1 INTRODUCTION

1.1 Project Overview

Used car resale market in India was marked at 24.2 billion US dollars in 2019. Due to the huge requirement of used cars and lack of experts who can determine the correct valuation, there is an utmost need of bridging this gap between sellers and buyers. This project focuses on building a system that can accurately predict a resale value of the car based on minimal features like kms driven, year of purchase etc. without manual or human interference and hence it remains unbiased.In this project we have used different algorithms with different techniques for developing Car resale value prediction systems considering different features of the car. In a nutshell, car resale value prediction helps the user to predict the resale value of the car depending upon various features like kilometers driven, fuel type, etc. This resale value prediction system is made for general purpose to just predict the amount that can be roughly acquired by the user. We try to predict the amount of resale by best 70% accuracy so the user can get estimated value before he resales the car and doesn't make a deal in loss.

1.2 Purpose

The sale of second-hand imported cars is increasing as the usage also increases. 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 LITERATURE SURVEY

2.1 Existing problem

With the recent arrival of internet portals, buyers and sellers may obtain an appropriate status of the factors that ascertain the market price of a used automobile. Lasso Regression, Multiple Regression, and Regression Trees are examples of machine learning algorithms. We will try to develop a statistical model that can forecast the value of a pre-owned automobile based on prior customer details and different parameters of the vehicle. [2] This paper aims to compare the efficiency of different models' predictions to find the appropriate one. On the subject of used automobile price prediction, several previous studies have been conducted. To anticipate the value of pre-owned automobiles in Mauritius, Pudaruth employed naive Bayes, k-nearest neighbors, multiple linear regression, and decision trees. However, because there were fewer cars observed, their results were not good for prediction. In his article, Pudaruth concluded that decision trees and naive Bayes are ineffective for continuous-valued variables.[4] To anticipate the price of a vehicle, Noor and Jan employed multiple linear regression. They used a variable selection methodology to determine the variables that had the highest influence and then eliminated the remainder. Only a few variables are included in the data, which were utilized to create the linear regression model. With an R-square of 98 percent, the outcome was outstanding. [4] Peerun et al. conducted a study to assess the neural network's performance in predicting used automobile prices. However, especially on higher-priced cars, the estimated value is not very close to the real price. In forecasting the price of a used car, they found that support vector machine regression outperformed neural networks and linear regression by a little margin. [4] To accurately anticipate the price of a car, many different approaches have been used in the digital world, ranging from machine learning approaches like multiple linear regression, k-nearest neighbor, and naive bayes to random forest and decision tree to the SAS enterprise miner. In [7], [8], [9], [10] and [11] all of these solutions took into account distinct sets of attributes when making predictions based on the historical data used to train the model. We attempted to construct a web application where a user may verify the effective market price of their automobiles using a model for prediction based on the factors that have the greatest impact on vehicle prices.[12]The whole data set collected in this research has been split into training (90%) and testing (10%) subsets and Artificial Neural Network, Support Vector Machine and Random Forest classifiers models were built.This research, PHP scripts were built to normalize, standardize, and clean data to avoid unnecessary noise for machine learning algorithms.[13]The process started with pre-processing of data by filling missing values, encoding categorical data,

splitting the data and feature scaling. RandomizedSearchCV is used for tuning the hyper-parameter. Random Forest Algorithm and Extra Tree Regression algorithm. Is used for model construction. Cross-validation is an analysis technique and it is used for the assessment of the results.Good at learning complex and non-linear relationships.[14]In,Training phase: The system is trained by using the data in the data set and fits a model (line/curve) based on the algorithm chosen accordingly,At,Testing phase: the system is provided with the inputs and is tested for its working.In,Linear regression, Lasso regression, and ridge regression are used for constructing the model.Good accuracy is obtained by combining three different machine learning algorithms like Linear Regression, Lasso Regression and Ridge Regression.[15]An efficient machine learning model is built by training, testing, and evaluating three machine learning regressors named Random Forest Regressor, Linear Regression, and Bagging Regressor.As a result of pre-processing and transformation, Random Forest

2.2 References

[1] Doan Van Thai, “Prediction car prices using quantify qualitative data and knowledge-based system.”

[2] Pattabiraman Venkatasubbu, “Used Cars Price Prediction using Supervised Learning Techniques.”

[3] Nitis Monburinon, “Prediction of Prices for Used Car by Using Regression Models”

[4] https://towardsdatascience.com/used-car-priceprediction-using-machine-learninge3be02d977b2

[5] https://www.semanticscholar.org/paper/vehiclePrice-Prediction-System-using-Machine-NoorJan/fc87ead6754b188b1b8629db77badf361fd24a22

[6] https://www.docsity.com/en/research-projectproposal-online-car-rental-system/5232831/

[7] Comparative Analysis of Used Car Price Evaluation Models, Tongji University, Shanghai 200000, China.

[8] Nitis Monburinon, “Prediction of Prices for Used Car by Using Regression Models”, 5th International Conference on Business and Industrial Research, (ICBIR), Bangkok, Thailand,2018

[9] Jaideep A Muley, “Prediction of Used Cars’ Prices by Using SAS EM”, Oklahoma State University

[10] Nabarun Pal, “A methodology for predicting used cars prices using Random Forest”, Future of Information and Communications Conference, 2018

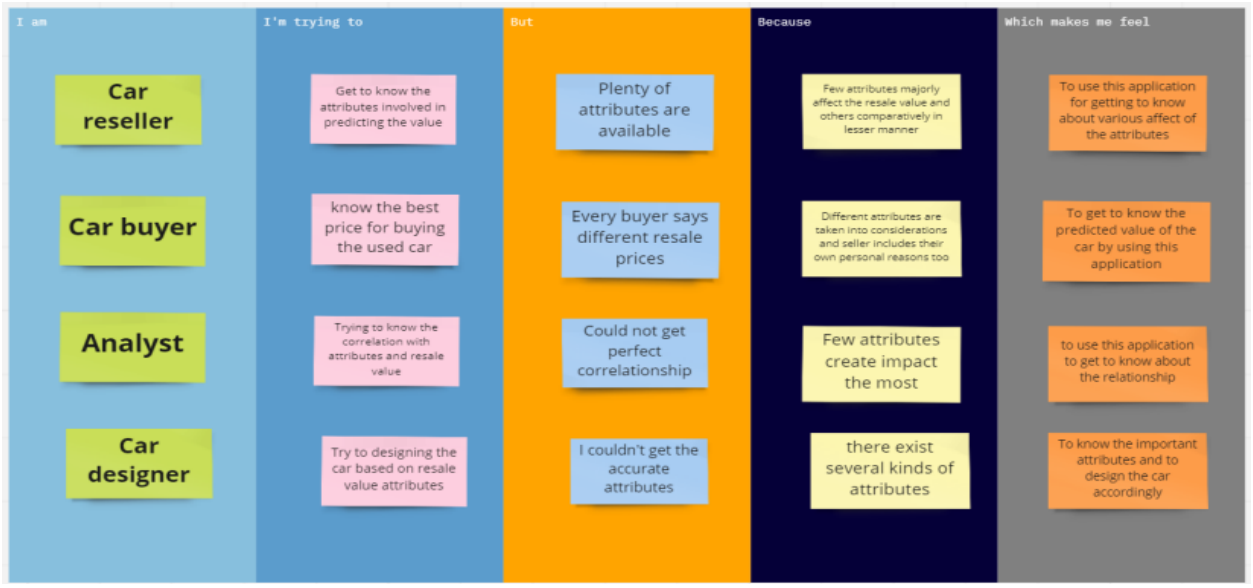
[11] Kuiper, Shonda, "Introduction to Multiple Regression: How Much Is Your Car Worth?" - Journal Of Statistics Education, 2008.

[12] Car Price Prediction using Machine Learning Techniques Enis Gegic, Becir Isakovic,

Dino Keco, Zerina Masetic, Jasmin Kevric [Feb 2019] TEM Journal
[13] Car's Selling Price Prediction using Random Forest Machine Learning AlgorithmAbhishek Pandey, Vanshika Rastogi, Sanika Singh[2019] 5th International Conference on Next Generation Computing Technologies
[14] Used car price prediction Praful Rane, Deep Pandya, Dhawal Kotak[Apr 2021]International Research Journal of Engineering and Technology (IRJET)
[15] Used Cars Price Prediction and Valuation using Data Mining Techniques Abdulla AlShared[Dec 2021 RIT scholar works(theses)

2.3 Problem Statement Definition

The sale of second-hand imported cars is increasing as the usage also increases. 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.



3 Ideation & Proposed Solution

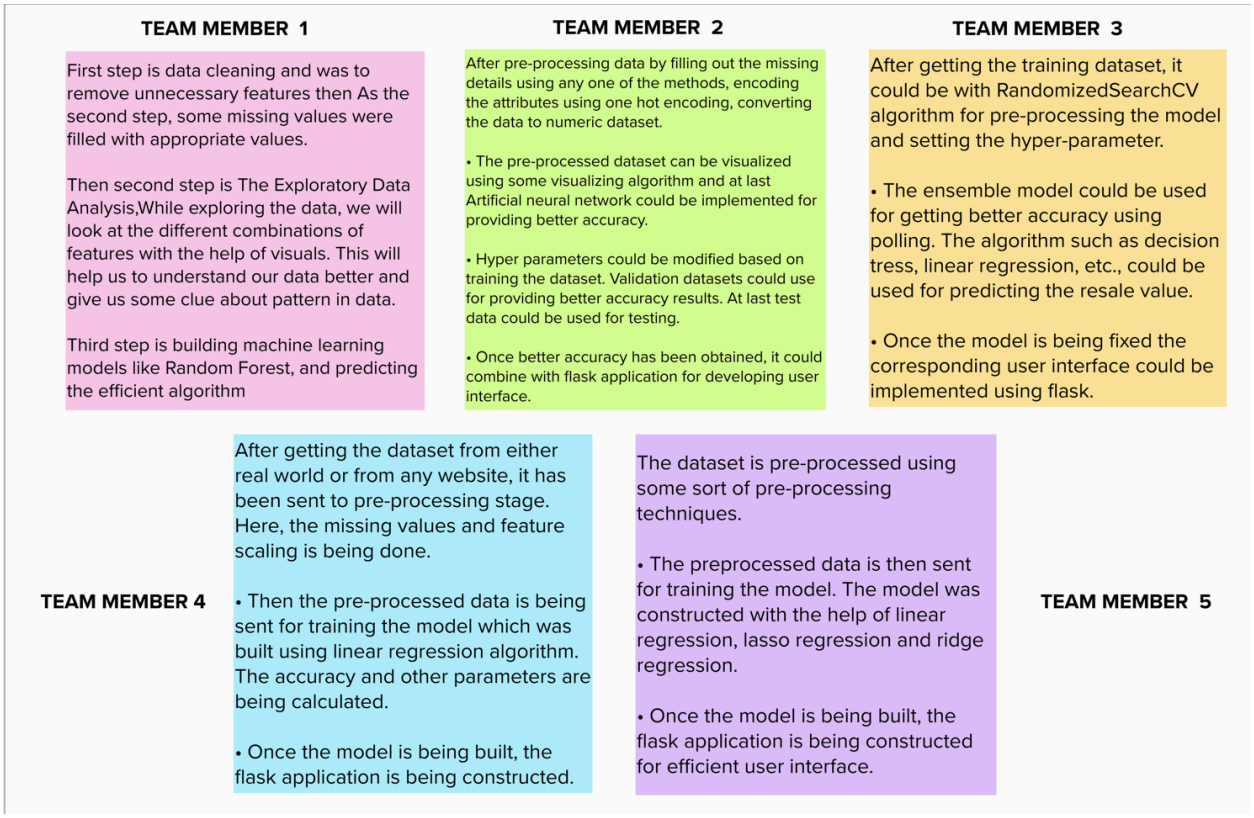
3.1 Empathy Map Canvas

An empathy map is a simple, easy-to-digest visual that captures knowledge about a user's behaviours and attitudes. It is a useful tool to helps teams better understand their users. Creating an effective solution requires understanding the true problem and the person who is experiencing it. The exercise of creating the map helps participants consider things from the user's perspective along with his or her goals and challenges.

3.2 Ideation & Brainstorming

The data is collected from various sources. After finalizing the dataset, the pre-processing step is being carried out. It includes handling null values, normalization, aggregation, feature selection, attribute selection, one hot encoding and outlier analysis. Then preprocessed dataset is trained and tested using several regression models like multiple linear regression, decision tree, support vector regression, lasso regression, random forest, ridge regression, neural network regression, KNN, gaussian and gradient descent. After testing is

being performed, the model is evaluated with metrics such as Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), R squared, Max error, etc., Based on these values, the best model is chosen and used for implementation. As the final step, the web application is created using flask for launching to the users.



3.3 Proposed Solution:

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The sale of second-hand imported cars is being increasing as the usage also increases. 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	The data is collected from various sources. After finalizing the dataset, the pre-processing step is being carried out. It includes **handling null values, normalization, aggregation, feature selection, attribute selection, one hot encoding and outlier analysis**. Then pre-processed dataset is trained and tested using several regression models like **multiple linear regression, decision tree, support vector regression, lasso regression, random forest, ridge regression, neural network regression, KNN, gaussian and gradient descent**. After testing is being performed, the model is evaluated with metrics such as **Mean Absolute Error (MAE), Root Mean Squared Error (RMSE), R squared, Max error**, etc., Based on these values, the best model is chosen and used for implementation. As the final step, the web application is created using flask for launching to the users.
3.	Novelty / Uniqueness	The solution includes several models. As many models are used for evaluation, the better performance would be achieved. Further several pre-processing steps are being carried out to improve the performance. The model is being finalized by allowing the dataset to train and test through at most all the regression models.Launching as the website allows every user to correctly analyse their resale value of the car. The accuracy helps the seller to resale their car at a valid price. It also helps customer in buying the car with appropriate price.

4.	Social Impact / Customer Satisfaction	This act as the solution for seller and for buyer. The seller no need to get worried about the price that is required to resale. The customer could also accept the resale price of the car as algorithms considers even minute factors. It doesn't need anyone to reach out a particular person regarding this. This also helps in identifying buyers, who resale the car fault price. Apart from this, one could also get to know about the features or attributes which are involved in detecting the resale value. It is feasible and could be efficiently used by anyone from anywhere. It decreases the workload of both customer and buyer.
5.	Business Model (Revenue Model)	It could be visualized as the business model as it sounds as the efficient application in predicting the price of the car. One could tie with the buyer and sale the application to him. This could also be used in the separate analysing department. The analyser would receive a greater profit over this.

6.	Scalability of the Solution	With the same model that is being built, the application could be used anywhere and at any time. Depending on the countries it is being implemented, the number of features could be added. It helps in accessing this application across several countries. It reduces the workload of individual, in identifying seller and knowing the resale value. The user just needs to fill all known details on the website to predict the resale value.

3.4 Problem Solution fit:

The Problem-Solution Fit simply means that you have found a problem with your customer and that the solution you have realized for it actually solves the customer's problem. It helps entrepreneurs, marketers and corporate innovators identify behavioral patterns and recognize what would work and why

Purpose:

❑ Solve complex problems in a way that fits the state of your customers.
❑ Succeed faster and increase your solution adoption by tapping into existing mediums and channels of behavior.
❑ Sharpen your communication and marketing strategy with the right triggers and messaging.
❑ Increase touch-points with your company by finding the right problem-behavior fit and building trust by solving frequent annoyances, or urgent or costly problems.
❑ Understand the existing situation in order to improve it for your target group.

Define CS, fit into CC

1. CUSTOMER SEGMENT(S) CS

- Individuals interested in selling the car.
- People interested in buying the car.
- Data analyzer.
- One who is fond of analyzing various characteristic of the car

6. CUSTOMER CONSTRAINTS CC

- Network Connection
- Details of the car
- Available devices

5. AVAILABLE SOLUTIONS AS

Many tools are available online to check the value of used cars

Pros:

- It's simple and takes only a few seconds.
- Few parameters are considered for evaluation.

Cons:

- It may be inaccurate.

Explore AS, differentiate

2. JOBS-TO-BE-DONE / PROBLEMS J&P

- **Economic Factors**
- **Vehicle Make**
- **Vehicle Class and Body Style**
- **Mileage**
- **Exterior Condition**
- **Mechanical Wear and Tear history**
- **Maintenance History**
- **Accident History**

9. PROBLEM ROOT CAUSE RC

The customer need to know the car resale value because, resale value is one of the most important aspects to look at when buying a car. **All cars depreciate in value every year**. But along with a vehicle's annual depreciation, there are certain other factors that can bring down the resale value

7. BEHAVIOUR BE

It's simple and takes only a few seconds. Just fill in your car's details like Brand, Model, Variant, Year of registration, etc, and click on the 'Check Valuation' button. And that's it - the Used Car Valuation tool will work

3. TRIGGERS TR

- Customers budget and lifestyle
- Fuel economy and performance
- Easy insurance and Mileage
- Easy financing
- Comfort and safety
- Models of the cars

4. EMOTIONS: BEFORE / AFTER EM

- Not sure of cost -> Predict the value
- Cheated by few -> cheating reduced

10. YOUR SOLUTION SL

- Predict the car resale value from the characteristics.
- Several algorithms are used for prediction.
- Atlast best algorithm is identified and implemented.
- Provided a web-based application to clients.

8. CHANNELS of BEHAVIOUR CH

8.1 ONLINE

- Test the predicted value by entering the features' values.

8.2 OFFLINE

- Confirmation can be done offline after verifying the car.

4 REQUIREMENT ANALYSIS

4.1 Functional Requirements:

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form(optional)
FR-2	User Confirmation	Confirmation for registration
FR-3	Car details registration	Registering the Car details
FR-4	Value Prediction	Predicting the resale value of the registeredcar

4.2 Non-functional Requirements:

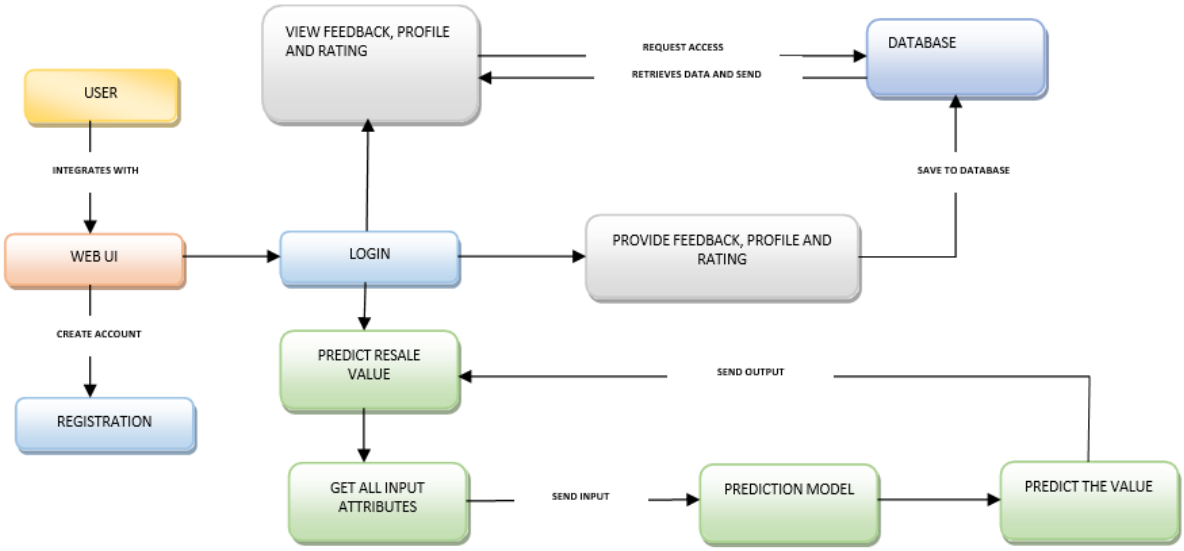
Following are the non-functional requirements of the proposed solution.

FR No.	Non-Functional Requirement	**Description**
NFR-1	**Usability**	Car resale value prediction system is made with the purpose of predicting the correct valuation of used cars that helps users to sell the car
NFR-2	**Security**	Providing security to the website which is developed such that no personal details of the users are hacked
NFR-3	**Reliability**	Providing reliability by predicting the resale values of different types of cars.
NFR-4	**Performance**	Providing high performance by comparing various machine learning algorithms for prediction.
NFR-5	**Availability**	It is available for a diverse range of cars
NR-6	**Scalability**	It is scalable for a large variety of users i.e., eventhough different users are using the website at the same time the efficiency of prediction will not vary

5 PROJECT DESIGN

5.1 Data flow diagram:

A Data Flow Diagram (DFD) is a traditional visual representation of the information flows within a system. A neat and clear DFD can depict the rightamount of the system requirement graphically. It shows how data enters and leaves the system, what changes the information, and where data is stored.



For the proposed system the data flows in the manner in such a way that user interacts with UI using web link. The user will be navigated to the page for creating the account that is registration. Then the flow reaches login page. Upon successful login, it will be redirected to the dashboard. On clicking feedback the page leads to the providing feedback page. On theb dashboard the history details will be displayed. On clicking feedback icon one could view all the feedback list. Upon clicking predict button, the navigation moves on to the predict page. In the predict page the values entered to each attribute by the user will be passed to the api where the model is built. The model is built and deployed on ibm. The requested data will be taken as input by the model and predicted value will be sent as repsonse in json format. After that particular value can be retrieved and displayed to the user. On adding feedback, craeting account and making prediction, the corresponding data will be added to the models in database. Logout helps the user to logout from the account and returns to their home page.

5.2 Solution & Technical Architecture:

The Deliverable shall include the architectural diagramas below and the information as per the table1& table 2

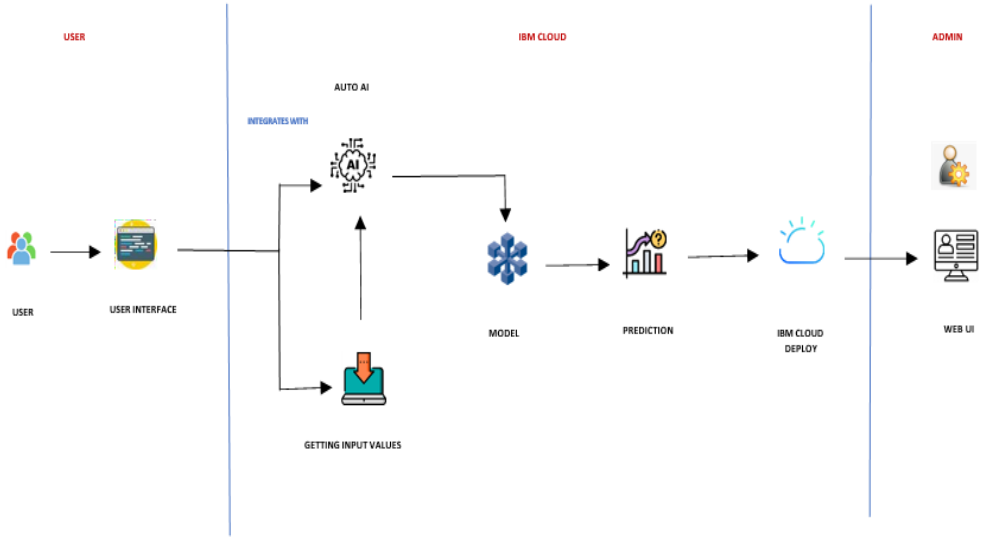


Table-1 : Components & Technologies:

S.No	Component	Description	Technology
1.	User Interface	The userinteracts with theapplication through Web UI.	HTML, CSS, JavaScript, Flask - Python framework.
2.	Application Logic-1 (Login)	Before enjoying the functionalities provided by the website, the user needs to register. Registration ismeant to be done initially. Once an account is created, the login can be made usingusername and password.	Flask

3.	Application Logic-2 (predicting resalevalue)	After logging in, users can provide all thenecessary details that are involved in predicting theresale value of the car. The details will be given to the trainedML model for making predictions. The predicted output will be displayed to the user.	Regression model(Machine learning)
4.	Application Logic-3 (Feedback)	This allows the users to enter their feedback. The provided feedback will be stored in the databases.This database helpsin listing all the feedback provided by the user.	SQLite 3, Flask
5.	Database	Data Type,Configurations etc.	SQLite3
6.	Cloud Database	Database Service on Cloud	Watson studio, Watson studio for Machine Learning
7.	File Storage	File storagerequirements	IBM cloud object storage
8.	External API-1	Machine learning model trained API	IBM cloud
9.	External API-2	None	None
10.	Machine Learning Model	To predict the resale value of the car from the provided values for the attributes.	Regression model. [RandomForestRegressor]
11.	Infrastructure (Server/ Cloud)	IBM cloud service is used for the purpose ofdeployment.	IBM cloud.

Table-2: Application Characteristics:

S.No	CHARACTERISTICS	DESCRIPTION	TECHNOLOGY
1.	Open-Source Frameworks	For developing web application Flask is being used. It's a web development framework. It uses python as its language. It helps in developing frontend of the website. It helps in integrating backendwith the IBM cloud.	Flask
2.	Security Implementations	Fewalgorithms which are inbuilt for network is being used. The framework contains internetprotocol, etc.	e.g., IAM Controls.
3.	Scalable Architecture	This can be used from anywhere across with the help of internet. Since it's goingto be launched in the IBM cloud,it can be accessed by anyone.	IBM cloud
4.	Availability	This is available to all the persons having internet facility and website linkwith them.	IBM cloud
5.	Performance	The model that is trained enhances the performance and provides results in few minutes.	Trained model.

5.3 User Stories:

User Type	Functional Requirement (Epic)	User Story Number	User Story/ Task	Acceptance criteria	Priority	Release
Customer (Web user)	Registration	USN-1	As a user, I canregister for the application byentering my email, password, and confirming my password.	I can access my account / dashboard	High	Sprint-1
Customer (Web user)		USN-2	As a user,I will receive a confirmation email onceI have registered for the application.	I can receive confirmation email&click confirm	High	Sprint-1
Customer (Web user)	Login	USN-3	As a user,I can log intothe application by entering email& password.	I can access the dashboard	High	Sprint-1
Customer (Web user)	Dashboard	USN-4	Users can get to know about all thefunctionaliti es of the system.	I can view all the functionaliti es in the dashboard	High	Sprint-1
Customer (Web user)	View feedback	USN-5	Users can view others' feedback.	I can view the feedback	Low	Sprint-2
Customer (Web user)	Provide feedback	USN-6	Users canenter their feedback andratings.	I can enter the feedback	High	Sprint-2
Customer (Web user)	Predict resale value	USN-7	Users can enterall the values of the attribute to predict the resale value.	I can predict the resale value by providing input	High	Sprint-2

User Type	Functional Requirement (Epic)	User Story Number	User Story/ Task	Acceptance criteria	Priority	Release
Customer Care Executive	Query handling	USN-8	Executive can handlequeries from the user	I can response to the queries	Medium	Sprint-2
Administrator	Administration	USN-9	Administrators can solvethe database related issues.	I can solve the issuesrelated to database or backend.	Medium	Sprint-2

6 PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation:

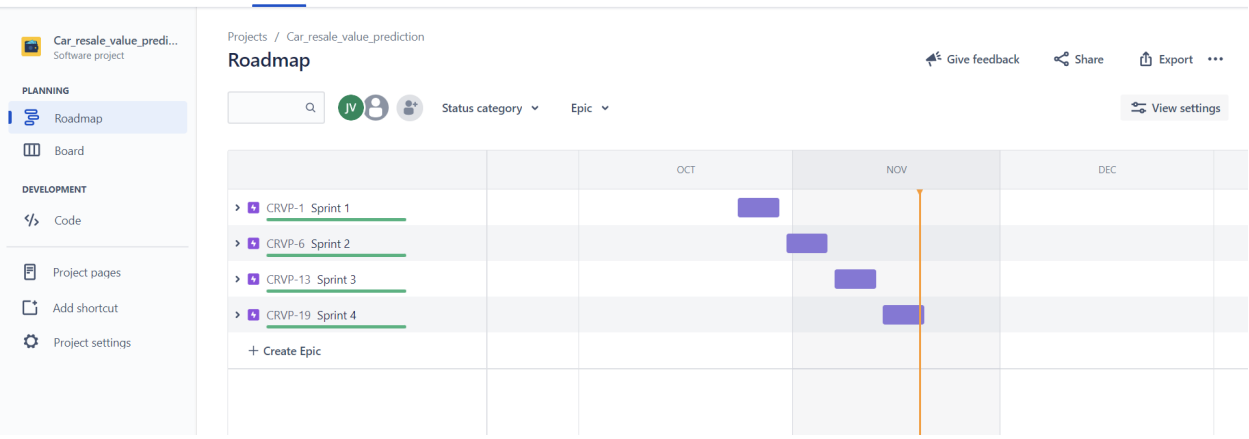
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirming my password.	2	High	Karthik, Jeevana
Sprint-1	Registration	USN-2	As a user, I will receive confirmation email onceI have registered for the application	1	High	Karthik, Karthick sriram
Sprint-2	Registration	USN-3	As a user,I can register for the applicationthrough Facebook	2	Low	Karthik, Jayapriyaa
Sprint-1	Registration	USN-4	As a user,I can register for the applicationthrough Gmail	2	Medium	Karthik, Harish kumar
Sprint-1	Login	USN-5	As a user,I can log intothe application byentering email & password	1	High	Karthik
Sprint-2	Dashboard	USN-6	Users can get to know aboutall the functionalities of the system	2	High	Harish Kumar
Sprint-2	View feedback	USN-7	Users can view others' feedback	1	Low	Jayariyaa
Sprint-2	Provider feedback	USN-8	Users can enter theirfeedback and ratings	2	High	Jayapriyaa
Sprint-3	Predict resalevalue	USN-9	Users can enter all the valuesof the attribute to predict the resale value	4	High	Jeevana
Sprint-4	Query Handling	USN-10	Executive can handlequeries from theusers	2	Medium	Jeevana, Jayapriyaa
Sprint-4	Administration	USN-11	Administrators can solvethe database relatedissues	2	Medium	Karthick sriram

6.2 Sprint Delivery Schedule:

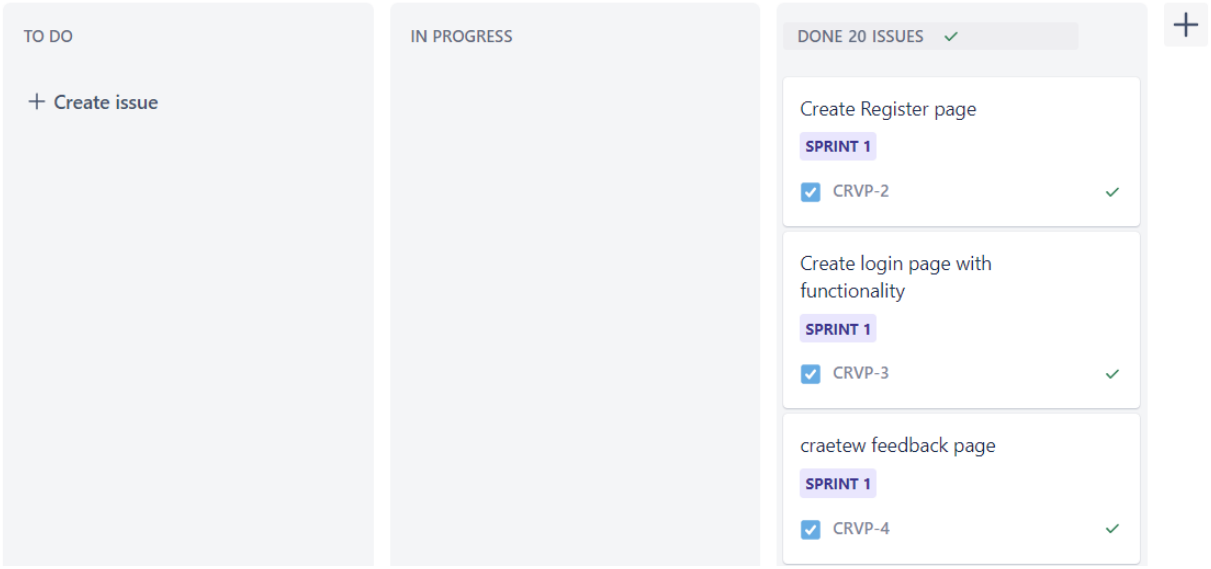
Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on PlannedEnd Date)	Sprint Release Date (Actual)
Sprint-1	20	6 Days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint-2	20	6 Days	31 Oct 2022	05 Nov 2022	20	05 Nov 2022
Sprint-3	20	6 Days	07 Nov 2022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

6.3 Reports from JIRA:

Roadmap

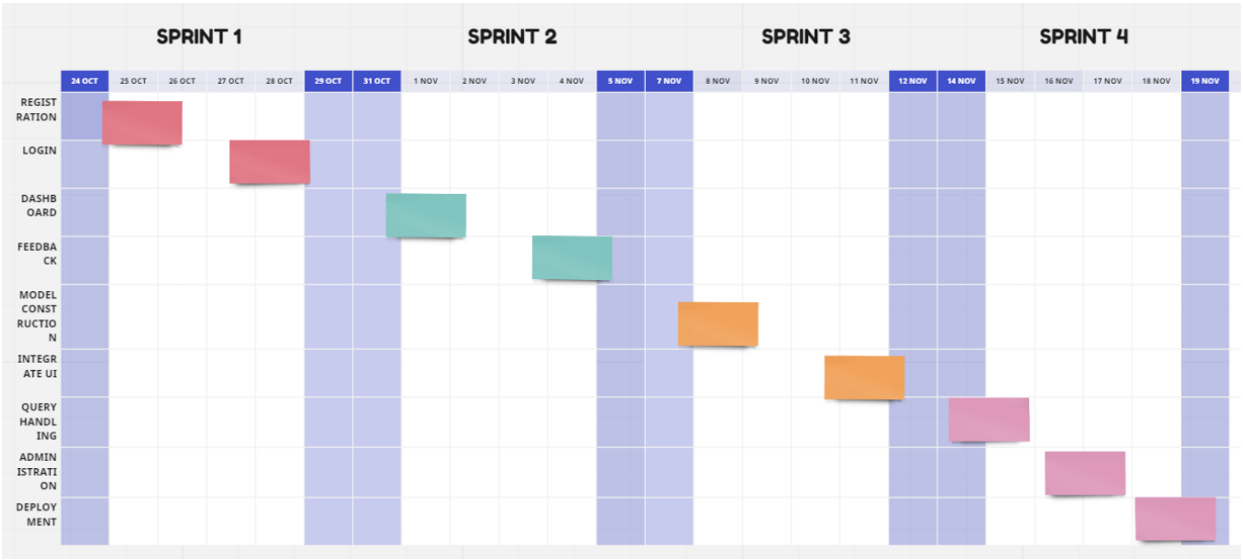


CRVP Board



Burndown Chart:

A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.



7. CODING AND SOLUTIONING

7.1 Register ang Login

Registration allows the user to create new account and enjoys the functionality of the website. It is mandatory for everyone to create their own account before getting their predicted resale value of the car. The user's name, email and password is collected from the user for registering them into the website. Once the users successfully got registered, they could login using their email and password.

app.py > registeruser

```
74
75  @app.route('/register', methods = ["GET"])
76  def register():
77      return render_template('Register.html')
78
79  @app.route('/login', methods = ["GET"])
80  def login():
81      return render_template('Login.html')
82
83  @app.route('/registeruser', methods = ["POST","GET"])
84  def registeruser():
85      msg=""
86      if request.method == "POST":
87          name = request.form["username"]
88          email = request.form["email"]
89          password = request.form["userpassword"]
90          hashed_password = generate_password_hash(password, method='sha256')
91          new_user = User(username=name, email=email, userpassword=hashed_password)
92          db.session.add(new_user)
93          db.session.commit()
94          # msg=DBHelper.save(name, email, hashed_password)
95          msg="New User Registered"
96          return jsonify(result=msg)
97      else:
98          return jsonify(result="Error")
```

```
99
100 @app.route('/loginuser', methods = ["POST"])
101 def loginuser():
102     if request.method == "POST":
103         useremail = request.form["useremail"]
104         userpassword = request.form["userpassword"]
105         user = User.query.filter_by(email=useremail).first()
106         if user:
107             if check_password_hash(user.userpassword, userpassword):
108                 if request.form["remember"]=='':
109                     rem = True
110                 else:
111                     rem = False
112                 login_user(user, remember=rem)
113                 return jsonify(result="Login Successfull")
114             return jsonify(result="Invalid Password")
115         else:
116             return jsonify(result="Invalid User Emaiil")
117     else:
118         return jsonify(result="Error")
119
```

7.2 Resale Value Prediction

This is the important feature of the website. Here user could enter their respective value for the features and get their predicted resale value. The user needs to provide cars_name, cars_brand, model, model_year, car_type, kms, owner, gosoliene_type, city and state. These values passed to the trained model on ibm. The response from the model through API will be displayed as the predicted output. After getting the response, the input value and output will get stored in the Resale model (database).

app.py > registeruser

```python
@app.route('/predict', methods=['GET', 'POST'])
@login_required
def predict():
    if request.method == 'POST':
        pricer=predictprice(request)
        ypred = ' ₹ {:,.2f}'.format(pricer)
        return jsonify(result=ypred)
    else:
        return render_template('Predict.html')

def predictprice(request):

    cars_name = request.form.get('brand')
    cars_brand = cars_name.split(' ')[0]
    model = request.form.get('modeltype')
    model_year = int(request.form.get('regyear'))
    car_type = request.form['gearbox']
    kms = float(request.form['kms'])
    owner = float(request.form['owner'])
    gasoliene_type = request.form['fuel']
    city = request.form.get('cityname')
    state = request.form.get('statename')

    new_row = {'cars_name':[cars_name], 'cars_brand':[cars_brand], 'model':[model], 'model_year':[model_year],
            'car_type':[car_type], 'kms':[kms], 'owner':[owner], 'gasoliene_type':[gasoliene_type],
            'city':[city], 'state':[state]}

    res_row = {'cars_name':cars_name,   'cars_brand':cars_brand, 'model':model, 'model_year':model_year,
            'car_type':car_type, 'kms':kms, 'owner':owner, 'gasoliene_type':gasoliene_type,
            'city':city, 'state':state}

   print(new_row)
   print("hi")
   #new_df = pd.DataFrame(new_row)
   #print(new_df)

   new_df = pd.DataFrame(new_row)
   print(new_df)
   # Encode the user inputs using the stored encoded data
   labels = ['cars_name', 'cars_brand', 'model', 'model_year', 'gasoliene_type', 'car_type',  'city', 'state']
   for i in labels:
       print("hello")
       label = LabelEncoder()
       label.classes_=np.load('Data/'+str('classes'+i+'.npy'), allow_pickle=True)
       tr = label.transform(new_df[i])
       new_df[i]=tr
       #tr = label.transform(new_row[i])
       res_row[i]=int(tr[0])

   print(res_row)

   field = [list(res_row.keys())]
   value = [list(res_row.values())]
   #t = [[cars_name, cars_brand, model_year, car_type, kms, owner, gasoliene_type, city, state]]
   payload_scoring = {"input_data": [{"fields": field, "values": value}]}

   response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/5664a664-73fe-4670-b0f5-c6dbe2bf152e/predictions?
headers={'Authorization': 'Bearer ' + mltoken})
   print("Scoring response")
```

```
    print(response_scoring.json())


    pred = response_scoring.json()
    print(response_scoring.json())
    #y_prediction = model_rand.predict(new_df)
    y_prediction = pred['predictions'][0]['values'][0][0]
    print(y_prediction)

    # Inserting the new search data and results
    newsearch =  ResaleQuery(cars_name = cars_name, cars_brand = cars_brand,model = model, model_year = model_year,
                             car_type = car_type,kms = kms,owner = owner,gasoliene_type = gasoliene_type,
                             city = city,state = state,price = y_prediction)

    db.session.add(newsearch)
    db.session.commit()
    return y_prediction

```

7.3 Providing Feedback

This helps the user in providing the feedback of the website. The model or user interfcae could be modified based on the feedback from the user. This really plays an crucial role. Then the feedback would be recorded in the feedback model (database).

```
@app.route('/feedback', methods = ["POST"])
@login_required
def feedback():
    feedback = request.form.get('feedback')
    comment = request.form.get('comment')
    newfeedback = Feedback(username=current_user.username.upper(),rating=feedback, comment=comment)
    db.session.add(newfeedback)
    db.session.commit()
    return jsonify(result="Thanks")
```

7.4 Viewing Feedback

Here it helps in retrieving the records from the feedback. One could view the feedback provided by other. This also helps them when facing some issues with the website. The solution and problem addressed in the feedback helps in improving features.

```
@app.route('/feedbacklist')
@login_required
def feedbacklist():
    feedbacks = Feedback.query
    return render_template('feedback.html', feedbacks=feedbacks)

```

7.5 Viewing History

This displays the overall history of the website from the Resale model (database). The user could find other users who are interested in selling the car. They could contact others. This also helps in getting to know resale value of the other cars. Analysis can be peformed based on this. The company which owns best resale value and significance of each features of the car could be analyzed.

```
136  @app.route('/dashboard')
137  @login_required
138  def dashboard():
139      formdata = { 'name':current_user.username, 'sourcecount':'3592',
140                   'usercount': db.session.query(User.id).count(),
141                   'feedbackcount':db.session.query(Feedback.id).count(),
142                   'searchcount':db.session.query(ResaleQuery.id).count() }
143      resalequeries = ResaleQuery.query
144      return render_template('dashboard.html', data=formdata, enquiries=resalequeries)
145
```

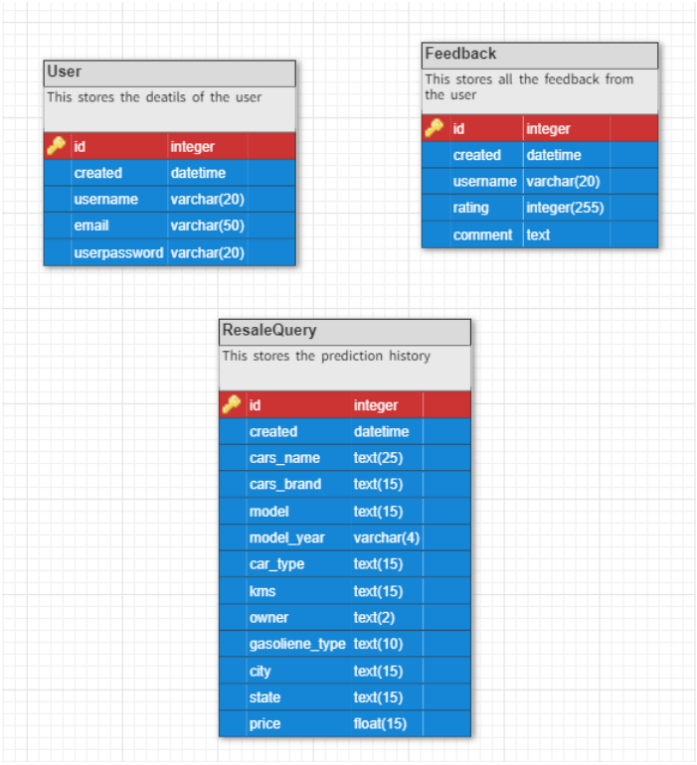
7.6 Database Schema

Here three models is used. One for storing user details, another for storing the history of predictions made and the last one for storing the feedback from the user.

- User - It stores the details of all registered user. It helps during the process of login and registration.

- Feedback - It stores feedback collected from all the user. This plays a role during providing feedback and viewing feedback feature.

- ResaleQuery - It stores the details of the prediction made. This plays a role during predicting and displaying history.



app.py > User

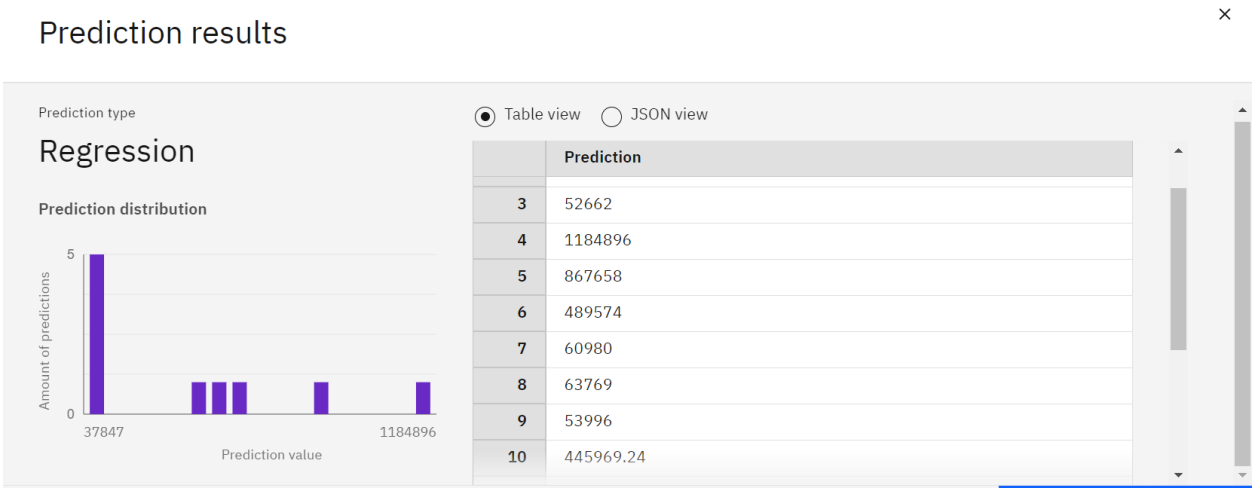
```python
class User(UserMixin, db.Model):
    __tablename__ = 'users'
    id = db.Column(db.Integer, primary_key=True )
    created = db.Column(db.DateTime(timezone=False), server_default=func.now())
    username = db.Column(db.String(15), unique=True)
    email = db.Column(db.String(50), unique=True)
    userpassword = db.Column(db.String(80))
class Feedback(db.Model):
    __tablename__ = 'feedback'
    id = db.Column(db.Integer, primary_key=True )
    created = db.Column(db.DateTime(timezone=False), server_default=func.now())
    username = db.Column(db.String(15))
    rating = db.Column(db.Integer)
    comment = db.Column(db.String(1000))
class ResaleQuery(db.Model):
    __tablename__ = 'resalequery'
    id = db.Column(db.Integer, primary_key=True )
    created = db.Column(db.DateTime(timezone=False), server_default=func.now())
    cars_name = db.Column(db.String(25))
    cars_brand = db.Column(db.String(15))
    model = db.Column(db.String(15))
    model_year = db.Column(db.String(4))
    car_type = db.Column(db.String(15))
    kms = db.Column(db.String(15))
    owner = db.Column(db.String(2))
    gasoliene_type = db.Column(db.String(10))
    city = db.Column(db.String(15))
    state = db.Column(db.String(15))
    price = db.Column(db.Float)
```

8.TESTING

8.1 Test Cases

The tescase is basically performed after splitting the dataset into training and testing. The trained model is tested with the testing dataset. Based on r2_score from all the models RandomForestRegressor is choosen to be the best one.

Few test cases is passed and got the results



The below image dispalys the r2_score for both training and testing the model.

```
regressor_rf = RandomForestRegressor()
regressor_rf.fit(x_train,y_train)
y_train_pred = regressor_rf.predict(x_train)
y_test_pred = regressor_rf.predict(x_test)

mse_rf = metrics.mean_squared_error(y_test,y_test_pred)
rmse_rf = np.sqrt(metrics.mean_squared_error(y_test,y_test_pred))
print('MSE: ',mse_rf)
print('RMSE: ',rmse_rf)

r2_train_rf = r2_score(y_train,y_train_pred)
print('train_r2_score: ',r2_train_rf)
r2_test_rf = r2_score(y_test,y_test_pred)
print('test_r2_score: ',r2_test_rf)
```

```
MSE:  9625128855.392355
RMSE:  98107.74105743316
train_r2_score:  0.9871007059467278
test_r2_score:  0.9241981018252814
```

8.2 User Acceptance Testing

Test Cases Report

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)	BUG ID	Executed By
Register_TC_001	Functional	Home Page	Verify user is able to see the Register New User popup when user clicked on Register New User button	Email id	1.Enter URL and click go 2.Click on Register New User button 3.Verify Register New User popup displayed or not	http://localhost:5000/register	Register New User popup should display	Working as expected	Pass				
Register_TC_002	UI	Home Page	Verify the UI elements in Login/Signup popup	Email	1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup with below UI elements: your name, email, password and repeat password	http://localhost:5000/register	Application should contain UI elements like name, email, password and repeat password.	Working as expected	Pass				
LoginPage_TC_003	Functional	Home Page	Verify the UI elements in Login/Signup popup	Email id, password	1.Enter URL and click go 2.Click on Register New User button 3.Verify Register New User popup displayed or not	https://localhost:5000/login	Sign In will pop up	Working as expected	Pass				
LoginPage_TC_004	UI	Home Page	Verify the UI elements in Login/Signup popup	Email id, password	1.Enter URL and click go 2.Click on My Account dropdown button 3.Verify login/Singup popup with below UI elements: a.email text box b.password text box c.Remember password check icon	https://localhost:5000/login	Application should show below UI elements: email text box b.password text box c.Remember password check icon	Working as expected	Pass				
LoginPage_TC_005	Functional	Home Page	Verify user is able to log into application with Valid credentials	Email id, password	1.Enter URL(https://localhost:5000/login) and click go 3.Enter Valid email in Email text box 4.Enter valid password in password text box and check the remember password box. 5.Click on login button	Username: test@gmail.com password: Testing123	User should navigate to dashboard	Working as expected	Pass				
LoginPage_TC_006	Functional	Login page	Verify user is able to log into application with InValid credentials	Email id, password	1.Enter URL(https://localhost:5000/login) and click go 3.Enter Valid email in Email text box 4.Enter valid password in password text box and check the remember password box. 5.Click on login button	Username: test@gmail password: Testin	Application should show 'login failed'.	Working as expected	Pass				
Dashboard_TC_007	UI	Dashboard	Verify whether user can able to see history of predictions, predict button, search box, feedback button, source data section, enquiry section, feedback section, registered user section	Login is essential	1.Enter URL(https://localhost:5000/dashboard) and click go. You could see the following features: History of predictions, predict button, search box, feedback button, source data section, enquiry section, feedback section, registered user section	https://localhost:5000/dashboard	Application should displaj UI with following components history of predictions, predict button, search box, feedback button, source data section, enquiry section, feedback section, registered user section	Working as expected	Pass				
Feedbacklist_TC_008	UI	Feedbacklist	Verify whether user can see all the feedbacks from all the users	Login is essential	1.Enter URL(https://localhost:5000/feedbacklist) and click go. You could see all the feedback of all the users.	https://localhost:5000/feedbacklist	Application displayed all the feedbacks	Working as expected	Pass				
Feedback_TC_009	Functional	Feedback	Verify whether UI dispalys upon clicking feedback button	Login is essential	1.Enter URL(https://localhost:5000/predict) and click go. Click on feedback button.	http://localhost:5000/predict	On clicking feedback button user can view the feedback form	Working as expected	Pass				
Feedback_TC_010	UI	Feedback	Verify whether user can see UI with rating and comment box component	Login is essential	1.Enter URL(https://localhost:5000/predict) and click go. You could UI with rating and comment box.	http://localhost:5000/predict	Application dispalays feedback section with rating and comment box	Working as expected	Pass				
History_TC_011	UI	History	Verify whether user can see all the prediction history in the dashboard	Login is essential	1.Enter URL(https://localhost:5000/dashboard) and click go. At bottom user can see the history	https://localhost:5000/dashboard	On navigating to dashboard user can view prediction history	Working as expected	Pass				
Predict_TC_012	Functional	Predict	Verify whether upon clicking predict button user can view the predict form	Login and details of the car is essential	Enter URL(http://localhost:5000/predict) and click go. Verify whether form displays or not.	http://localhost:5000/predict	On clicking predict button predict form is dispalyed	Working as expected	Pass				
Predict_TC_013	Functional	Predict	On entering all the fields in predict form whether it provides predict result	Login and details of the car is essential	Enter URL(http://localhost:5000/predict) and click go. Enter all the fields. Click entry	Car brand: AUDI A4, model: 2.0 TDI PREMIUM PLUS, Model year: 2018, owner hand:1, kilometers Driver: 50000, car type: manual, car fuel type: diesel, state: harjana, city:	On entering all the fields and clicking predict button, user should get desired predicted value.	Working as expected	Pass				
Predict_TC_014	Functional	Predict	On missing out any of the fields in predict form whether it provides predict result	Login and details of the car is essential	Enter URL(http://localhost:5000/predict) and click go. Enter few fields. Click entry	Car brand: AUDI A4, model: 2.0 TDI PREMIUM PLUS, Model year: 2018, owner hand:1, kilometers Driver: , car type: manual, car fuel type: diesel, state: harjana, city: gurgaon.	On enterin few of the fields and clicking predict button, user should get failed prompt.	Working as expected	Pass				
Predict_TC_015	UI	Predict	On clicking the predict button on dashboard whether the predict form displays or not	Login and details of the car is essential	Enter URL(http://localhost:5000/predict) and click go. The UI with brand, model, year, owner, km, type,state and city component must be visible	http://localhost:5000/predict	On clicking the URL, predict form with brand, model, year, owner, km, type,state and city componen is visible	Working as expected	Pass				
Logout_TC_016	Functional	Logout	On clicking logout whether user is redirected to home page or not	Login is essential	Enter URL (http://localhost:5000/dashboard) and click got. Click logout button.	http://localhost:5000/dashboard	On clicking logout button in the dashboard, the user redirected to the home page.	Working as expected	Pass				

Defect Aanalysis

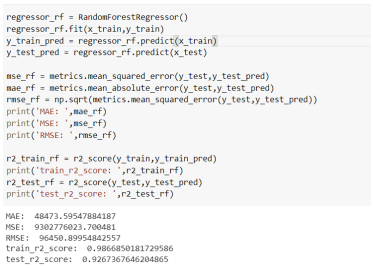
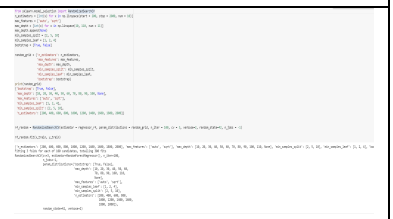
Resolution	Severity 1	Severity 2	Severity 3	Severity 4	Subtotal
By Design	5	2	1	0	8
Duplicate	1	0	0	1	2
External	2	0	1	0	3
Fixed	7	4	2	3	16
Not Reproduced	0	1	0	1	2
Skipped	1	0	0	1	2
Won't Fix	0	1	0	0	1
Totals	16	8	4	6	34

Test Case Analysis

Section	Total Cases	Not Tested	Fail	Pass
Login	4	0	0	4
Register	2	0	0	2
Dashboard	1	0	0	1
Predict	4	0	0	4
History	1	0	0	1
Feedbacklist	1	0	0	1
Feedback	2	0	0	2
Logout	1	0	0	1

9. RESULTS

9.1 Performance Metrics

S.No.	Parameter	Values	Screenshot
1.	Metrics	**Regression Model:** MAE - 48473.59547884187, MSE - 9302776023.700481, RMSE - 96450.89954842557, R2 score – Training - 0.9866850181729586 R2 score – Training – 0.9267367646204865	
2.	Tune the Model	Hyperparameter Tuning – n_estimators = [200, 400, 600, 800, 1000, 1200, 1400, 1600, 1800, 2000], max_features = ['auto', 'sqrt'], max_depth = [10, 20, 30, 40, 50, 60, 70, 80, 90, 100, None], min_samples_split= [2, 5, 10], min_samples_leaf = [1, 2, 4], bootstrap =[True, False] Validation Method - Cross Validation	

10. ADVANTAGES & DISADVANTAGES

- The advantages of the system include predicting the most accurate value, can be used for analysis, ease for user to navigate and make predictions via website and at last feedback help in resolving queries and improving features of the current website.
- The disadvantages of the system is meant in terms of time complexity. The model takes few seconds for making predictions.

11 CONCLUSION

The increased prices of new cars and the financial incapability of the customers to buy them, Used Car sales are on a global increase. Therefore, there is an urgent need for a Used Car Price Prediction system which effectively determines the worthiness of the car using a variety of features. The proposed system will help to determine the accurate price of used car price prediction

12 FUTURE SCOPE

In future this machine learning model may bind with various website which can provide real time data for price prediction. Also we may add large historical data of car price which can help to improve accuracy of the machine learning model. We can build an android app as user interface for interacting with user. For better performance, we plan to judiciously design deep learning network structures, use adaptive learning rates and train on clusters of data rather than the whole dataset

13 APPENDIX

SOURCE CODE

app.py

```
from flask import Flask, render_template, redirect, Response,  url_for, request, jsonify
from flask_bootstrap import Bootstrap
# from flask_wtf import FlaskForm
# from wtforms import StringField, PasswordField, BooleanField
# from wtforms.validators import InputRequired, Email, Length
from flask_sqlalchemy import SQLAlchemy
from sqlalchemy.sql import func
from werkzeug.security import generate_password_hash, check_password_hash
from flask_login import LoginManager, UserMixin, login_user, login_required, logout_user, current_user
from sklearn.preprocessing import LabelEncoder
import pandas as pd
import numpy as np
import os

import requests

# NOTE: you must manually set API_KEY below using information retrieved from your IBM Cloud account.
API_KEY = "C6cCIcP5VNHJr8YrK_X9dcCqNtb-IDwe6t0BHtU3qPJo"
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}



app = Flask(__name__)
app.config['SECRET_KEY'] = 'Thisisatopsecretmissiontopredictcarresalevalues!'
app.config['SQLALCHEMY_DATABASE_URI'] = 'sqlite:///Data/ResaleDB.db'
```

```
app.config['SQLALCHEMY_TRACK_MODIFICATIONS'] = 'False'
bootstrap = Bootstrap(app)
db = SQLAlchemy(app)
login_manager = LoginManager()
login_manager.init_app(app)
login_manager.login_view = 'login'

class User(UserMixin, db.Model):
    __tablename__ = 'users'
    id = db.Column(db.Integer, primary_key=True )
    created = db.Column(db.DateTime(timezone=False), server_default=func.now())
    username = db.Column(db.String(15), unique=True)
    email = db.Column(db.String(50), unique=True)
    userpassword = db.Column(db.String(80))
class Feedback(db.Model):
    __tablename__ = 'feedback'
    id = db.Column(db.Integer, primary_key=True )
    created = db.Column(db.DateTime(timezone=False), server_default=func.now())
    username = db.Column(db.String(15))
    rating = db.Column(db.Integer)
    comment = db.Column(db.String(1000))
class ResaleQuery(db.Model):
    __tablename__ = 'resalequery'
    id = db.Column(db.Integer, primary_key=True )
    created = db.Column(db.DateTime(timezone=False), server_default=func.now())
    cars_name = db.Column(db.String(25))
    cars_brand = db.Column(db.String(15))
    model = db.Column(db.String(15))
    model_year = db.Column(db.String(4))
    car_type = db.Column(db.String(15))
    kms = db.Column(db.String(15))
    owner = db.Column(db.String(2))
    gasoliene_type = db.Column(db.String(10))
    city = db.Column(db.String(15))
    state = db.Column(db.String(15))
    price = db.Column(db.Float)
```

```
@login_manager.user_loader
def load_user(user_id):
    return User.query.get(int(user_id))

@app.route('/')
def index():
    return render_template('Index.html')

@app.route('/register', methods = ["GET"])
def register():
    return render_template('Register.html')

@app.route('/login', methods = ["GET"])
def login():
    return render_template('Login.html')

@app.route('/registeruser', methods = ["POST","GET"])
def registeruser():
    msg=""
    if request.method == "POST":
        name = request.form["username"]
        email = request.form["email"]
        password = request.form["userpassword"]
        hashed_password = generate_password_hash(password, method='sha256')
        new_user = User(username=name, email=email, userpassword=hashed_password)
        db.session.add(new_user)
        db.session.commit()
        # msg=DBHelper.save(name, email, hashed_password)
                msg="New User Registered"
        return jsonify(result=msg)
    else:
        return jsonify(result="Error")

@app.route('/loginuser', methods = ["POST"])
def loginuser():
```

```
    if request.method == "POST":
        useremail = request.form["useremail"]
        userpassword = request.form["userpassword"]
        user = User.query.filter_by(email=useremail).first()
        if user:
            if check_password_hash(user.userpassword, userpassword):
                if request.form["remember"]==":
                    rem = True
                else:
                                        rem = False
                login_user(user, remember=rem)
                return jsonify(result="Login Successfull")
            return jsonify(result="Invalid Password")
                 else:
            return jsonify(result="Invalid User Emaiil")
    else:
        return jsonify(result="Error")

@app.route('/logout')
@login_required
def logout():
    logout_user()
    return redirect(url_for('index'))

@app.route('/feedback', methods = ["POST"])
@login_required
def feedback():
    feedback = request.form.get('feedback')
    comment = request.form.get('comment')
    newfeedback = Feedback(username=current_user.username.upper(),rating=feedback, comment=comment)
    db.session.add(newfeedback)
    db.session.commit()
    return jsonify(result="Thanks")

@app.route('/dashboard')
@login_required
```

```
def dashboard():
    formdata = { 'name':current_user.username, 'sourcecount':'3592',
                'usercount': db.session.query(User.id).count(),
                'feedbackcount':db.session.query(Feedback.id).count(),
                'searchcount':db.session.query(ResaleQuery.id).count() }
    resalequeries = ResaleQuery.query
    return render_template('dashboard.html', data=formdata, enquiries=resalequeries)

@app.route('/feedbacklist')
@login_required
def feedbacklist():
    feedbacks = Feedback.query
        return render_template('feedback.html', feedbacks=feedbacks)

@app.route('/predict', methods=['GET', 'POST'])
@login_required
def predict():
    if request.method == 'POST':
        pricer=predictprice(request)
        ypred = ' ₹ {:,.2f}'.format(pricer)
        return jsonify(result=ypred)
    else:
        return render_template('Predict.html')

def predictprice(request):

    cars_name = request.form.get('brand')
    cars_brand = cars_name.split(' ')[0]
    model = request.form.get('modeltype')
    model_year = int(request.form.get('regyear'))
    car_type = request.form['gearbox']
    kms = float(request.form['kms'])
    owner = float(request.form['owner'])
    gasoliene_type = request.form['fuel']
    city = request.form.get('cityname')
    state = request.form.get('statename')
```

```
    new_row = {'cars_name':[cars_name], 'cars_brand':[cars_brand], 'model':[model],
'model_year':[model_year],
            'car_type':[car_type], 'kms':[kms], 'owner':[owner], 'gasoliene_type':[gasoliene_type],
                        'city':[city], 'state':[state]}

    res_row = {'cars_name':cars_name,   'cars_brand':cars_brand, 'model':model, 'model_year':model_year,
            'car_type':car_type, 'kms':kms, 'owner':owner, 'gasoliene_type':gasoliene_type,
            'city':city, 'state':state}

    print(new_row)
         print("hi")
    #new_df = pd.DataFrame(new_row)
    #print(new_df)

    new_df = pd.DataFrame(new_row)
    print(new_df)
    # Encode the user inputs using the stored encoded data
    labels = ['cars_name', 'cars_brand', 'model', 'model_year', 'gasoliene_type', 'car_type',  'city', 'state']
    for i in labels:
        print("hello")
        label = LabelEncoder()
        label.classes_=np.load('Data/'+str('classes'+i+'.npy'), allow_pickle=True)
        tr = label.transform(new_df[i])
        new_df[i]=tr
        #tr = label.transform(new_row[i])
        res_row[i]=int(tr[0])

         print(res_row)

    field = [list(res_row.keys())]
    value = [list(res_row.values())]
    #t = [[cars_name, cars_brand, model_year, car_type, kms, owner, gasoliene_type, city, state]]
    payload_scoring = {"input_data": [{"fields": field, "values": value}]}

    response_scoring = requests.post('https://us-south.ml.cloud.ibm.com/ml/v4/deployments/5664a664-73fe-
```

```
4670-b0f5-c6dbe2bf152e/predictions?version=2022-11-17', json=payload_scoring,
 headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring response")
    print(response_scoring.json())
    pred = response_scoring.json()
    print(response_scoring.json())
    #y_prediction = model_rand.predict(new_df)
    y_prediction = pred['predictions'][0]['values'][0][0]
    print(y_prediction)
        # Inserting the new search data and results
    newsearch =  ResaleQuery(cars_name = cars_name, cars_brand = cars_brand,model = model, model_year =
model_year,
                        car_type = car_type,kms = kms,owner = owner,gasoliene_type = gasoliene_type,
                        city = city,state = state,price = y_prediction)

    db.session.add(newsearch)
    db.session.commit()
    return y_prediction

if  __name__ == '__main__':

    if not os.path.exists("Data/ResaleDB.db"):
        db.create_all()
    print("Starting Flask Application")
    app.config['TEMPLATES_AUTO_RELOAD'] = True
    app.run(host='localhost', debug=False, threaded=False)
    print("Stopping Application")
```

GITHUB LINK

https://github.com/IBM-EPBL/IBM-Project-7324-1658852844

PROJECT DEMO LINK

https://vimeo.com/772196585