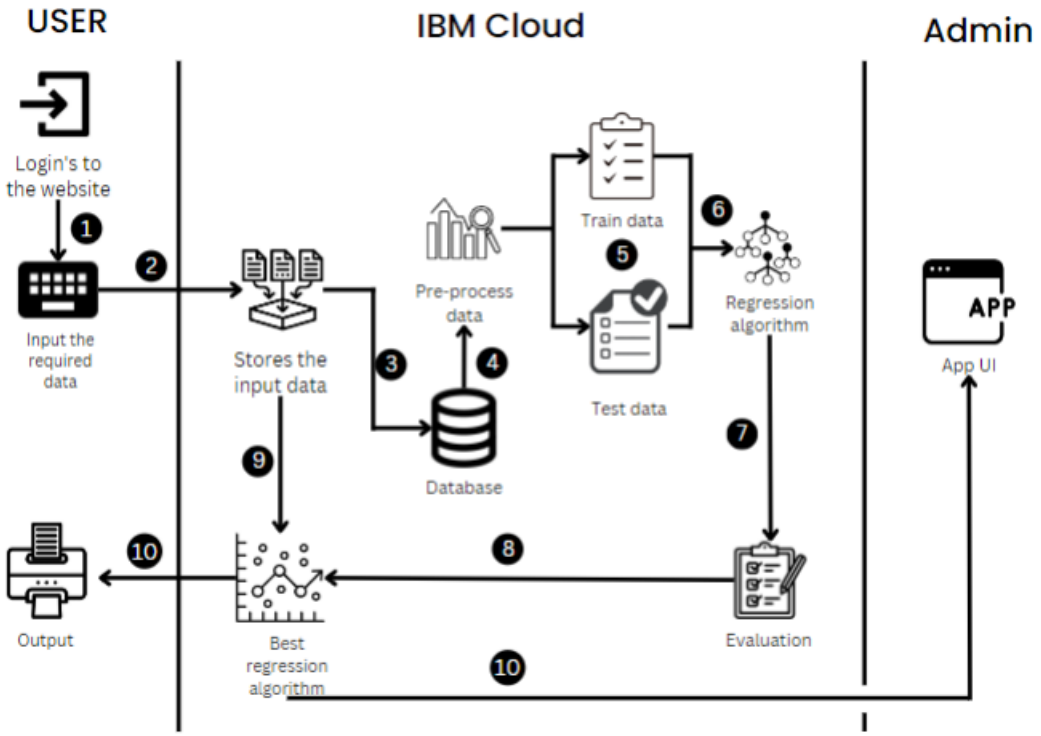
FR No.	Non-Functional Requirement	Description
NFR-1	**Usability**	User interface (UI) is very much user-friendly and easy to use for a seamless flow of the process
NFR-2	**Security**	Must guard against SQL injection and other types of attacks that could lead to data theft.
NFR-3	**Reliability**	A trustworthy source where user information is encrypted and protected from attackers
NFR-4	**Performance**	The user interface needs to be fast and able to handle a significant quantity of network traffic.
NFR-5	**Availability**	The website must not crash because of network load and must always be accessible to users.
NFR-6	**Scalability**	The website must be able to withstand fluctuations in network traffic and resource usage and be error-free.

5. PROJECT DESIGN

5.1 Data Flow Diagrams



5.2 Solution & Technical Architecture



5.3 user stories

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Home page	USN-1	As a user, I can view my home page of the application.	20	Low	Venkatesh, Trinity
Sprint-2	Car resale value display	USN-2	As a user, I will be able to enter the data in the application	20	Medium	Sharukesh, Sutharsan
Sprint-3	Data entry	USN-3	As a user, there will be fields in which I need to give my data	20	High	Vinod, Trinity
Sprint-4	Resale Value Prediction	USN-4	As a user, I will expect my predicted value to be displayed and cloud deployment	20	High	Venkatesh, Vinod

6. PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	08 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022	12 Nov 2022	20	12 Nov 2022
Sprint-4	20	6 Days	14 Nov 2022	19 Nov 2022	20	18 Nov 2022

6.2 Sprint Delivery Schedule

Pre-requites

- Imported the necessary libraries
- Reading the data
- Cleaning the data
- Exporting the pre-processed data
- Creating the .npy files
- Splitting the train and testing data
- Save the ML model
- Create HTML pages
- Integrate with flask
- Deploy the ML model
- Develop the scoring endpoint

6.3 Reports from JIRA

Summary	Issue key	Issue id	Issue Type	Status	Project key	Project name	Project type	Project lead
Train the ML model on IBM	CRVP1607-26	10033	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Integrate flask with scoring end point	CRVP1607-25	10032	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Register for IBM cloud	CRVP1607-24	10031	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Build python flask app	CRVP1607-23	10030	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Execute and test the model	CRVP1607-22	10029	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Build an HTML page	CRVP1607-21	10028	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Save the model	CRVP1607-20	10027	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Check the metric of model	CRVP1607-19	10026	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Choose the appropriate model	CRVP1607-18	10025	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Splitting of data Splitting of data	CRVP1607-17	10024	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Cleaning the dataset	CRVP1607-6	10013	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Read the dataset	CRVP1607-5	10012	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
import required libraries	CRVP1607-4	10011	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Collect dataset	CRVP1607-3	10010	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Project folder structure	CRVP1607-2	10009	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T
Pre-requisite	CRVP1607-1	10008	Story	Done	CRVP1607	Car Resale Value Prediction-1	software	venkatesh T

Project lead id	Priority	Resolution	Assignee	Reporter	Reporter Id	Creator	Creator Id	Sprint	Status Category
637506615fc160544e17c0fd	Medium	Done	venkatesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 4	Done
637506615fc160544e17c0fd	Medium	Done	venkatesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 4	Done
637506615fc160544e17c0fd	Medium	Done	Trinity	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 4	Done
637506615fc160544e17c0fd	Medium	Done	Sharukesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 3	Done
637506615fc160544e17c0fd	Medium	Done	Vinod T H	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 3	Done
637506615fc160544e17c0fd	Medium	Done	Sharukesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 3	Done
637506615fc160544e17c0fd	Medium	Done	Trinity	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 2	Done
637506615fc160544e17c0fd	Medium	Done	venkatesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 2	Done
637506615fc160544e17c0fd	Medium	Done	Vinod T H	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 2	Done
637506615fc160544e17c0fd	Medium	Done	Trinity	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 2	Done
637506615fc160544e17c0fd	Medium	Done	venkatesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 1	Done
637506615fc160544e17c0fd	Medium	Done	venkatesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 1	Done
637506615fc160544e17c0fd	Medium	Done	Vinod T H	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 1	Done
637506615fc160544e17c0fd	Medium	Done	Sharukesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 1	Done
637506615fc160544e17c0fd	Medium	Done	venkatesh	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 1	Done
637506615fc160544e17c0fd	Medium	Done	Trinity	venkatesh T	637506615fc160544e17c0fd	venkatesh T	637506615fc160544e17c0fd	CRVP1607 Sprint 1	Done

7. CODING & SOLUTIONING

7.1 Sprint-1

1. Imported the necessary libraries

```
import pandas as pd
import numpy as np
import matplotlib as plt
from sklearn.preprocessing import LabelEncoder
import pickle
from sklearn.model_selection import cross_val_score, train_test_split
```

2. Reading the data

```
df = pd.read_csv("Data/autos.csv", header=0, sep=',', encoding='Latin1',)
```

3. Cleaning the entire data

```
df[df.seller != 'gewblich']
df=df.drop('seller',1)

print(df.offerType.value_counts())

df[df.offerType != 'Gesuch']
df = df.drop('offerType',1)

print(df.shape)
df=df[(df.powerPS>50)&(df.powerPS<900)]
print(df.shape)

df=df[(df.yearOfRegistration >= 1950)&(df.yearOfRegistration<2017)]
print(df.shape)

df.drop(['name','abtest','dateCrawled','nrOfPictures','lastSeen','postalCode','dateCreated'],
axis='columns',inplace=True)

new_df=df.copy()
```

```
new_df=new_df.drop_duplicates(['price','vehicleType','yearOfRegistration','gearbox','powerPS','model'
,'kilometer','monthOfRegistration','fuelType','notRepairedDamage'])
```

```
new_df=new_df[(new_df.price>=100)&(new_df.price<=150000)]

new_df['notRepairedDamage'].fillna(value='not-declared',inplace=True)
new_df['fuelType'].fillna(value='not-declared',inplace=True)
new_df['gearbox'].fillna(value='not-declared',inplace=True)
new_df['vehicleType'].fillna(value='not-declared',inplace=True)
new_df['model'].fillna(value='not-declared',inplace=True)
```

4. Conversion of German data to English

```
new_df.gearbox.replace(('manuell','automatik'),('manual','automatic'), inplace=True)
new_df.fuelType.replace(('benzin','andere','elecktro'),('petrol','others','electic'), inplace=True)
new_df.vehicleType.replace(('kleinwagen','cabrio','kambi','andere'),('small
car','convertible','combination','others'), inplace=True)
new_df.notRepairedDamage.replace(('ja','nein'),('Yes','No'), inplace=True)
```

5. Exporting the pre-processed data

```
new_df.to_csv("autos_preprocessed.csv")
```

6. Creating the .npy files

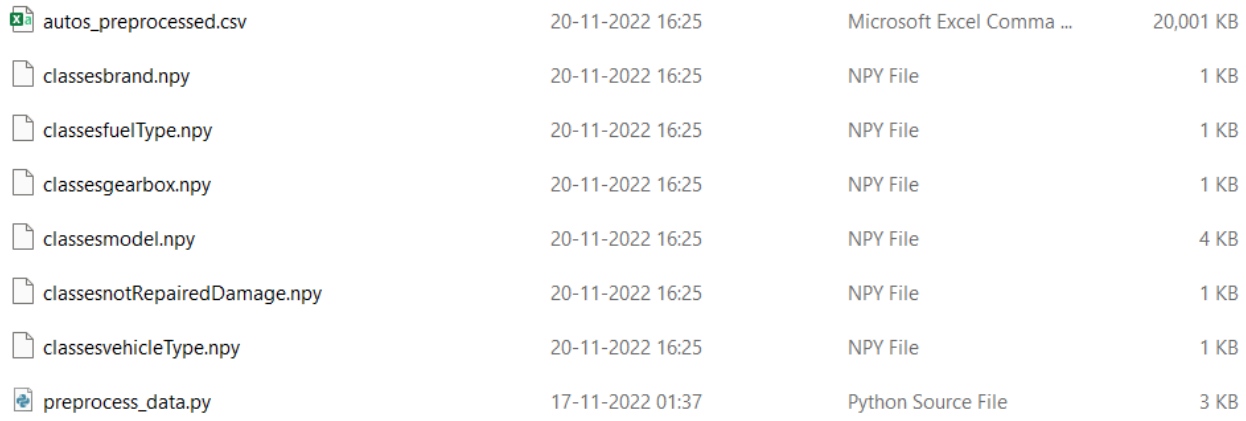
```
l=['gearbox','notRepairedDamage','fuelType','vehicleType','model','brand']

m={}
for i in l:
    m[i]=LabelEncoder()
    m[i].fit(new_df[i])
    tr=m[i].transform(new_df[i])
    np.save(str('classes'+i+'.npy'),m[i].classes_)
    print(i,":",m[i])
    new_df.loc[:,i+'_labels']=pd.Series(tr,index=new_df.index)

l2=new_df[['price','yearOfRegistration','powerPS','kilometer','monthOfRegistration'] + [x+"_labels"
for x in l]]
```

7. Splitting the train and testing data

```
print(l2.columns)

Y=l2.iloc[:,0].values
X=l2.iloc[:,:].values
```

```
Y=Y.reshape(-1,1)

X_train,X_test,Y_train,Y_test=train_test_split(X,Y, test_size=0.3, random_state=3)
```

Output:



autos_preprocessed.csv	20-11-2022 16:25	Microsoft Excel Comma ...	20,001 KB
classesbrand.npy	20-11-2022 16:25	NPY File	1 KB
classesfuelType.npy	20-11-2022 16:25	NPY File	1 KB
classesgearbox.npy	20-11-2022 16:25	NPY File	1 KB
classesmodel.npy	20-11-2022 16:25	NPY File	4 KB
classesnotRepairedDamage.npy	20-11-2022 16:25	NPY File	1 KB
classesvehicleType.npy	20-11-2022 16:25	NPY File	1 KB
preprocess_data.py	17-11-2022 01:37	Python Source File	3 KB

```
  df=df.drop('seller',1)
notRepairedDamage : LabelEncoder()
fuelType : LabelEncoder()
vehicleType : LabelEncoder()
model : LabelEncoder()
brand : LabelEncoder()
Index(['price', 'yearOfRegistration', 'powerPS', 'kilometer',
       'monthOfRegistration', 'gearbox_labels', 'notRepairedDamage_labels',
       'fuelType_labels', 'vehicleType_labels', 'model_labels',
       'brand_labels'],
      dtype='object')
```

7.2 Sprint-2

Here we developed the machine learning model and saved it as a sav file

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import r2_score
r=RandomForestRegressor(n_estimators=1000,max_depth=10,random_state=34)

r.fit(X_train,np.ravel(Y_train,order='C'))

y_pred=r.predict(X_test)

print(r2_score(Y_test,y_pred))

fn="resale_model.sav"
pickle.dump(r,open(fn,'wb'))
```

We are creating the Random Forest Regressor instance named r and then we fit that into our own train data and we are testing it the X_test.

Then we will be comparing its r2_score with Y_test and y_pred

The r2_score for our model is 0.8345401773383525

7.3 Sprint-3

Here we will be developing the web application using the Flask,

The things we have done here are:

1. Created our HTML pages
    a. index.html

```
<!doctype html>
<html lang="en">
  <head>
    <meta charset="utf-8">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Used Car Price Predict</title>
    <!-- Bootstrap CSS -->
    <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
    <!-- Own CSS -->
    <!-- <link rel="stylesheet" href="style.css"> -->
    <style>
    .carousel-item {
    height: 100vh;
    min-height: 350px;
    background: no-repeat center center scroll;
    -webkit-background-size: cover;
    -moz-background-size: cover;
    -o-background-size: cover;
    background-size: cover;
  }
  /* body{
    background-image:url("{{url_for('static',filename='img/img-2.jpg')}}");
     background-image:url("img/img-1.jpg");
  }
  */
</style>
  </head>

  <body>
    <!-- Front page -->
    <nav class="navbar navbar-expand-lg navbar-light bg-light fixed-top">
```

```
      <div class="container">
        <a class="navbar-brand" href="#">CAR RESALE PROJECT</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarResponsive" aria-controls="navbarResponsive" aria-expanded="false" aria-
label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
        <div class="collapse navbar-collapse" id="navbarResponsive">
            <button type="button" class="btn btn-dark ms-auto" data-bs-toggle="modal" data-bs-
target="#cardetails">Predict</button>

        </div>
       </div>
     </nav>

     <header>

       <div id="carouselExampleCaptions" class="carousel slide" data-bs-ride="carousel">
        <div class="carousel-indicators">
          <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="0"
class="active" aria-current="true" aria-label="Slide 1"></button>
          <button type="button" data-bs-target="#carouselExampleCaptions" data-bs-slide-to="1" aria-
label="Slide 2"></button>
        </div>
        <div class="carousel-inner">
          <div class="carousel-item active" style="background-image:
url('{{url_for('static',filename='img/img-1.jpg')}}')"></div>
          <div class="carousel-item" style="background-image:
url('{{url_for('static',filename='img/img-2.jpg')}}')"></div>
        </div>
        <button class="carousel-control-prev" type="button" data-bs-
target="#carouselExampleCaptions" data-bs-slide="prev">
          <span class="carousel-control-prev-icon" aria-hidden="true"></span>
          <span class="visually-hidden">Previous</span>
        </button>
        <button class="carousel-control-next" type="button" data-bs-
target="#carouselExampleCaptions" data-bs-slide="next">
          <span class="carousel-control-next-icon" aria-hidden="true"></span>
          <span class="visually-hidden">Next</span>
        </button>
       </div>
     </header>

    <!-- modal code -->
     <div class="modal fade" id="cardetails" tabindex="-1" aria-labelledby="exampleModalLabel"
aria-hidden="true">
       <div class="modal-dialog">
        <div class="modal-content">
```

```
        <div class="modal-header">
          <h1 class="modal-title fs-5" id="exampleModalLabel">New message</h1>
          <button type="button" class="btn-close" data-bs-dismiss="modal" aria-
label="Close"></button>
        </div>
        <div class="modal-body">
          <!-- FORM-->
          <form action = "http://127.0.0.1:5000/y_predict", method="GET">
            <div class="mb-3">
              <label for="Registration Year" class="col-form-label">Registration Year</label>
              <input type="text" class="form-control" id="Registration Year"
name="regyear",maxlength=50>
            </div>
            <div class="mb-3">
              <label for="regmonth" class="col-form-label">Registration Month (Enter the month
number)</label>
              <input class="form-control",type=text, id="regmonth",name="regmonth">
                <!--
                <option value=""></option>
                <option value=1>Jan</option>
                <option value=2>Feb</option>
                <option value=3>Mar</option>
                <option value=4>Apr</option>
                <option value=5>May</option>
                <option value=6>Jun</option>
                <option value=7>July</option>
                <option value=8>Aug</option>
                <option value=9>Sep</option>
                <option value=10>Oct</option>
                <option value=11>Nov</option>
                <option value=12>Dec</option>
              </select> -->
            </div>
            <div class="mb-3">
              <label for="Power of car" class="col-form-label">Power of car</label>
              <input type="text" class="form-control" id="powerps" name="powerps",maxlength=5>
            </div>

            <div class="mb-3">
              <label for="Kilometer the car driven" class="col-form-label">Kilometer the car
driven</label>
              <input type="text" class="form-control" id="kms" name="kms",maxlength=7>
            </div>

            <div class="mb-2">
              <label for="Gear Type" class="col-form-label">Gear Type</label>
            </div>
            <div class="form-check form-check-inline">
```

```
                <input class="form-check-input" type="radio" name="gearbox" id="gearbox"
value="Manual">
                <label class="form-check-label" for="Manual">Manual</label>
              </div>
              <div class="form-check form-check-inline">
                <input class="form-check-input" type="radio" name="gearbox" id="gearbox"
value="Automatic">
                <label class="form-check-label" for="Autoamatic">Automatic</label>
              </div>
              <div class="form-check form-check-inline">
                <input class="form-check-input" type="radio" name="gearbox" id="gearbox" value="Not
declared">
                <label class="form-check-label" for="ND">Not Declared</label>
              </div>

              <div class="mb-2">
                <label for="Gear Type" class="col-form-label">Car is damaged or repaired</label>
              </div>
              <div class="form-check form-check-inline">
                <input class="form-check-input" type="radio" name="dam" id="dam" value="Yes">
                <label class="form-check-label" for="Manual">Yes</label>
              </div>
              <div class="form-check form-check-inline">
                <input class="form-check-input" type="radio" name="dam" id="dam" value="No">
                <label class="form-check-label" for="Autoamatic">No</label>
              </div>
              <div class="mb-3">
                <label for="Model Type" class="col-form-label">Model Type</label>
                <select class="form-select" aria-label="Default select
example",id="model",name="model">
                  <option value=""></option>
                  <option value="transporter">Transporter </option>
                  <option value="punto">Punto </option>
                  <option value="e_klasse">E Klasse </option>
                  <option value="clio">Clio </option>
                  <option value="kadett">Kadett </option>
                  <option value="kangoo">Kangoo </option>
                  <option value="corsa">Corsa </option>
                  <option value="one">One </option>
                  <option value="fortwo">Fortwo </option>
                  <option value="1er">1er </option>
                  <option value="b_klasse">B Klasse </option>
                  <option value="focus">Focus </option>
                  <option value="tt">Tt </option>
                  <option value="a6">A6 </option>
                  <option value="impreza">Impreza </option>
                  <option value="vectra">Vectra </option>
                  <option value="berlingo">Berlingo </option>
```

```
<option value="80">80 </option>
<option value="m_klasse">M Klasse </option>
<option value="tiguan">Tiguan </option>
<option value="i_reihe">I Reihe </option>
<option value="espace">Espace </option>
<option value="sharan">Sharan </option>
<option value="6_reihe">6 Reihe </option>
<option value="panda">Panda </option>
<option value="up">Up </option>
<option value="seicento">Seicento </option>
<option value="ceed">Ceed </option>
<option value="5_reihe">5 Reihe </option>
<option value="yeti">Yeti </option>
<option value="octavia">Octavia </option>
<option value="mii">Mii </option>
<option value="rx_reihe">Rx Reihe </option>
<option value="6er">6er </option>
<option value="modus">Modus </option>
<option value="fox">Fox </option>
<option value="matiz">Matiz </option>
<option value="beetle">Beetle </option>
<option value="jazz">Jazz </option>
<option value="omega">Omega </option>
<option value="slk">Slk </option>
<option value="7er">7er </option>
<option value="80">80 </option>
<option value="147">147 </option>
<option value="glk">Glk </option>
<option value="100">100 </option>
<option value="z_reihe">Z Reihe </option>
<option value="sportage">Sportage </option>
<option value="sorento">Sorento </option>
<option value="v40">V40 </option>
<option value="5er">5er </option>
<option value="ibiza">Ibiza </option>
<option value="3er">3er </option>
<option value="mustang">Mustang </option>
<option value="signum">Signum </option>
<option value="astra">Astra </option>
<option value="a8">A8 </option>
<option value="jetta">Jetta </option>
<option value="fiesta">Fiesta </option>
<option value="c_klasse">C Klasse </option>
<option value="micra">Micra </option>
<option value="vito">Vito </option>
<option value="sprinter">Sprinter </option>
<option value="156">156 </option>
<option value="escort">Escort </option>
```

```
<option value="x_type">X Type </option>
<option value="5_reihe">5 Reihe </option>
<option value="ducato">Ducato </option>
<option value="s_type">S Type </option>
<option value="x_trail">X Trail </option>
<option value="toledo">Toledo </option>
<option value="altea">Altea </option>
<option value="7er">7er </option>
<option value="voyager">Voyager </option>
<option value="calibra">Calibra </option>
<option value="bravo">Bravo </option>
<option value="range_rover">Range Rover </option>
<option value="4_reihe">4 Reihe </option>
<option value="5er">5er </option>
<option value="6_reihe">6 Reihe </option>
<option value="legacy">Legacy </option>
<option value="pajero">Pajero </option>
<option value="auris">Auris </option>
<option value="niva">Niva </option>
<option value="5_reihe">5 Reihe </option>
<option value="s60">S60 </option>
<option value="nubira">Nubira </option>
<option value="vivaro">Vivaro </option>
<option value="g_klasse">G Klasse </option>
<option value="lodgy">Lodgy </option>
<option value="850">850 </option>
<option value="serie_2">Serie 2 </option>
<option value="6er">6er </option>
<option value="charade">Charade </option>
<option value="croma">Croma </option>
<option value="outlander">Outlander </option>
<option value="gl">Gl </option>
<option value="doblo">Doblo </option>
<option value="musa">Musa </option>
<option value="amarok">Amarok </option>
<option value="156">156 </option>
<option value="move">Move </option>
<option value="9000">9000 </option>
<option value="v60">V60 </option>
<option value="145">145 </option>
<option value="aveo">Aveo </option>
<option value="200">200 </option>
<option value="300c">300c </option>
<option value="b_max">B Max </option>
<option value="delta">Delta </option>
<option value="terios">Terios </option>
<option value="rangerover">RangeRover </option>
<option value="90">90 </option>
```

```
<option value="materia">Materia </option>
<option value="kalina">Kalina </option>
<option value="elefantino">Elefantino </option>
<option value="i3">I3 </option>
<option value="kappa">Kappa </option>
<option value="serie_3">Serie 3 </option>
<option value="48429">48429 </option>
<option value="serie_1">Serie 1 </option>
<option value="discovery_sport">Discovery Sport </option>
<option value="antara">Antara </option>
<option value="tucson">Tucson </option>
<option value="q7">Q7 </option>
<option value="citigo">Citigo </option>
<option value="jimny">Jimny </option>
<option value="cx_reihe">Cx Reihe </option>
<option value="wrangler">Wrangler </option>
<option value="lybra">Lybra </option>
<option value="range_rover_sport">Range Rover Sport </option>
<option value="lancer">Lancer </option>
<option value="159">159 </option>
<option value="freelander">Freelander </option>
<option value="captiva">Captiva </option>
<option value="c2">C2 </option>
<option value="500">500 </option>
<option value="range_rover_evoque">Range Rover Evoque </option>
<option value="sandero">Sandero </option>
<option value="note">Note </option>
<option value="900">900 </option>
<option value="147">147 </option>
<option value="grand">Grand </option>
<option value="fabia">Fabia </option>
<option value="3er">3er </option>
<option value="2_reihe">2 Reihe </option>
<option value="andere">Andere </option>
<option value="c_max">C Max </option>
<option value="3_reihe">3 Reihe </option>
<option value="passat">Passat </option>
<option value="navara">Navara </option>
<option value="ka">Ka </option>
<option value="forester">Forester </option>
<option value="xc_reihe">Xc Reihe </option>
<option value="scenic">Scenic </option>
<option value="a4">A4 </option>
<option value="a1">A1 </option>
<option value="insignia">Insignia </option>
<option value="combo">Combo </option>
<option value="eos">Eos </option>
<option value="touran">Touran </option>
```

```
<option value="getz">Getz </option>
<option value="a3">A3 </option>
<option value="almera">Almera </option>
<option value="megane">Megane </option>
<option value="7er">7er </option>
<option value="1er">1er </option>
<option value="lupo">Lupo </option>
<option value="r19">R19 </option>
<option value="zafira">Zafira </option>
<option value="caddy">Caddy </option>
<option value="2_reihe">2 Reihe </option>
<option value="mondeo">Mondeo </option>
<option value="cordoba">Cordoba </option>
<option value="colt">Colt </option>
<option value="c1">C1 </option>
<option value="rio">Rio </option>
<option value="touareg">Touareg </option>
<option value="logan">Logan </option>
<option value="spider">Spider </option>
<option value="cuore">Cuore </option>
<option value="s_max">S Max </option>
<option value="a2">A2 </option>
<option value="x_reihe">X Reihe </option>
<option value="a5">A5 </option>
<option value="galaxy">Galaxy </option>
<option value="c3">C3 </option>
<option value="viano">Viano </option>
<option value="s_klasse">S Klasse </option>
<option value="1_reihe">1 Reihe </option>
<option value="avensis">Avensis </option>
<option value="sl">Sl </option>
<option value="roomster">Roomster </option>
<option value="q5">Q5 </option>
<option value="kaefer">Kaefer </option>
<option value="santa">Santa </option>
<option value="cooper">Cooper </option>
<option value="leon">Leon </option>
<option value="4_reihe">4 Reihe </option>
<option value="500">500 </option>
<option value="laguna">Laguna </option>
<option value="ptcruiser">Ptcruiser </option>
<option value="clk">Clk </option>
<option value="primera">Primera </option>
<option value="exeo">Exeo </option>
<option value="159">159 </option>
<option value="transit">Transit </option>
<option value="juke">Juke </option>
<option value="qashqai">Qashqai </option>
```

```
<option value="carisma">Carisma </option>
<option value="accord">Accord </option>
<option value="corolla">Corolla </option>
<option value="lanos">Lanos </option>
<option value="phaeton">Phaeton </option>
<option value="boxster">Boxster </option>
<option value="verso">Verso </option>
<option value="swift">Swift </option>
<option value="rav">Rav </option>
<option value="kuga">Kuga </option>
<option value="picanto">Picanto </option>
<option value="kalos">Kalos </option>
<option value="superb">Superb </option>
<option value="stilo">Stilo </option>
<option value="alhambra">Alhambra </option>
<option value="911">911 </option>
<option value="mx_reihe">Mx Reihe </option>
<option value="m_reihe">M Reihe </option>
<option value="roadster">Roadster </option>
<option value="ypsilon">Ypsilon </option>
<option value="cayenne">Cayenne </option>
<option value="galant">Galant </option>
<option value="justy">Justy </option>
<option value="90">90 </option>
<option value="sirion">Sirion </option>
<option value="crossfire">Crossfire </option>
<option value="6_reihe">6 Reihe </option>
<option value="agila">Agila </option>
<option value="duster">Duster </option>
<option value="cr_reihe">Cr Reihe </option>
<option value="v50">V50 </option>
<option value="discovery">Discovery </option>
<option value="c_reihe">C Reihe </option>
<option value="v_klasse">V Klasse </option>
<option value="yaris">Yaris </option>
<option value="c5">C5 </option>
<option value="aygo">Aygo </option>
<option value="cc">Cc </option>
<option value="carnival">Carnival </option>
<option value="fusion">Fusion </option>
<option value="bora">Bora </option>
<option value="forfour">Forfour </option>
<option value="100">100 </option>
<option value="cl">Cl </option>
<option value="tigra">Tigra </option>
<option value="156">156 </option>
<option value="300c">300c </option>
<option value="100">100 </option>
```

```
                <option value="147">147 </option>
                <option value="q3">Q3 </option>
                <option value="spark">Spark </option>
                <option value="v70">V70 </option>
                <option value="defender">Defender </option>
                <option value="cherokee">Cherokee </option>
                <option value="clubman">Clubman </option>
                <option value="samara">Samara </option>
                <option value="2_reihe">2 Reihe </option>
                <option value="1er">1er </option>
                <option value="3er">3er </option>
                <option value="601">601 </option>
                <option value="3_reihe">3 Reihe </option>
                <option value="polo">Polo </option>
                <option value="twingo">Twingo </option>
                <option value="a_klasse">A klasse </option>
                <option value="scirocco">Scirocco </option>
                <option value="5er">5er </option>
                <option value="meriva">Meriva </option>
                <option value="arosa">Arosa </option>
                <option value="c4">C4 </option>
                <option value="civic">Civic </option>

              </select>
            </div>
            <div class="mb-3">
              <label for="Brand" class="col-form-label">Brand</label>
              <select class="form-select" aria-label="Default select
example",id="brand",name="brand">
                <option value=""></option>
                <option value="mini">Mini </option>
                <option value="smart">Smart </option>
                <option value="hyundai">Hyundai </option>
                <option value="sonstige_autos">Sonstige Autos </option>
                <option value="alfa_romeo">Alfa Romeo </option>
                <option value="subaru">Subaru </option>
                <option value="volvo">Volvo </option>
                <option value="mitsubishi">Mitsubishi </option>
                <option value="kia">Kia </option>
                <option value="suzuki">Suzuki </option>
                <option value="lancia">Lancia </option>
                <option value="porsche">Porsche </option>
                <option value="toyota">Toyota </option>
                <option value="chevrolet">Chevrolet </option>
                <option value="dacia">Dacia </option>
                <option value="daihatsu">Daihatsu </option>
                <option value="trabant">Trabant </option>
                <option value="saab">Saab </option>
```

```
                <option value="chrysler">Chrysler </option>
                <option value="jaguar">Jaguar </option>
                <option value="daewoo">Daewoo </option>
                <option value="rover">Rover </option>
                <option value="land_rover">Land Rover </option>
                <option value="lada">Lada </option>
                <option value="volkswagen">Volkswagen </option>
                <option value="audi">Audi </option>
                <option value="peugeot">Peugeot </option>
                <option value="ford">Ford </option>
                <option value="mazda">Mazda </option>
                <option value="nissan">Nissan </option>
                <option value="renault">Renault </option>
                <option value="mercedes_benz">Mercedes Benz </option>
                <option value="opel">Opel </option>
                <option value="seat">Seat </option>
                <option value="citroen">Citroen </option>
                <option value="jeep">Jeep </option>
                <option value="skoda">Skoda </option>
                <option value="bmw">Bmw </option>
                <option value="honda">Honda </option>
                <option value="fiat">Fiat </option>
              </select>
            </div>
            <div class="mb-3">
              <label for="Fuel Type" class="col-form-label">Fuel type</label>
              <select class="form-select" aria-label="Default select example",id="fuel",name="fuel">
                <option value=""></option>
                <option value="petrol"> Petrol </option>
                <option value="diesel"> Diesel </option>
                <option value="hybrid">Hybrid </option>
                <option value="lpg">LPG </option>
                <option value="cng">CNG </option>
                <option value="electric">Electric </option>
                <option value="others">Others </option>
                <option value="not-declared"> Not Declared </option>

              </select>
            </div>
            <div class="mb-3">
              <label for="Vehicle Type" class="col-form-label">Vehicle type</label>
              <select class="form-select" aria-label="Default select
example",id="vehicletype",name="vehicletype">
                <option value=""></option>
                <option value="kleinwagen">Kleinwagen </option>
                <option value="limousine">Limousine </option>
                <option value="coupe">Coupe </option>
                <option value="bus">Bus </option>
```

```
                <option value="kombi">Kombi </option>
                <option value="andere">Andere </option>
                <option value="volkswagen">Volkswagen </option>
                <option value="suv">SUV </option>
                <option value="cabrio">Cabrio </option>

              </select>
            </div>
            <button type="submit" value="submit">Predict</button>
          </form>
        </div>
        <!-- <div class="modal-footer">

            <button type="submit" name="submit" class="btn btn-primary"></button>

        </div> -->
      </div>
    </div>
  </div>
  <!-- Bootstrap JS -->
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>
  <!-- OWN JS -->
  <script src="{{url_for('static',filename='script.js')}}"></script>
 </body>
</html>
```

b. predict.html

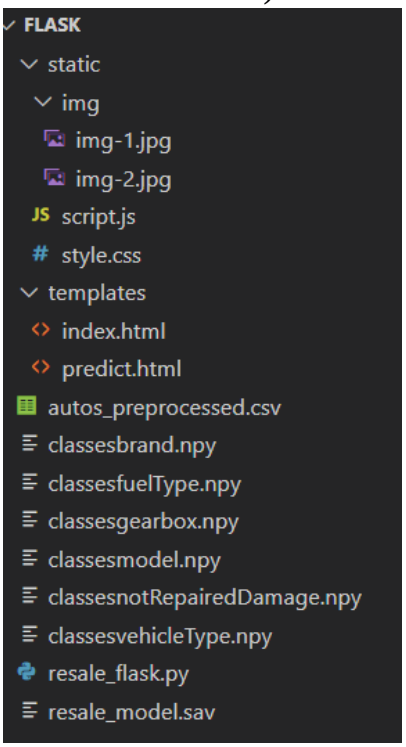
```
<!doctype html>
<html lang="en">
 <head>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <title>Bootstrap demo</title>
  <!-- Bootstrap CSS -->
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1WTRi"
crossorigin="anonymous">
  <!-- Own CSS -->
  <link rel="stylesheet" href="style.css">

 </head>
 <body>
  <nav class="navbar navbar-expand-lg navbar-light bg-light fixed-top">
    <div class="container">
```

```
        <a class="navbar-brand" href="index.html">CAR RESALE PROJECT</a>
        <button class="navbar-toggler" type="button" data-bs-toggle="collapse" data-bs-
target="#navbarResponsive" aria-controls="navbarResponsive" aria-expanded="false" aria-
label="Toggle navigation">
          <span class="navbar-toggler-icon"></span>
        </button>
      </div>
    </nav>

    <div class="container mt-5 pt-5">
        <h1 class="text-center">{{msg}}</h1>
    </div>

    <!-- Bootstrap JS -->
    <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.js"
integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qbsw3"
crossorigin="anonymous"></script>
    <!-- OWN JS -->
    <script src="script.js"></script>
  </body>
```

2. Then we created our CSS file for styling.

```
.carousel-item {
   height: 100vh;
   min-height: 350px;
   background: no-repeat center center scroll;
   -webkit-background-size: cover;
   -moz-background-size: cover;
   -o-background-size: cover;
   background-size: cover;
  }
```

3. Now all HTML files must be placed inside the templates folder (if you don't have done create it) and all the other CSS and image files in static



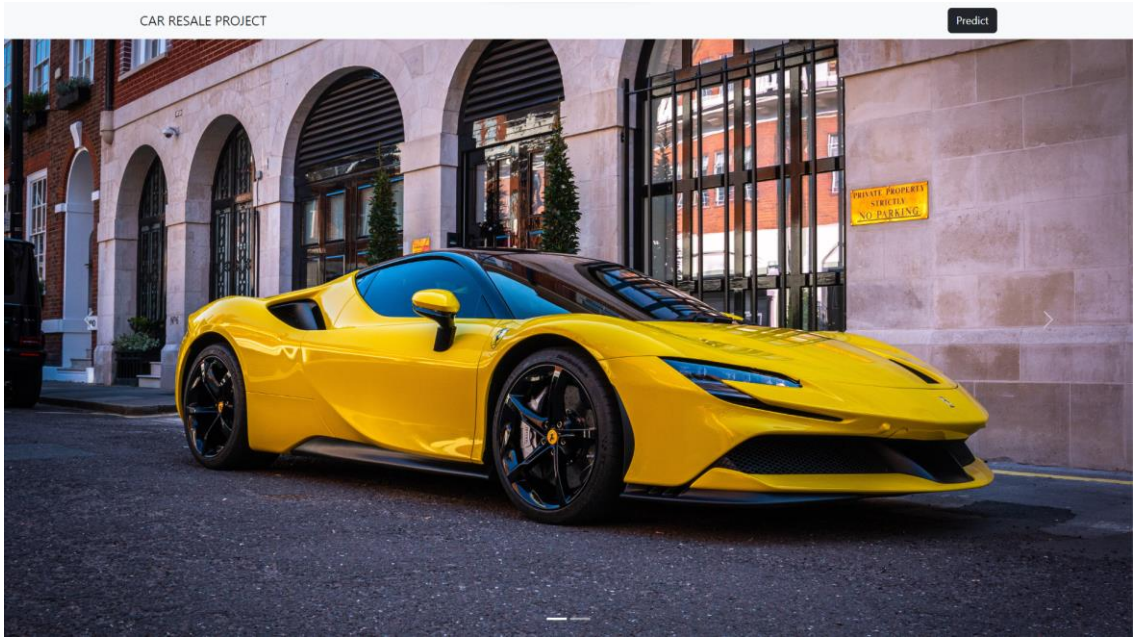
4. Now we create the complete flask application

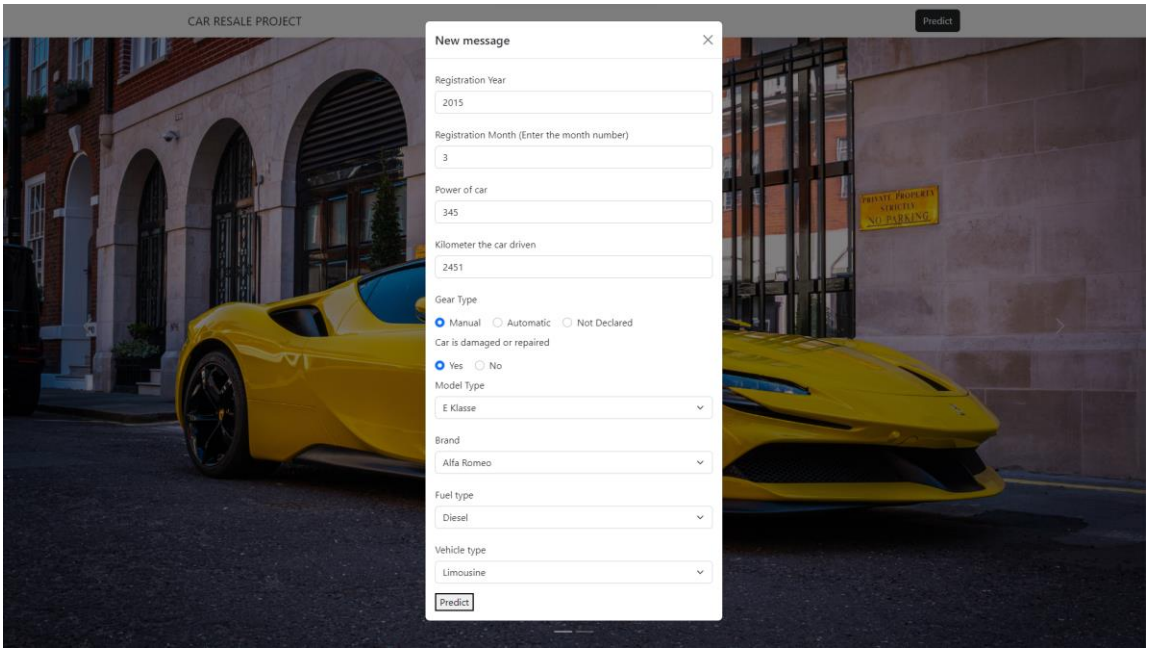
```
from flask import Flask, render_template, request
import pandas as pd
import numpy as np
import pickle
from sklearn.preprocessing import LabelEncoder

app = Flask('_name_',template_folder='templates')
filename ='resale_model.sav'
model_rand = pickle.load(open('resale_model.sav','rb'))
@app.route('/')
def index():
    return render_template('index.html')
def y_predict():
    print(request.args.get('regyear'))
    print(type(request.args.get('regyear')))
    regyear= int(request.args.get('regyear'))
    powerps = float(request.args.get('powerps'))
    kms = float(request.args.get('kms'))
    regmonth = request.args.get('regmonth')
    gearbox = request.args.get('gearbox')
    damage = request.args.get('dam')
    brand = request.args.get('brand')
    model = request.args.get('model')
    fuelType = request.args.get('fuel')
```

```
    vehicletype = request.args.get('vehicletype')
    new_row = {'yearOfRegistration':regyear, 'powerPS':powerps, 'kilometer': kms,
       'monthofRegistration':regmonth, 'gearbox':gearbox,'model':model,
        'brand':brand, 'fuelType':fuelType,
       'vehicleType':vehicletype}
    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 = ['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)
    new_df.describe()
    labeled = new_df[ ['yearOfRegistration','powerPS','kilometer' ,'monthofRegistration',]+[x+'_Labels'
for x in labels]]
    X = labeled.values.tolist()
    y_prediction = model_rand.predict(X)
    return render_template('predict.html',msg = 'The resale value predicted is {:.2f}
$'.format(y_prediction[0]))
app.run(debug=True)
```

Output:

```
 * Serving Flask app '_name_'
 * Debug mode: on
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
 * Running on http://127.0.0.1:5000
Press CTRL+C to quit
 * Restarting with stat
 * Debugger is active!
 * Debugger PIN: 114-051-335
```





The resale value predicted is 52731.38 $

7.4 Sprint-4

In this sprint we deployed our model in the IBM cloud

The below code is used to insert the dataset into the IBM cloud where we must train and run our model

```
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
    ibm_api_key_id='nDK3erqrsighqUE5UEOFDu-ZyuiImZ6U4QoP2_2Ri2Zu',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')

bucket = 'carresalevaluepredictionibmklncei-donotdelete-pr-x5jzl2jrfwjnzx'
object_key = 'autos1.csv'

body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType( __iter__, body )

df = pd.read_csv(body)
df.head()
```

Then we need to install the ibm_watson_machine_learning package

```
!pip install ibm_watson_machine_learning
```

Now we must get our url and api key and must insert in the below code

```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url":"https://us-south.ml.cloud.ibm.com",
    "apikey":"DceKrEN9EBsmpXev8xO21lXYhIohOV5BPm4DVs72adUW"
}
client=APIClient(wml_credentials)
```

You can get your region link from
https://cloud.ibm.com/docs/overview?topic=overview-locations

Then get the API key

API key details

Name
carresalemodel

Description

ID
ApiKey-3180b458-2d3c-4f49-808e-e824c007187f

Status
Unlocked

Email
910619205068@smartinternz.com

Created by
Venkatesh T

Date created
2022-11-20 06:47 GMT

Last authentication
2022-11-21 10:00:12:192 GMT

Auth count
69

Now create space and then give the software spec needed

```
def guid_from_space_name(client,space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources'] if item['entity']["name"] == space_name)['metadata']['id'])
```

```
space_uid=guid_from_space_name(client,'carresale model')
print(space_uid)
```

```
2c9c0aec-de8e-4fa1-8dd6-9f48044b8de2
```

```
client.set.default_space(space_uid)
```

```
'SUCCESS'
```

```
client.software_specifications.list()
```

```
software_spec_uid = client.software_specifications.get_uid_by_name('runtime-22.1-py3.9')
software_spec_uid
```

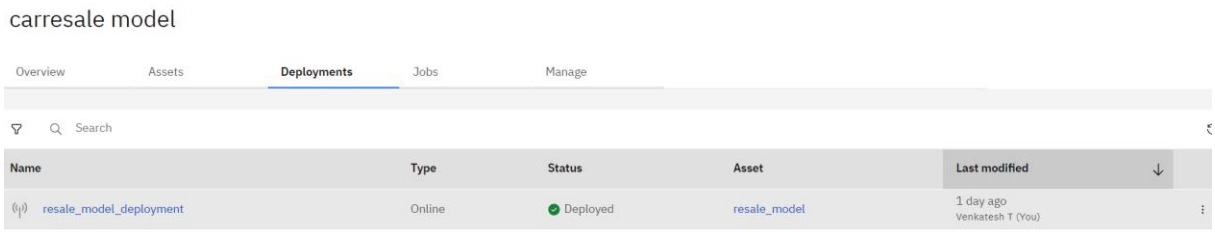
```
'12b83a17-24d8-5082-900f-0ab31fbfd3cb'
```

```
model_details=client.repository.store_model(model=r,meta_props={
    client.repository.ModelMetaNames.NAME:"resale_model",
    client.repository.ModelMetaNames.TYPE:"scikit-learn_1.0",
    client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid
})

model_id = client.repository.get_model_id(model_details)
```

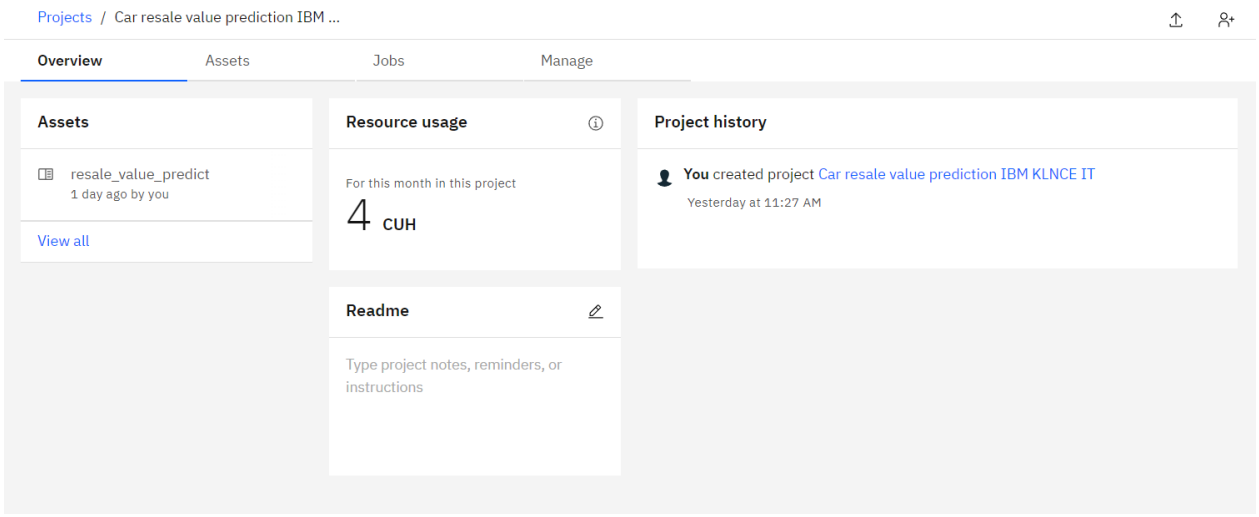
```
model_id
```

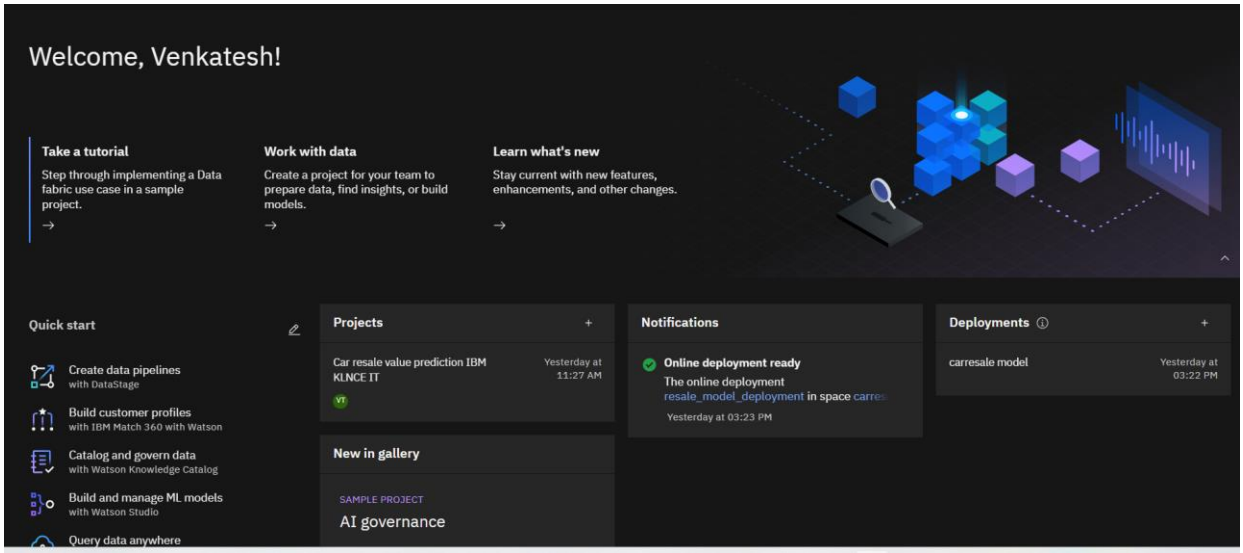
```
'8480c953-e0a8-47e7-b573-be0d6370b164'
```

Now your model will be deployed once you have clicked the deploy button from the assets

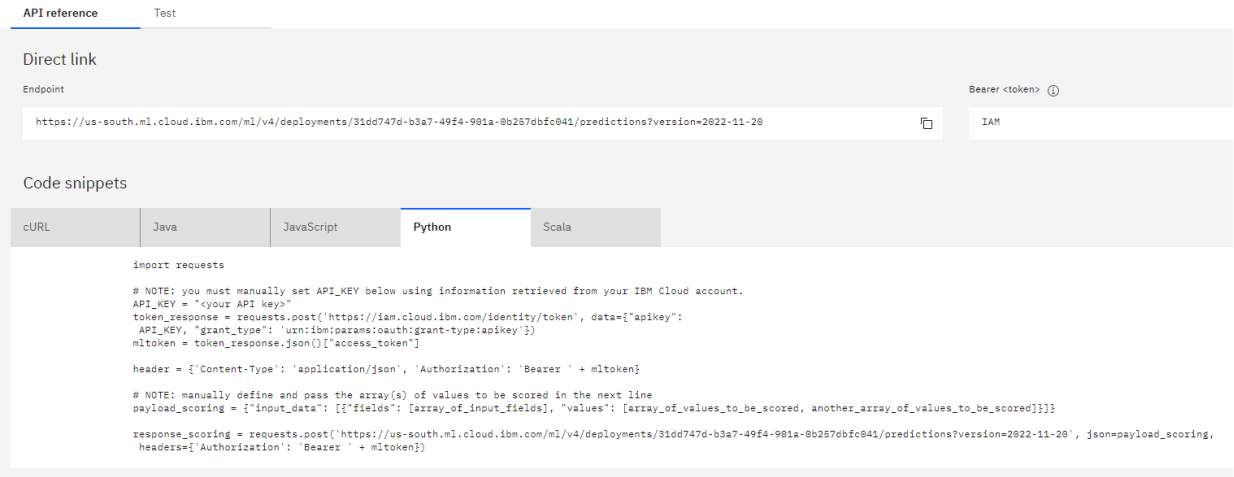


Screenshots of the dashboards





The endpoint and response coding are below



8. TESTING

8.1 Test Cases

Test case ID	Feature Type	Component	Test Scenario	Pre-Requisite	Steps To Execute	Test Data	Expected Result	Actual Result	Status	Commnets	TC for Automation(Y/N)	Executed By
TC_001	UI	Home Page	Page loading and proper integration with CSS and Bootstrap	Laptop or moble with URL	Enter the url and check the homepage	URL	Login/Signup popup should display	Working as expected	Pass	All works fine	N	Sharukesh B
TC_002	Functional	Register form	Able to enter all the data and send to backend	Laptop or moble with URL	Enter the details and click predict	Some random data	All data must be acquired	Working as expected	Pass	Less time consumption	N	Trinity E
TC_003	Functional	Backend	Ability to capture the entered data and send it to ML model in the cloud	Liberaries like pandas, numpy, sklearn, flask	Use GET or POST method to get the data	Input data	User should navigate to user account homepage	Working as expected	Pass	easy to test	Y	Vinod T H
TC_004	Functional	Backend	To ensure the end point scoring is properly integrated	API key and End point link with the region link	Use IBM Cloud to get the key and end point link and search for the region link in the web	autos.csv file (Our dataset)	Application should show 'Incorrect email or password ' validation message.	Working as expected	Pass	IBM cloud was easy to use	N	Venkatesh T
TC_005	Functional	Backend	Prediction is proper?	R2score, MSE, MAE etc	Use the test and train data to compute the values	Train and Test data	Application should show 'Incorrect email or password ' validation message.	Working as expected	Pass	Accurate results	Y	Sutharsan R

8.2 User Acceptance Testing

1. Defect Analysis

This report shows the number of resolved or closed bugs at each severity level, and how they were resolved

Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	8	2	9	3	22
Duplicate	0	0	0	0	0
External	4	3	0	1	8
Fixed	11	5	8	4	28
Not Reproduced	0	0	0	0	0
Skipped	0	0	1	0	1
Won't Fix	0	0	0	0	0
Totals	22	10	18	8	59

2. Test Case Analysis

This report shows the number of test cases that have passed, failed, and untested

Section	Total Cases	Not Tested	Fail	Pass
Cleaning data	15	0	0	15
Creating .npy	3	0	0	3
ML model building	4	0	0	4
Flask app	23	0	0	23
ML model delpoyment	18	0	0	18
Final Report Output	6	0	0	6

9. RESULTS

9.1 Performance Metrics

- 'mae': 1635.2268363398357,
- 'mse': 11836295.137243608,
- 'rmse': 3440.391712762314,
- 'rmsle': 8.143340613873793,
- 'r2': 0.8345401773383525,
- 'adj_r2_score': 0.8345316919146367

10. ADVANTAGES & DISADVANTAGES

Advantages:

- An effective and efficient model
- Has a good accuracy
- Simple and easy UI
- Data privacy is maintained

Disadvantages:

- Not many options
- Chat bot is not available

11. CONCLUSION

Considering the rising cost of new cars, purchasing a used vehicle is a more financially viable option.

The consumer in need may find it more helpful if there is a system in place to estimate used automobile values.

The suggested system will assist in calculating an accurate used car price.

12. FUTURE SCOPE

We would like to add the features like

- Providing chatbot function
- Sending the details to third parties if user wants to
- Provide digital signatures for documets

13. APPENDIX

Source code:

Resale_flask.py

```
# -*- coding: utf-8 -*-
"""
Created on Thu Nov 17 01:23:56 2022

@author: vkedu
"""
from flask import Flask, render_template, request
import pandas as pd
import numpy as np
import pickle
from sklearn.preprocessing import LabelEncoder
import requests
import json

app = Flask('_name_',template_folder='templates')
@app.route('/')
def index():
    return render_template('index.html')
@app.route('/predict')
def predict():
    return render_template('resalepredict.html')
@app.route('/y_predict', methods=['GET', 'POST'])
def y_predict():
    regyear= int(request.args.get('regyear'))
    powerps = float(request.args.get('powerps'))
    kms = float(request.args.get('kms'))
    regmonth = request.args.get('regmonth')
```

```
    gearbox = request.args.get('gearbox')
    damage = request.args.get('dam')
    brand = request.args.get('brand')
    model = request.args.get('model')
    fuelType = request.args.get('fuel')
    vehicletype = request.args.get('vehicletype')
    regmonth=3
    new_row = {'yearOfRegistration':regyear, 'powerPS':powerps, 'kilometer': kms,
       'monthofRegistration':regmonth, 'gearbox':gearbox,'model':model,
        'brand':brand, 'fuelType':fuelType,
       'vehicleType':vehicletype}
    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 = ['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)

    new_df.describe()
    labeled = new_df[ ['yearOfRegistration'
                       ,'powerPS'
                       ,'kilometer'
                       ,'monthofRegistration',
                       ]+[x+'_Labels' for x in labels]]

    X = labeled.values.tolist()
    API_KEY = "DceKrEN9EBsmpXev8xO21lXYhIohOV5BPm4DVs72adUW"
    token_response = requests.post('https://iam.cloud.ibm.com/identity/token', data={"apikey":
    API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-type:apikey'})
    mltoken = token_response.json()["access_token"]

    header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' + mltoken}

    # NOTE: manually define and pass the array(s) of values to be scored in the next line
    payload_scoring = {"input_data": [{"field":
[['yearOfRegistration','monthofRegistration','powerPS','kilometer','gearbox', 'notRepairedDamage',
'model','brand', 'fuelType', 'vehicleType']], "values": X}]}

    response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/31dd747d-
b3a7-49f4-901a-0b257dbfc041/predictions?version=2022-11-20', json=payload_scoring,
    headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring response")
    print(response_scoring.json())
```

```
    y_prediction=response_scoring.json()
    print(y_prediction)
    return render_template('predict.html',msg = 'The resale value predicted is {:.2f}
$'.format(y_prediction['predictions'][0]['values'][0][0]))

app.run(debug=True)
```

Github link

https://github.com/IBM-EPBL/IBM-Project-32084-1660207984

Demo video link

https://drive.google.com/drive/folders/1jQTy1kHEpJp4R2OSf3RR5QdfOsiYeOtq?usp=share_link