

INTELLIGENT VEHICLE DAMAGE ASSESSMENT & COST ESTIMATOR FOR INSURANCE COMPANIES.



TEAM ID:PNT2022TMID10706



Submitted by

P.SUJITHA(4211191021098)

M.SWEDHA(4211191021101)

P.SWETHA(4211191021102)

P.YAMINI(4211191021118)

In partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

IFET COLLEGE OF ENGINEERING

(An Autonomous Institution)

Guided by

Mr.JAYAKUMAR

Assistant Professor,

Department of Computer Science and Engineering,

IFET COLLEGE OF ENGINEERING, VILLUPURAM.



**IFET COLLEGE OF ENGINEERING
(AN AUTONOMOUS INSTITUTION)
VILLUPURAM-605108**

BONAFIDE CERTIFICATE

Certified that this Project Phase-I report on **“INTELLIGENT VEHICLE DAMAGE ASSESSMENT & COST ESTIMATOR FOR INSURANCE COMPANIES”** is a bonafide work of **“SUJITHA.P(4211191021098), SWEDHA.M(421119102101), SWETHA.P(421119102102), YAMINI.P(4211191021118)”** who carried out the project work under my supervision.

SUPERVISOR

HEAD OF THE DEPARTMENT

The project phase-1 report submitted for the viva voice held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

TABLE OF CONTENT

CHAPTER	CONTENT	PAGE NO
1	Introduction 1.1 Project overview 1.2 Purpose	4
2	Literature Survey 2.1 Existing system 2.2 Problem statement	5
3	Ideathon & Proposed system 3.1 Empathy Map Canvas 3.2 Ideation & Brainstorming 3.3 Proposed Solution	8
4	Requirement analysis 4.1 Functional requirements 4.2 Non Functional requirements	9
5	Project Design 5.1 Data flow Diagrams 5.2 Technical architecture	12
6	Project planning 6.1 Sprint planning & estimation 6.2 Sprint delivery & schedule 6.3 Modules	13
7	Testing	16
8	Results	17
9	Advantages & Disadvantages	20
10	Conclusion and future scope	21
11	Appendix	21

CHAPTER-1

INTRODUCTION

1.1 PROJECT OVERVIEW

Nowadays, a lot of money is being wasted in the car insurance business due to leakage claims. Claims leakage Underwriting leakage is characterized as the discrepancy between the actual payment of claims made and the sum that should have been paid if all of the industry's leading practices were applied. Visual examination and testing have been used to may these results. However, they impose delays in the processing of claims. The aim of our project is to build a VGG16 model that can detect the area of damage on a car. The rationale for such a model is that it can be used by insurance companies for faster processing of claims if users can upload pics and the model can assess damage scratch and estimates the cost of damage. This model can also be used by lenders if they are underwriting a car loan, especially for a used car.

1.2 PURPOSE OF THE PROJECT

In today's world, Vehicles are increasing heavily. Because of increasing the vehicles, accidents are very common because the peoples are driving a car very fastly on the road. The people claim the money for repair the car through vehicle insurance when the accident happens. Because of incorrect claims, the company behaves badly and doesn't make payments currently. This happens due to claims leakage, the claims leakage refers to the difference between the amounts secured by the company to the amount that company should have secured based on the claims.

CHAPTER-2

LITERATURE SURVEY

In this literature survey several methods have been proposed for detection of car damage. Srimal et al. [4] proposed a solution which uses 3D Computer Aided Design for the discernment of car damage from the picture, the system only detect damage at edge portion only. Detection of the car damage through CAD software requires some knowledge about the software. S Gontscharov et al [5] ,the proposed system designed by using YOLO(you only look once) algorithm to detect tha car damage, Here the multi sensor data fusion technique is allows to locate the portion of damage more accurately and performs detection faster compared to other algorithms which is fully automatic and doesn't require much human intervention. Phyu Mar Kyu et al [3], the proposed system uses deep learning based algorithm are VGG16 and VGG19 damaged car detection in the real world. This algorithm notice the severity of the damaged car based on the location. Finally the author concludes that L2 regularization work greater. Girish N et al [2], the proposed system uses vehicle damage detection technique depends on transfer learning and mask RCNN, The mask regional convolution neural network determines a damaged car by its position and estimate the depth of the damage. A Neela Madheswari et al [1], the proposed system uses convolution neural network is use to accept that image contains a car damage or not. It take as great opportunities to attempt by classifying the car damage into different classes.

2.1 EXISTING SYSTEM

- In existing they uses 3D Computer Aided Design for the discernment of car damage from the picture, the system only detect damage at edge portion only. Detection of the car damage through CAD software requires some knowledge about the software.
- Used YOLO algorithm to detect the car damage, Here the multi sensor data fusion technique is allows to locate the portion of damage more accurately and performs detection faster compared to other algorithms which is fully automatic and doesn't require much human intervention. This algorithm notice the severity of the damaged car based on the location.
- Used vehicle damage detection technique depends on transfer learning and mask RCNN, The mask regional convolution neural network determines a damaged car by its position and estimate the depth of the damage.

2.2 Problem Statement

Nowadays, a lot of money is being wasted in the car insurance business due to leakage claims. Claims leakage Underwriting leakage is characterized as the discrepancy between the actual payment of claims made and the sum that should have been paid if all of the industry's leading practices were applied. Visual examination and testing have been used to may these results. However, they impose delays in the processing of claims.

Reference: <https://miro.com/templates/customer-problem-statement/>

CHAPTER-3

IDEATION AND PROPOSED SOLUTION

3.1 Empathy Map Curve

An Empathy map is a simple, easy to digest visual that captures knowledge about a user 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 users perspective along with his or her goals and challenges.



3.2 Brainstorm & Idea Prioritization Template

Brainstorming provides a free and open environment that encourages everyone within a team to participate in the creative thinking process that leads to problem solving. Prioritizing volume over value, out-of-the-box ideas are welcome and built upon, and all participants are encouraged to collaborate, helping each other develop a rich amount of creative solutions.

3.3 PROPOSED SOLUTION

Proposed Solution Template:

Project team shall fill the following information in proposed solution template.

S. No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	Insurance firms frequently losses because they are unable to accurately estimate the cost of damaged automobiles and they are unable to calculate the cost of damaged cars precisely, insurance companies regularly incurred losses.
2.	Idea / Solution description	Car damage is automatically identified and classified using Deep Learning and pattern recognition technology.
3.	Novelty / Uniqueness	Automated calculator for the cost of filing an insurance claim.
4.	Social Impact / Customer	Vehicle's damage analysis used to get compensation, submit the created report

	Satisfaction	and Process that saves time and money.
5.	Business Model (Revenue Model)	The Proposed method was implemented using the Convolutional Neural Network feature extraction and damage detection / localization than pre-trained model VGG16.
6.	Scalability of the Solution	It can be used by insurance companies for faster processing of claims and can also be used to underwriting a car loan, especially for a used car.

CHAPTER-4

REQUIREMENT ANALYSIS

4.1 Functional Requirements:

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User Registration	Registration through Form Registration through Gmail Registration through LinkedIn
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	User details	Users are required to register their personal details. like name, age, date of birth, driving license, car number etc.
FR-4	User requirements	The user simply inputs vehicle damage images. The software will instantly generate an accurate reading of the based on the image detection analysis in a readable format familiar to the customer. It compares the information already given and states the defect percentage and cost in that vehicle damage image.

4.2 Non-functional Requirements

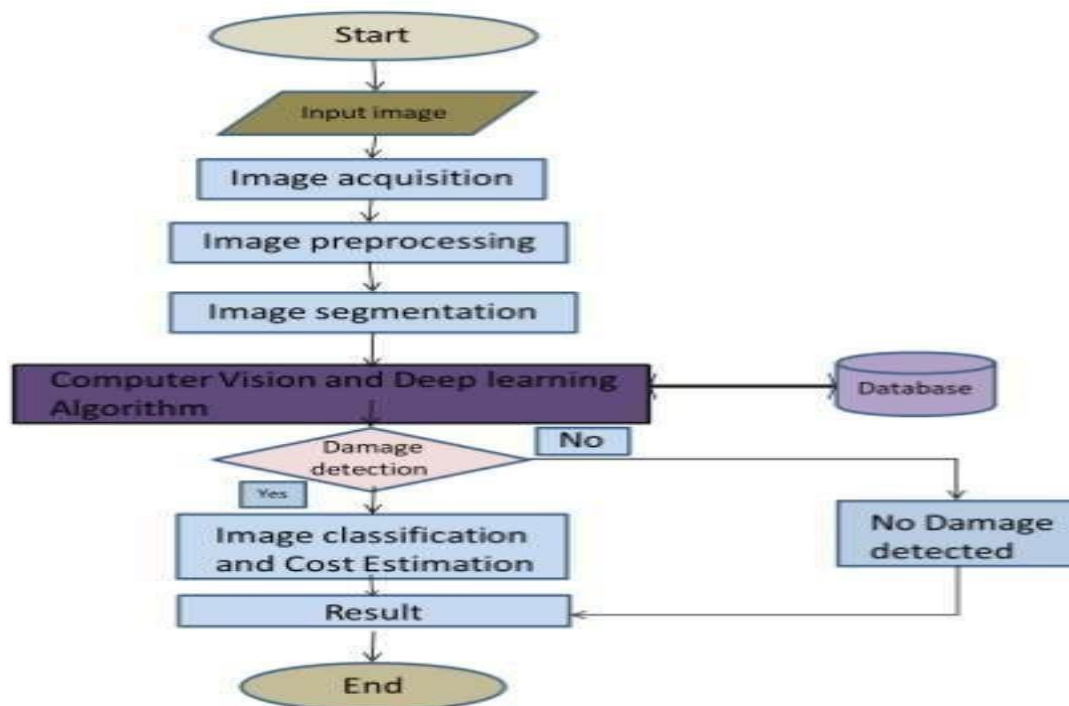
FR No.	Non-Functional Requirement	Description
NFR-1	Usability	More efficient for the frequent users. users can easily understand what the application does and feel satisfied with the system.
NFR-2	Security	<ul style="list-style-type: none">• AI powered vehicle damage assessment and cost estimator for insurance company should contain more security in which our data which entered or maintained should be more security.• With the help of the username and password it provides more security in which it can access more securable and the data are private .
NFR-3	Reliability	This application must perform without failure in 90 percent of use cases during a month.it is more reliable.
NFR-4	Performance	This application supporting 1,050 users per hour must provide 5 seconds or less response time in a desktop browser, including the rendering of text and images, over an LTE connection. The performance of this application is effective and efficient.

NFR-5	Availability	The web dashboard must be available to user's 99.9 percent of the time every month during business hours EST. Users can access anytime and anywhere.
NFR-6	Scalability	The application must be scalable enough to support 10,000 visits at the same time while maintaining optimal performance and efficient to retrieve image in large scale thus improving scalability.

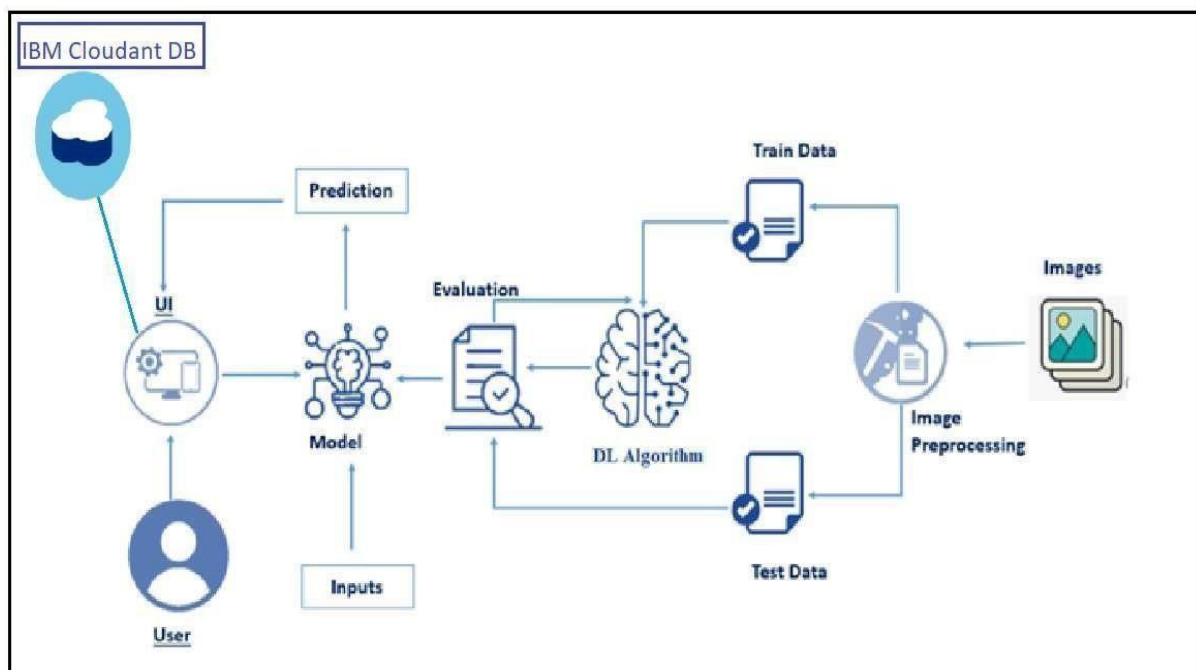
CHAPTER-5

PROJECT DESIGN

5.1 DESIGN FLOW DIAGRAMS



5.2 TECHNICAL ARCHITECTURE



CHAPTER-6

PROJECT PLANNING & DELIVERY SCHEDULE

6.1 Sprint Planning & Estimation

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Priority
Sprint-1	Views Prediction	USN-2	VGG16 Model to train and test dataset of car views like rear,side,front view.	High
Sprint-2	Damage Level Prediction	USN-3	Damage level Image dataset of which severe,mild,high damage.	Low
Sprint-3	Damage Amount Calculation	USN-4	Calculating insurance cost for damaged vehicle	Medium

6.2 Sprint delivery & schedule

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Sprint Release Date (Actual)
Sprint-1	20	2 Days	2 Nov 2022	02 Nov 2022	4 Nov 2022
Sprint-2	20	2 Days	4 Nov 2022	05 Nov 2022	6 Nov 2022
Sprint-3	20	2 Days	6 Nov 2022	12 Nov 2022	8 Nov 2022
Sprint-4	20	4 Days	8 Nov 2022	19 Nov 2022	13 Nov 2022

6.3 MODULES

The dataset folder contains two folders one is for car views and another is for damage level prediction which in turn contains folders for test and train . Each folder has the images for different views and damage level of the car.

The Flask folder has all the files necessary to build the flask application

The static folder has the images, style sheets, and scripts that are needed in building the web page.

- Templates folder has the HTML pages.
- Uploads folder has the uploads made by the user.
- Application.py is the python script for server-side computing.
- .h5 files are the model files that are to be saved after model building.
- Five car companies with their famous model's price and other details are include to find the premium amount.

- The below mentioned are the training and testing notebook.
- Sprint1.ipynb - training and testing of VGG16 model for the car view prediction.
- Sprint2.ipynb- The training and testing of the VGG16 model for the damage level prediction.

IBM folder contains IBM deployment files.

Data Collection:

Create Train and Test folders, each folder having subfolders with car images of different views. We have make use of the image dataset that were posted by Nalaiya Thiran executed by IBM.You can collect datasets from our git hub repository. Two datasets will be used, we will be creating two models one to predict the views of the car image like front, rear and side and second model is to predict the damage level of the car like low, mild and severe.

Image Preprocessing:

Now that we have all the data collected, let us use this data to train the model. Before training the model you have to preprocess the images and then feed them on to the model for training. We make use of Keras Image Data Generator class for image preprocessing.

CHAPTER 7

TESTING

7.1 TEST CASES

Test Case Type	Description	Test Step	Expected Result	Status
Functionality	Engine capacity and car model should belongs to the respected company	Input valid model numbers	Model and capacity in the request should be appropriate	Pass
Security	Verify mobile is valid or not	Enter a 10 digit password in accordance with rule	The user's mobile number will be accepted if it address to the rules	Pass
Usability	Ensure that all links are working properly	Have users click on various links on the page	Links will take users to another web page according to the on-page URL	Pass

CHAPTER 8

RESULTS

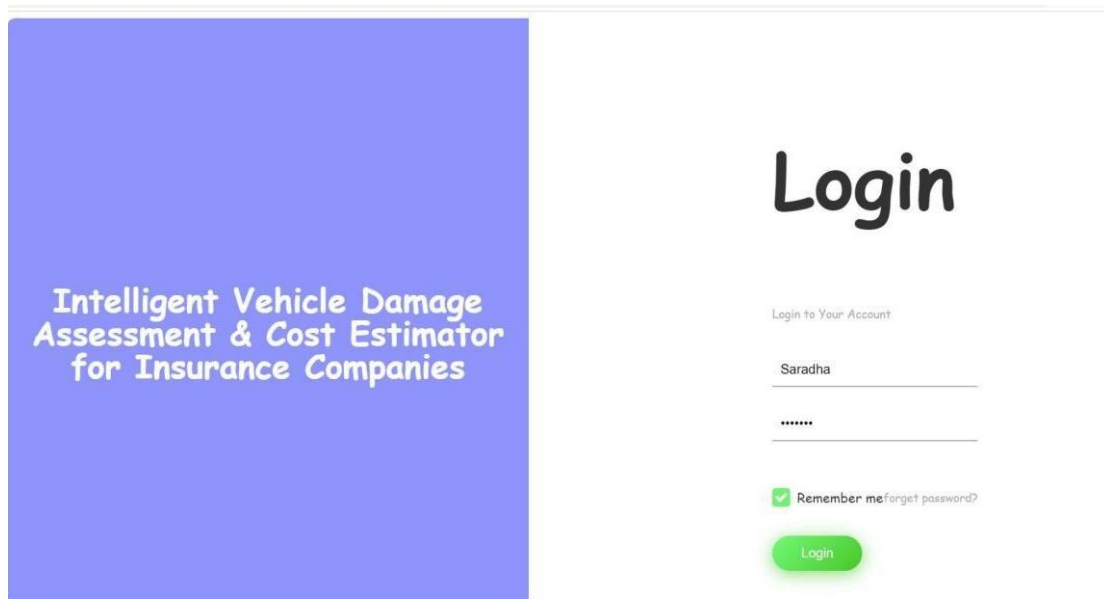


Figure 9.1 Login page



Figure 9.2 Description Login

Fill the details

Saradha
9897666598
renault
kwid
999
petrol

Choose File backside.jpg

Predict

Figure 9.3 Enter Car details and upload image



User details

customer name: Saradha
contact: 9897666598
VERIFIED....

Premium amount

View of the car: rear image
Damage level of the car: low damage
Original price: 292667
Depreciation rate: 20486.690000000002
IDV amount: 272180.31
Premium amount: 6437.491555295999

Figure 9.4 Premium amount calculation

CHAPTER 9

ADVANTAGES & DISADVANTAGES

9.1 Advantages

- Intelligent damage determination system can be used to determine the appearance damage of vehicles in small cases.
- Photographs of target vehicles and multiple trio vehicles were taken and uploaded, intelligent recognition, information input, intelligent recognition and event finalization are completed in accident investigation.
- Damage results including maintenance scheme recommendation and maintenance price recommendation are automatically given according to damage recognition results.

9.2 Disadvantages

- **Coverage failures:** The primary and major disadvantage of car insurance is that your policy does not cover the entire vehicle. Only the specific parts of the car are under damage coverage, the policyholder needs to verify hidden clauses in the document keenly before buying the policy.
- **Time taking Process:** Most insurance companies take a time frame to settle the claim amount, this is the problem most the policyholders are facing.

CHAPTER 10

CONCLUSION

In this project, based on the demand of automobile insurance claims and intelligent transportation, combined with abundant basic data and advanced machine vision algorithm, an intelligent damage determination system of 'Artificial Intelligence + Vehicle Insurance' is constructed. The rapid accumulation of data, the continuous improvement of computing power, the continuous optimization of algorithm models, and the rapid rise of multi-scene applications have made profound changes in the development environment of artificial intelligence. Thus, created a login page to enter the damage vehicle and the model show the result of how much insurance cost want to pay. This help the user to check the amount accurately. This model useful in insurance providing companies.

10.2 FUTURE SCOPE

In future, we will continue to explore the innovation of insurance technology of 'AI + Vehicle Insurance'. We hope that we can use the power of intelligent damage determination system. On the one hand, the owner can take photos by one click to achieve rapid loss determination, price estimation and immediate compensation. On the other hand, it assists insurance companies to achieve rapid and accurate pricing in the process of fixing losses and claims. Finally, by combining the rapid compensation of accident vehicles to relieve traffic pressure, to avoid more serious personal and property losses caused by secondary accidents.

CHAPTER 11

APPENDIX

Building a Model:

```
import tensorflow as tf
import os
import numpy as np
from tensorflow.keras.layers import
Input, Flatten, Dense
from tensorflow.keras.models import Model
    from tensorflow.keras.applications.vgg16
        import VGG16
    from tensorflow.keras.models import
        Sequential
import matplotlib.pyplot as plt
import gradio as gd
from tensorflow.keras.utils import
    array_to_img
    from tensorflow.keras.utils import load_img
    from tensorflow.keras.utils import img_to_array
```

2. Image data generator - data preprocessing

```
In[ ]: IMAGE_SIZE=224 BATCH_SIZE=64
train_datagen=tf.keras.preprocessing.image.Image
DataGenerator(rescale=1./255, zoom_range=0.2,
horizontal_flip=True,
validation_split=0.1) validation_datagen=tf.keras.p
```

```

reprocessing.image.ImageDataGenerator(rescale=
1./255, validation_split=0.1)In[ ]:

train="training"
train_generator=train_datagen.flow_from_directory(
train,
target_size=(IMAGE_SIZE,IMAGE_SIZE),
    batch_size=BATCH_SIZE)
test="validation"
validation_generator=validation_datagen.flow_from_directory(
test,target_size=(IMAGE_SIZE,IMAGE_SIZE),
    batch_size=BATCH_SIZE)
print("Integer values of classes:")
train_generator.class_indices
IMAGE_SIZE=[224,224]
vgg=VGG16(input_shape=IMAGE_SIZE+[3],
weights='imagenet',include_top=False)
vgg.output for layer in
vgg.layers:layer.trainable=False
x=Flatten()(vgg.output)
prediction=Dense(3,activation='softmax')(x)
model=Model(inputs=vgg.input,outputs=prediction)
model.summary()
# 4. Train the model
model.compile(loss='categorical_crossentropy',
optimizer='adam', metrics=['accuracy'])
#

```

```
In[ ]:model.save("train1.h5") fn11='log3.csv'
history_logger=tf.keras.callbacks.CSVLogge
r(fn11,separator=",",append=True) # In[91]:
epoch=10
```

```
history=model.fit(train_genarator,
                  steps_per_epoch=len(train_genarator),
                  epochs=epoch,
                  callbacks=[history_logger],
                  validation_data=validation_generator,
                  validation_steps=len(validation_generator)
plt.plot(history.history["accuracy"])
plt.plot(history.history['val_accuracy'])
plt.plot(history.history['loss'])
plt.plot(history.history['val_loss'])
plt.title("model accuracy")
plt.ylabel("Accuracy") plt.xlabel("Epoch")
plt.legend(["Accuracy","Validation
          Accuracy","loss","Validation Loss"])
plt.show()
```

5.Test the model

```
from tensorflow.keras.utils import load_img from
tensorflow.keras.utils import img_to_array import
numpy as np from tensorflow import keras
model1=keras.models.load_model("train1.h5")
img_pred=load_img("test/frontside.jpg",targe
```



```
t_size=(224,224))plt.imshow(img_pred,
cmap=plt.get_cmap('gray'))
```

```
img_pred=img_to_array(img_pred)img_pred=np.e
xpand_dims(img_pred, axis=0)rslt=
model1.predict(img_pred)print(rslt) print() if
rslt[0][0]>rslt[0][1]:if
rslt[0][2]>rslt[0][0]:prediction="side image"
else:prediction="front image"else:prediction="rear
image"print("VIEW OF THE CAR
IMAGE:")print(prediction)
```

(B)Damage_level_Image_dataset

1.Preprocessing

```
IMAGE_SIZE_damage=224
BATCH_SIZE_damage=32
train_datagen_damage=tf.keras.preprocessing
    g.image.ImageDataGenerator(
        rescale=1./255, zoom_range=0.2,
        horizontal_flip=True,
        validation_split=0.1)
```

```
validation_datagen_damage=tf.keras.preprocessing.image.ImageDataGenerator(
    rescale=1./255, validation_split=0.1
)
```

```
train_damage="training_damage"
train_generator_damage=train_datagen_damage.flow_from_directory(train_damage,target_size=(IMAGE_SIZE_damage,IMAGE_SIZE_damage),
    batch_size=BATCH_SIZE_damage)test_damage="validation_damage"
validation_generator_damage=validation_datagen_damage.flow_from_directory( test_damage,
    target_size=(IMAGE_SIZE_damage,IMAGE_SIZE_damage),
    batch_size=BATCH_SIZE_damage)print("Integer values of classes:")
train_generator_damage.class_indices
```

2.VGG16 model

```
IMAGE_SIZE_damage=[224,224]
vgg_damage=VGG16(input_shape=IMAGE_SIZE_damage+[3],weights='imagenet',include_top=False)
vgg_damage.outputfor layer_d in vgg_damage.layers: layer_d.trainable=False
x_d=Flatten()(vgg_damage.output)
prediction_damage=Dense(3,activation='softmax')(x_d)model_damage=Model(inputs=vgg_da
```

```

model.input,outputs=prediction_damage)
model_damage.summary()
model_damage.compile(loss='categorical_crossentropy', optimizer='adam',
metrics=['accuracy'])model_damage.save("train2.h
5") fn12='log1.csv'
logger=tf.keras.callbacks.CSVLogger
(fn12,separator=",",append=True)

```

3.Train the model

```

epoch_d=7
history_damage=model_damage.fit(train_gen
erator_damage,steps_per_epoch=len(train_generat
or_da mage), epochs=epoch_d,
callbacks=[logger],validation_data=validation_ge
nerator_da
mage,validation_steps=len(validation_generator_d
a image))
plt.plot(history_damage.history["accuracy"])
plt.plot(history_damage.history['val_accurac
y'])plt.plot(history_damage.history['loss'])
plt.plot(history_damage.history['val_loss'])
plt.title("model accuracy") plt.ylabel("Accuracy")
plt.xlabel("Epoch")
plt.legend(["Accuracy","ValidationAccuracy","los
s","Validation Loss"]) plt.show() from
tensorflow.keras.utils importarray_to_imgfrom
tensorflow.keras.utils import load_img from
tensorflow.keras.utils importimg_to_array from

```

```

tensorflow import keras from tensorflow import
keras model2=keras.models.load_model("train2.h")
import numpy as np
img_pred_1=load_img("test/damage2.jpg",ta
rget_size=(224,224))
plt.imshow(img_pred_1,cmap=plt.get_cmap('gray')

```

```

img_pred_1=img_to_array(img_pred_1)
img_pred_1=np.expand_dims(img_pred_1,
axis=0)

```

```

print()
rst=model2.predict(img_pred_1) if
rst[0][0]>rst[0][1]:
    if rst[0][2]>rst[0][0]:
        predicts="low damage" else:
        predicts="mild damage" else:
        predicts="severe damage"

```

```

print(rst)
print()
print("DAMAGE LEVEL:")

```

```

print() print(predicts)

```

```

# # New section --- Test both views and damage
level of the car
#class_view{0:front,1:rear,2:side}

```

```
#class_damage(0:low,1:mild,2:high) #function---
```

```
depreciation and IDV def calcidv(r,v,d):
```

```
    if(d==0):
```

```
        if(v==0):
```

```
            d_dep=0.5*r
```

```
        elif(v==1):
```

```
            d_dep=0.07*r else:
```

```
            d_dep=0.06*r
```

```
    elif(d==1):
```

```
        if(v==0):
```

```
            d_dep=0.12*r
```

```
        elif(v==0):
```

```
            d_dep=0.14*r else:
```

```
            d_dep=0.15*r
```

```
    elif(d==2):
```

```
        if(v==0):
```

```
            d_dep=0.17*r
```

```
        elif(v==1):
```

```
            d_dep=0.18*r else:
```

```
            d_dep=0.20*r
```

```
    print("DEPRECIATION_RATE ",d_dep)
```

```
    idv_idv=r-d_dep
```

```
print("IDV ",idv_idv) return
idv_idv
#funtion-----price def
calculate(c,m,e,f):if(model=
="tata" and
m=="tiago")price=649000
return price
else:
if(f=="cng"): price=296661
return price
```

```
else:
    price=292667
    return price
if(c=="renault" and m=="triber"):
    price=559000
    return price
else:
    if(e==999):
        price=470990
        return price
    else:
        price=413290
        return price
if(c=="dutsan" and m=="go"):
```

```

    price=528464
    return price
else:

    if(e==999):
        price=43765
        return price
    else:

        price=351832
        return price
if(c=="hyundai" and f=="cng"):

    price=547990
    return price
else:

    price=503990
    return price

```

```

#function-----premium amount calculator def
calculator(i):
    print("TOTAL PREMIUM AMOUNT:")
    own_damage=0.01970*i
    ncb_discount=0.2*own_damage
    od_premium=own_damage-ncb_discount
    net_premium=od_premium+100+50+1110
    gst=0.16*net_premium
    premium=gst+net_premium print("premium
    amount",premium) return premium
class_names=["front","rear","side","high","lo

```

```

w","severe"]          def
predict_model(img):#load model from
tensorflow import keras
model3=keras.models.load_model("train1.h5")
from tensorflow import keras
model4=keras.models.load_model("train2.h5")
img=img_to_array(img)
img=np.expand_dims(img, axis=0) result1=
model3.predict(img) return {class_name:}
print("_____") print()
result2=model4.predict(img)

if result2[0][0]>result2[0][1]:
    if result2[0][2]>result2[0][0]:
        predict="severe damage"
        class_damage=2 else:
            predict="mild damage" class_damage=1

else:
    predict="low damage" class_damage=0

return class_views,class_damage
# Premium amount calculation
models=["tiago","nano_genx","triber","
kwid","go","redi_go","santro"]

```



```
dictc={"tata":("tiago","nano
genx"),"renault":("triber","kwid"),"dats
un":("go","redi_go"),"hyundai":("santro")
}
dengine={"tiago":("1199"),"nano":("64
"),"kwid": ("999","799"),
"triber":("999"),
"go":("1198"),"redi":("999","799"),"san
tro":("1086")} #fuel type
cng={"nano_genx","santro",""} #-----
getting input from user-----
```

```
def premium(img,cmp_name,model,engine,f
uel_type):
#-----verification--entered company and other
details were real-----#-----function
calling-----if cmp_name in
dictc.keys(): l=list(dictc[cmp_name]) verify+=1 if
model in l:
    if(dengine[model]=="kwid"):
l_eng=list(dengine[model])
    else:
l_eng=str(dengine[model])
```

```

        verify+=1 if engine in
        l_eng: verify+=1 if
        fuel_type=="cng":
            if model in cng: verify+=1
                print("verified")

rate=calculate(cmp_name,model,engine,f
uel_type)
        print("PRICE OF THE
CAR:",rate)

idv=calcidv(rate,class_view,class_damag e)
premium=calculator(idv)
        else:
            verify+=1 print("")

rate=calculate(cmp_name,model,engine,f
uel_type)
        print("PRICE OF THE CAR:",rate)

idv=calcidv(rate,class_views,class_dama ge)
premium=calculator(idv)
        else: p#-----mainfunction-----

#-----variables-----

```

```

models=["tiago","nano_genx","triber","k
wid","go","redi_go","santro"]
dictc={"tata":("tiago","nano

genx"),"renault":("triber","kwid"),"datsun

":("go","redi_go"),"hyundai":("santro")}
dengine={"tiago":("1199"),"nano":("624"),
"kwid": ("999","799"), "triber":("999"),
"go":("1198"),"redi":("999","799"),"santr
o":("1086")} #fuel type
cng={"nano_genx","santro",""}

#-----getting input from user-----
print("Enter car details:")
cmp_name=input() model=input()
engine=str(input()) fuel_type=input()
verify=0

#-----verification--entered company and
other details were real-----#-----
function calling-----
--

if cmp_name in dictc.keys():
    l=list(dictc[cmp_name])
    verify+=1 if model in l:
        if(dengine[model]=="kwid"):

l_eng=list(dengine[model])

```

```

else:

l_eng=str(dengine[model])

verify+=1 if engine in
l_eng: verify+=1 if
fuel_type=="cng":
    if model in cng: verify+=1
        msg="verified"

rate=calculate(cmp_name,model,engine,f
uel_type)

idv=calcidv(rate,class_view,class_damag e)
premium=calculator(idv) msg = "verified"
return msg

else:

    verify+=1

rate=calculate(cmp_name,model,engine,f
uel_type)

idv=calcidv(rate,class_views,class_dama ge)
    premium=calculator(idv)
    msg="verified" return
    msg

else:

```

```

msg="please enter the valid engine

    capacity" return
    msg else:
msg="please enter the valid model

    name" return
    msg

else:

    msg="sorry!! <<<your car comany

    detail is not available>>>"

    return msg

```

FRONT END-PREDICT THE MODEL:

```

<!DOCTYPE html>

<head>

    <!-- CSS only -->

    <link
href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/css/bootstrap.min.css"
rel="stylesheet" integrity="sha384-
Zenh87qX5JnK2Jl0vWa8Ck2rdkQ2Bzep5IDxbcnCeuOxjzrPF/et3URy9Bv1W
TRi" crossorigin="anonymous">

    <!-- JavaScript Bundle with Popper -->

    <script
src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.2/dist/js/bootstrap.bundle.min.j
s" integrity="sha384-
OERcA2EqjJCMA+/3y+gxIOqMEjwtxJY7qPCqsdltbNJuaOe923+mo//f6V8Qb
sw3" crossorigin="anonymous"></script>

    <link rel="stylesheet" href="./style.css">

</head>

```

```

<body style="background-color: white;">
    <div>
        <nav class="navbar navbar-expand-lg navbar-light" style="background-color:
lightblue;">
            <a class="navbar-brand" href="#" style="margin-left: 60px; color:black;
">Intelligent Vehicle Damage Assessment & Cost Estimator for Insurance
Companies</a>
            <button class="navbar-toggler" type="button" data-toggle="collapse"
datatarget="#navbarNavAltMarkup" aria-controls="navbarNavAltMarkup"
ariaexpanded="false" aria-label="Toggle navigation">
                <span class="navbar-toggler-icon"></span>
            </button>
            <div class="collapse navbar-collapse" id="navbarNavAltMarkup">
                <div class="navbar-nav">
                    <a class="nav-item nav-link" href="login.html" style="margin-left:
350px;">Home <span class="sr-only"></span></a>
                    <a class="nav-item nav-link" href="predict.html">Predict</a>

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

    <br><br><br><br>

    <head>
        <link class="jsbin"
href="http://ajax.googleapis.com/ajax/libs/jqueryui/1/themes/base/jquery-ui.css"
rel="stylesheet" type="text/css" />
        <script class="jsbin"
src="http://ajax.googleapis.com/ajax/libs/jquery/1/jquery.min.js"></script>

```

```
<script class="jsbin"
src="http://ajax.googleapis.com/ajax/libs/jqueryui/1.8.0/jqueryui.min.js"></scri
pt>
```

```
<meta charset=utf-8 />
```

```
<script src="predict.js"></script>
```

```
<title>Predict</title>
```

```
<!--[if IE]>
```

```
<script src="http://html5shiv.googlecode.com/svn/trunk/html5.js"></script>
```

```
<![endif]-->
```

```
<style> article, aside, figure, footer, header,
    hgroup, menu, nav, section { display:
    block; }
```

```
</style>
```

```
</head>
```

```
<div>
```

```
<body class="one">
```

```
<div style="margin-left: 450px; background-image:
'images/backg.jpg' ;width: 600px; height: 500px;" >
```

```
<h4 style="margin-left:200px">Fill the details </h4>
```

```
</br>
```

```
<div class="inputs">
```

```
<input type="text" style="margin-left: 200px;" placeholder="user
name">
```

```
</br>
```

```
<input type="text" style="margin-left: 200px;" placeholder="contact number">
```

```
</br>
```

```
<input type="text" style="margin-left: 200px;" placeholder="Car company">
```

```
</br>
```

```
<input type="text" style="margin-left: 200px;" placeholder="Car model">
```

```
</br>
```

```
<input type="text" style="margin-left: 200px;" placeholder="Engine capacity">
```

```
</br>
```

```
<input type="text" style="margin-left: 200px;" placeholder="Fuel type">
```

```
<input type='file' onchange="readURL(this);" style="margin-left: 200px; margin-top: 60px;">
```

```
</br>
```

```
</br><a href="predicte.html"><button style="margin-left: 250px;">Predict</button></a></div> </body></body> </html>
```