INTELLIGENT VEHICLE DAMAGE ASSESSMENT AND COST ESTIMATOR FOR INSURANCE COMPANIES

A PROJECT REPORT

Submitted by

SWETHA B (923819104049)

SWETHA M (923819104052)

DIVYASRI P (923819104010)

AKILA K (923819104002)

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

MANGAYARKARASI COLLEGE OF ENGINEERING,

MADURAI 625 402

ANNA UNIVERSITY: CHENNAI 600 025

TABLE OF CONTENTS

CHAPTER-1

Introduction

1.1 Project Overview

The Intelligent vehicle damage assessment and cost estimation system is based on domain of Artificial Intelligence and powered by the IBM watson cloud. A responsive web application can be developed using the HTML and CSS which is connected to watson cloud. In the cloud, a database service by availing the service Instance of the IBM cloud and the database API key is collected and connected with the front-end using flash which is an python framework for designing the backend. Pages such as index.html, login.html, logout.html and prediction.html are used to interact with the web application. The user can register and the data of the user is saved in the database of the IBM cloud, during the time of login, the login ID is compared with the ID in the database and allow the user to the next page. The Deep Learning model is build using the VGG16 which is present in the keras library and the model is trained with the images of multiple car with various level cum types of damages. The model is deployed in the back-end using the flask and the prediction.html page is setted to collect the image from the user. The prediction algorithm is used treat the image and estimated the cost for the user. The project is based on the various components which helps to handle the back - end and front - end. Then front - end is build using html and css which is connected back - end which is build using the python and IBM cloud. The image will be detected by Object detection of VGG16 model and displayed in the web application. A web application is developed to visualize the damage and estimate the cost of damage in vehicle.

1.2 Purpose

The main motive of this project is to build a VGG16 model in deep learning and Computer Vision. 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 from and estimates the cost of damage. This model can also be used by lenders if they are underwriting a car loan, especially for used car.

CHAPTER-2

LITERATURE SURVEY

2.1 Existing problem

As the existing system only detect the location of damage and send the report to the insurance company . It takes more time to estimate the cost manually by the insurance company and lead to dissatisfaction of users.

2.2 References

1..Official webpage of the Phillongcollisioncenter at:https://phillongbodyshop.com/collision-repair-cost-calculator.html

2.Official webpage of the Researchgate at:https://www.researchgate.net/publication/310464310_Methodology_o

2.3 Problem Statement Definition

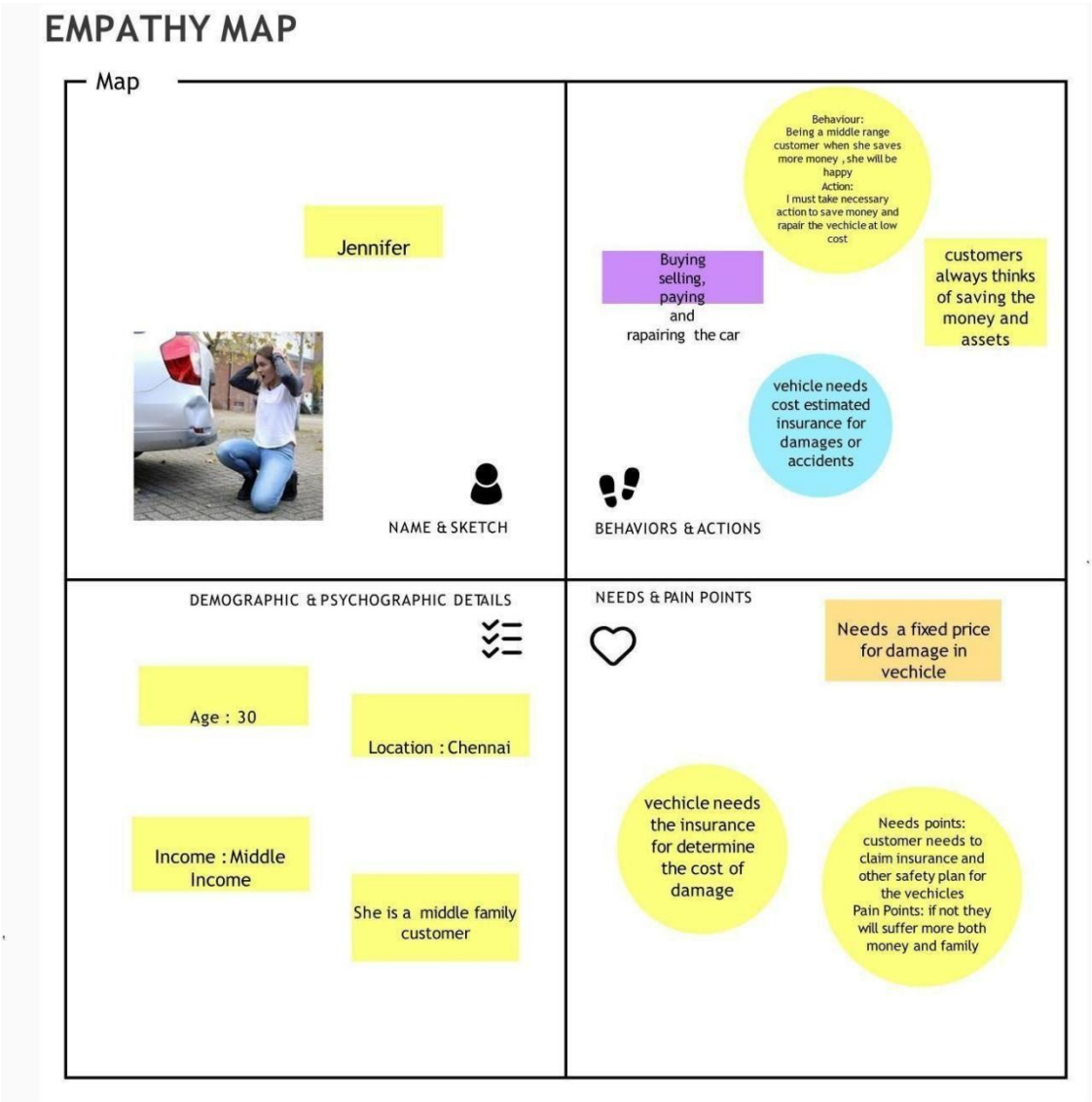
The Automobile section is one of the most important and major sector in India . Major problem faced by the customer on insurance companies are not having idea about the cost of the damage. Insurance Companies are failing to provide right amount for the car damage and the customer not able to claim for the damage. Developing a solution which can able to identify the right cost for the damage would be beneficial for many users.

CHAPTER-3

IDEATION AND PROPOSED SOLUTION

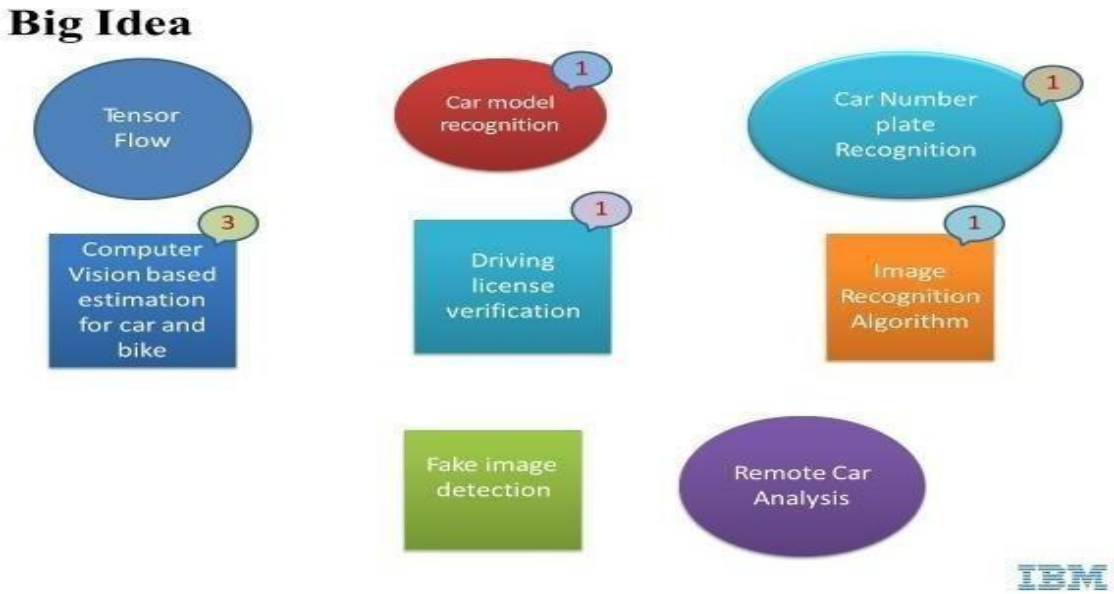
3.1 Empathy Map Canvas

An empathy map is a collaborative tool teams can use to gain a deeper insight into their customers. Much like a user persona, an empathy map can represent a group of users, such as a customer segment.



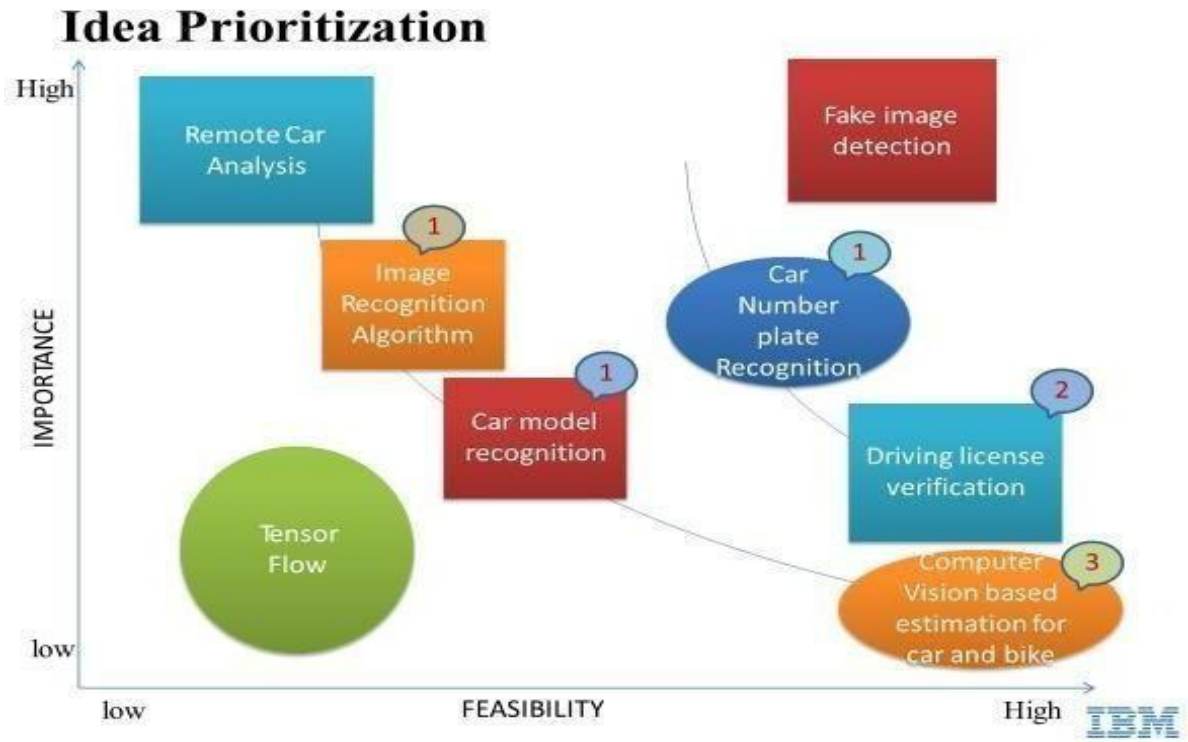
3.2 Big Ideas

It consists of all the ideas of instruments and equipments that we are going to implement in this project.

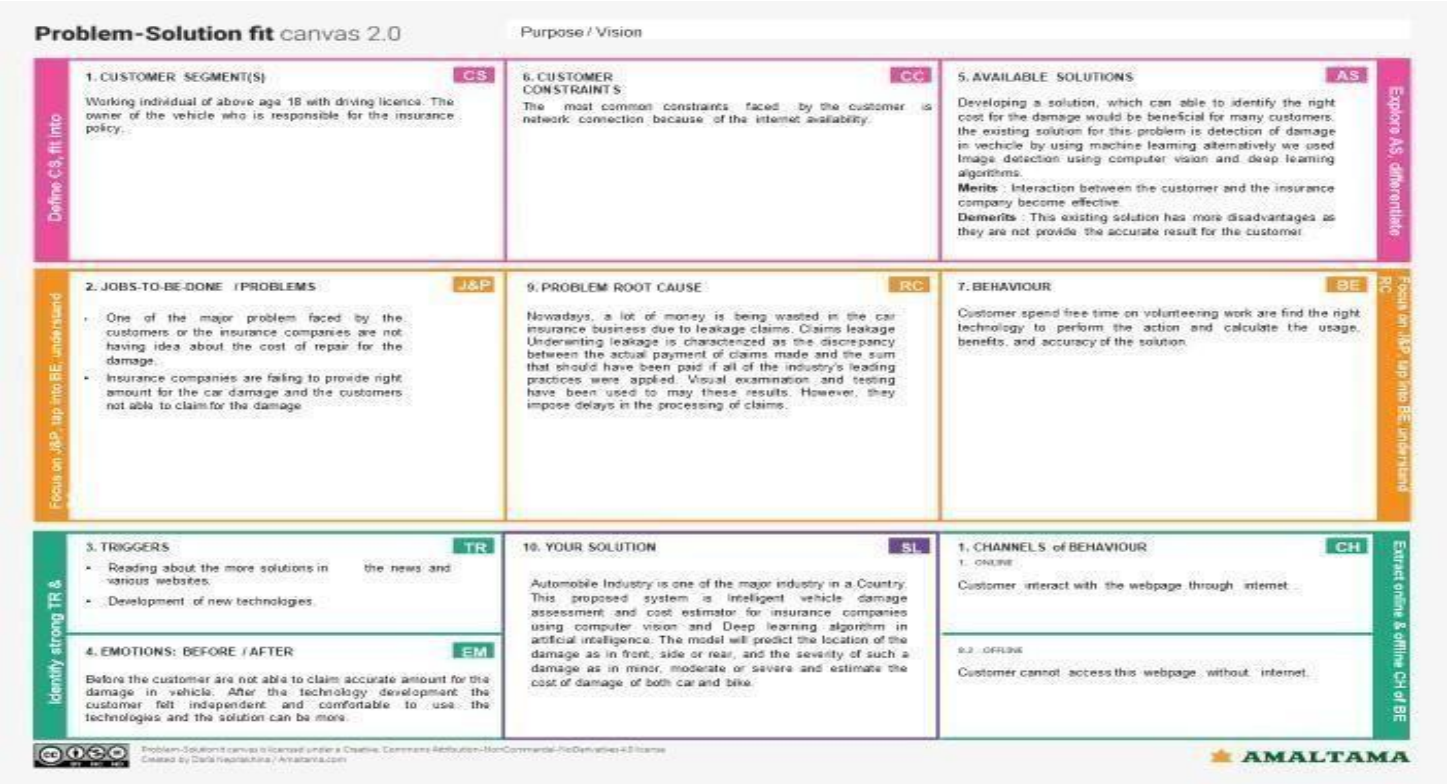


3.3 Idea Prioritization

It deals with the prioritizing of the big ideas in order of highest to lowest likes.



3.4 Problem Solution Fit



3.5 Proposed Solution

S.No.	Parameter	Description
1.	**Problem Statement (Problem to be solved)**	Intelligent vehicle damage assessment and cost estimator for insurance companies.
2.	**Idea / Solution description**	Automobile Industry is one of the major industry in a Country. This proposed system is Intelligent vehicle damage assessment and cost estimator for insurance companies using computer vision in artificial intelligence. The model will predict the location of the damage as in front, side or rear, and the severity of such a damage as in minor, moderate or severe and estimate the cost of damage of car or bike.
3.	**Novelty / Uniqueness**	Image analysis and damage detection using Artificial intelligence**.**

4.	**Social Impact / Customer Satisfaction**	The development of artificial intelligence continue to explore the innovation of insurance technology of 'AI + Vehicle Insurance'. 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.
5.	**Business Model (Revenue Model)**	• Can collaborate with insurance companies. • Can collaborate with car companies.
6.	**Scalability of the Solution**	**Computer Vision, Image detection land cost estimation of vechicle** This combines the rapid compensation of accident vehicles to relieve traffic pressure, to avoid more serious personal property losses caused by secondary accidents and estimate the cost accurately.

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
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 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 .

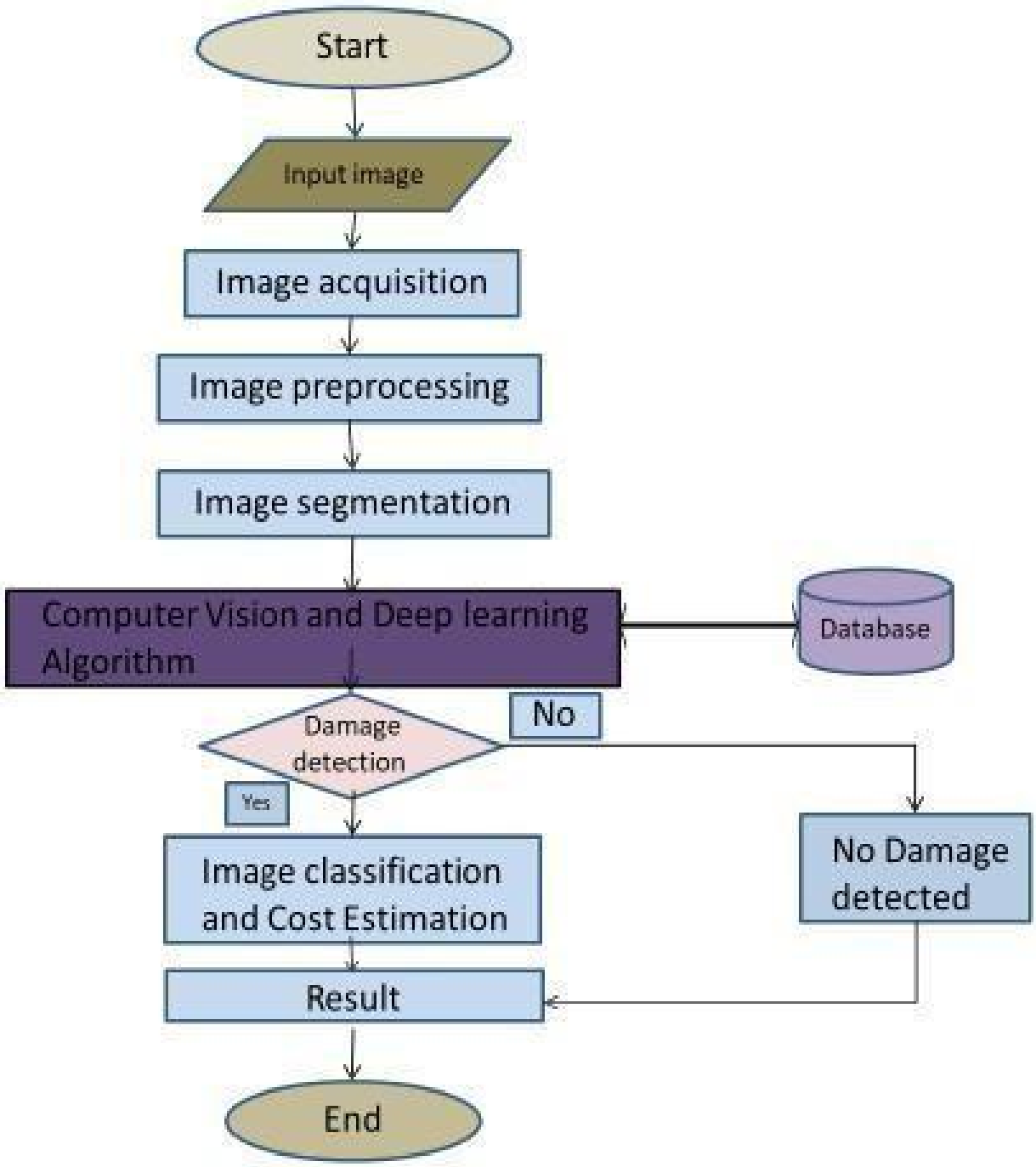
4.2 Non-Functional Requirements

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**	•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 any time and any where.
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.

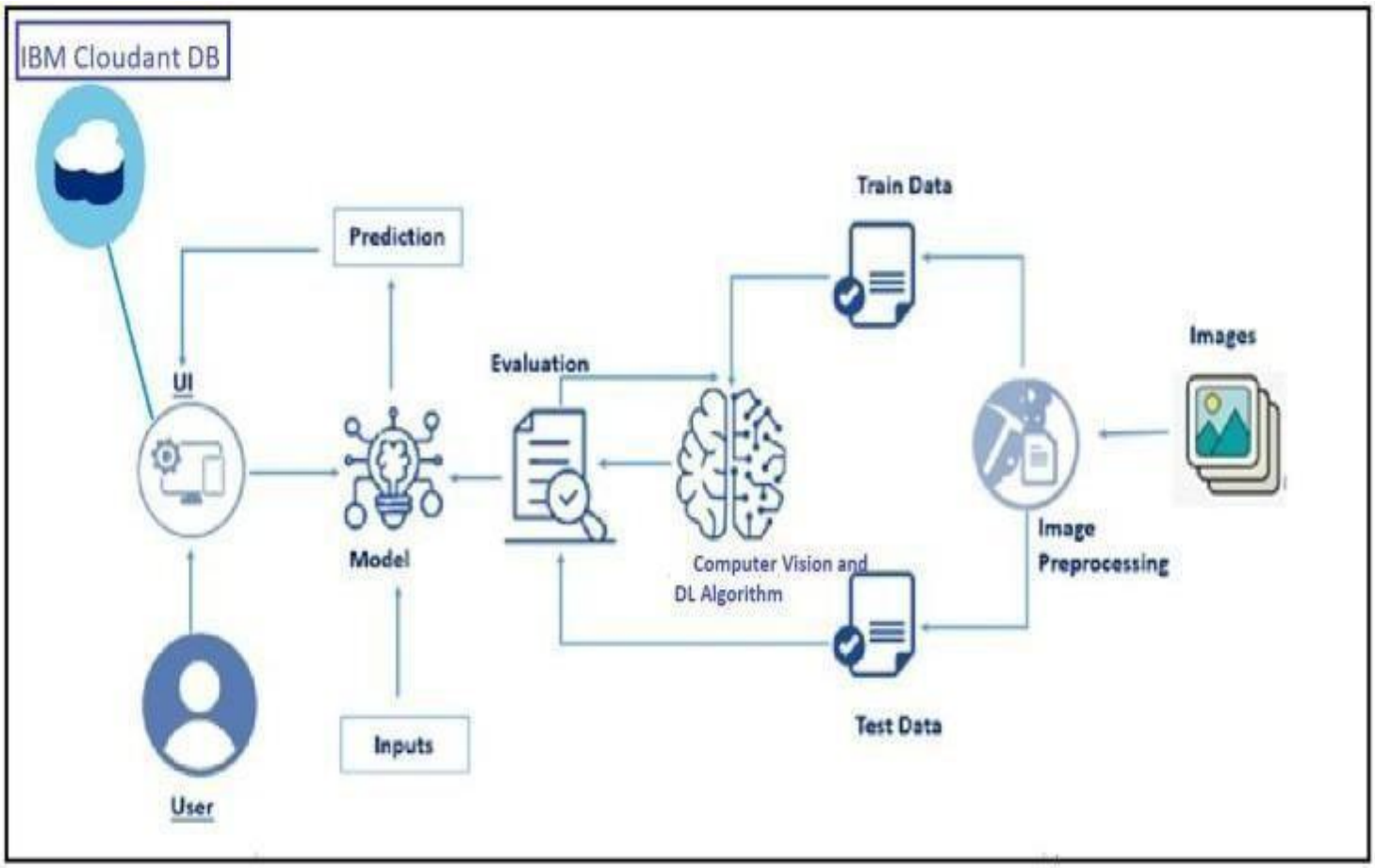
CHAPTER-5

PROJECT DESIGN

5.1 Data Flow Diagram



5.2 Solution Architecture



5.3 Customer Journey Map

User journey

by the Design Team of Accenture Interactive NL

People 2-9 | Time 30 min | Difficulty Beginner

Creating a user journey is a quick way to help you and your team gain a deeper understanding of who you're designing for, aka the stakeholder in your project. The information you add here should be representative of the observations and research you've done about your users.

1 Phases High-level steps your user needs to accomplish from start to finish	Requirements needs	Image collection	Image preprocessing and segmentation	Cost Estimation
2 Steps Detailed actions your user has to perform	Selection of Parameter; Selection of methods to predict; Estimation and Accuracy	Capture the image of the damage vehicle and check the damage is visible in image .Upload the image through the internet. Select the method of damage prediction and estimation of cost.	Measurement of damage level in vehicle by using image detection algorithms . The unnecessary images will be rejected. This image is processed analysed the information and interpret result	Finally the damage is predicted and the cost is estimated of damage vehicle. It will estimate by using the advanced artificial .intelligence algorithms.
3 Feelings What your user might be thinking and feeling at the moment				
	Less unused features; Less development rework; Some defects may occur	High specificity for target data. Detection limits below regulatory trigger criteria. The reasonable throughput for image collection is more quantity is difficult.	Difficult to manage over time and with large data set. Require operation to submit data, sometimes its configuration is required.	Usually feasible under exchange grants to a final estimated cost. but it is challenging to accomplish the specific result to produce.
4 Pain points Problems your user runs into	Undocumented process; Conflict Requirement; Need of new technologies	Lack of technology and human resources occur sometimes. Technical hurdles is one of the pain point. Sometimes it lead to denial of services	Collecting of dataset can be expensive. The large dataset can least to more time to obtain the result. Sometime incorrect may be an problem.	It still has a high require data. Good quality needed for all. To estimate the cost of vehicle is not a easy process.
5 Opportunities Potential improvements or enhancements to the experience	Lower cost of development; Higher level of requirements; More beneficial Measures.	Image detection increase the efficiency. It provides much quicker and accurate result.	Appropriate image detection gives an excellent output. Then it is easy to verify the parameters and can estimates the cost of damage vehicle.	The utilization of data in decision making allows us to make decisions based on evidence, and a!..o speed up the things by making it easier to share the prediction .It also has the advantage of making it easier to verify the result in future.

Share your feedback

CHAPTER-6

PROJECT PLANNING PHASE

6.1 Sprint Planning, Schedule & Estimation

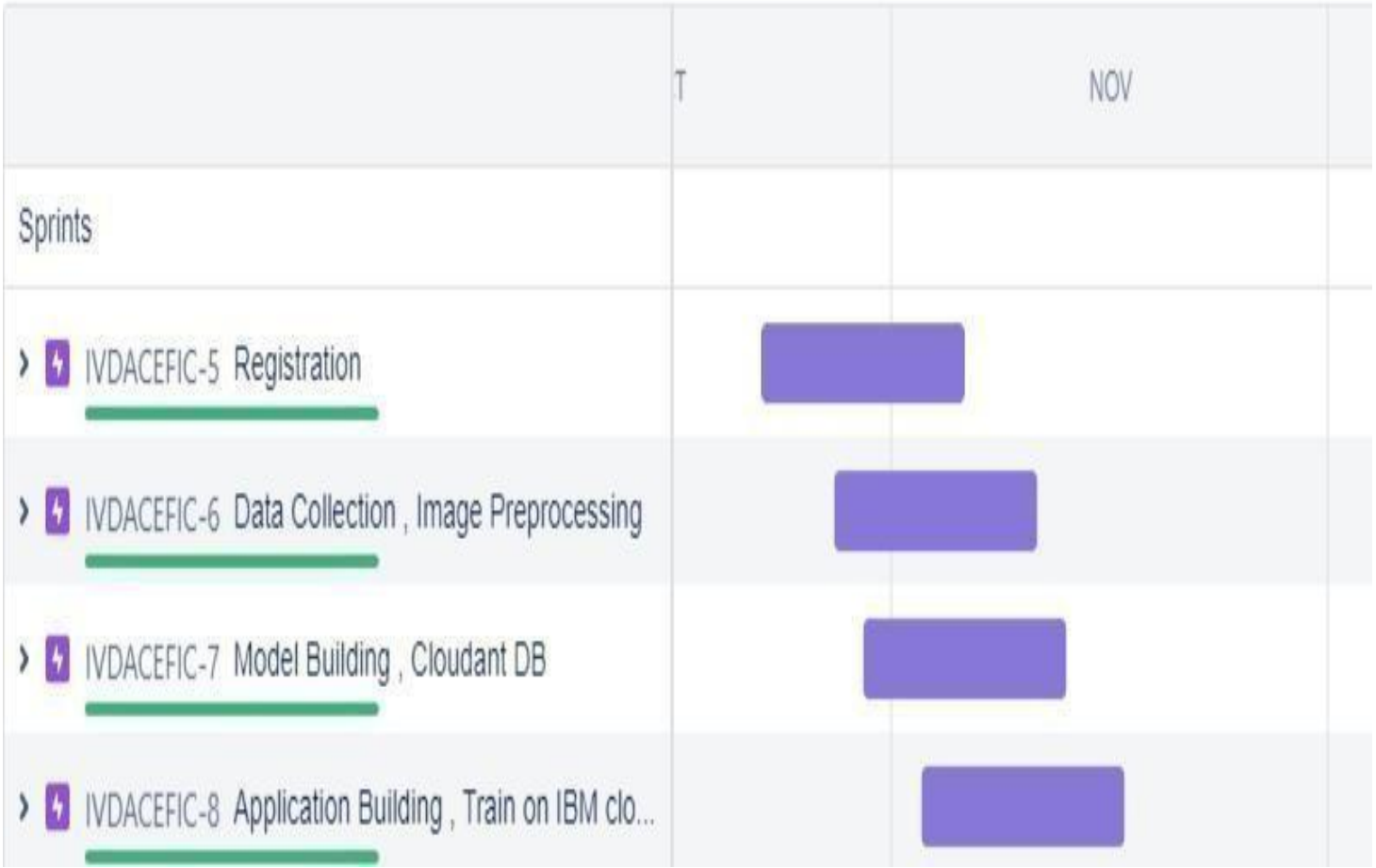
Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Members
Sprint-1	Registration	USN-1	As an owner of the particular vehicle , I can log into the application by entering email & password.	2	High	Swetha B Swetha M Divyasri P Akila K
Sprint-1	User Confirmation	USN-2	As an owner of a particular vehicle, I will receive confirmation email once I have registered for the application.	1	Medium	Swetha B Swetha M Divyasri P Akila K
Sprint-1	Login	USN-3	As an owner of a particular vehicle, I can log into the application by entering email & password.	2	High	Swetha B Swetha M Divyasri P Akila K
Sprint-2	Data Collection	USN-1	Download the dataset used in intelligent vehicle damage assessment & cost estimator for insurance companies.	2	High	Swetha B Swetha M Divyasri P Akila K
Sprint-2	Image Pre Processing	USN-1	Improve the image data that supress unwilling distortions or enhances some image features important for further processing, although performing some geometric transformations of images like rotation, scaling, etc.	2	High	Swetha B Swetha M Divyasri P Akila K
Sprint-3	Model Building	USN-1	Define the model architecture and adding CNN layer and testing ,saving the model.	2	High	Swetha B Swetha M Divyasri P Akila K
Sprint-3	Cloud DB	USN-1	Below are steps that need to follow for creating and using cloudant service. • Register & login to IBM cloud • Create service instance • Creating service credentials • Launch cloudant DB • Create database	2	High	Swetha B Swetha M Divyasri P Akila K

Sprint-4	Application Building	USN-1	Building a web application that is integrated into the model we built A UI is provided to the user where he has uploaded the image. Based on the saved model, the uploaded age will be analyzed and prediction is showcased on the UI.	2	High	Swetha B Swetha M Divyasri P Akila K
Sprint-4	Train The Model On IBM	USN-1	Build Deep learning model and computer vision Using the IBM cloud.	2	High	Swetha B Swetha M Divyasri P Akila K

6.2 Sprint Delivery Schedule

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	4 Days	24 Oct 2022	27 Oct 2022	20	29 Oct 2022
Sprint-2	20	5 Days	28 Oct 2022	01 Nov 2022	20	04 Nov 2022
Sprint-3	20	8 Days	02 Nov 2022	09 Nov 2022	20	11 Nov 2022
Sprint-4	20	9 Days	10 Nov 2022	18 Nov 2022	20	19 Nov 2022

6.3 Reports From JIRA



CHAPTER-7

CODING AND SOLUTION

7.1 Feature

Main.py

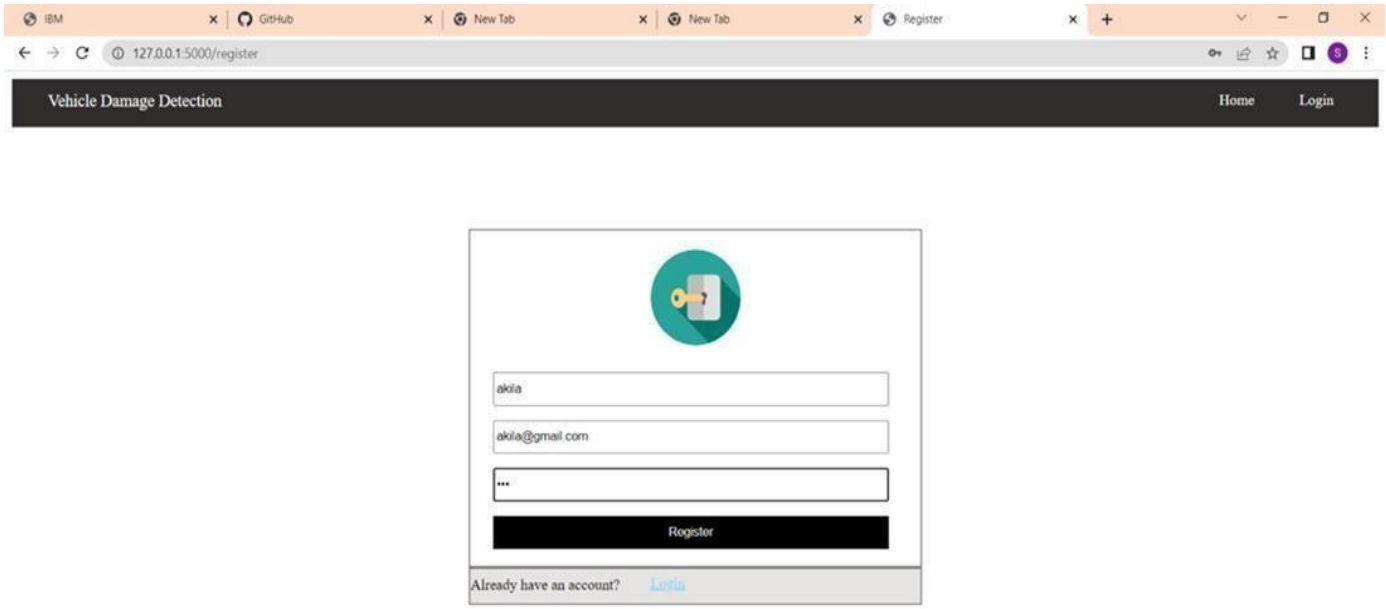
```
from flask import Flask, app, request, render_template import os
import flask
import flask_login import base64
from PIL import Image
from io import BytesIO import datetime
import cv2
import numpy as np
from tensorflow.keras.models import load_model from cloudant.client import Cloudant
from cloudant.error import CloudantException
from cloudant.result import Result, ResultByKey
#os.chdir('Project Development Phase\Sprint-4')
model1 = load_model('Model/level.h5')
model2 = load_model('Model/body.h5')
def detect(frame,model1,f):
    img = cv2.resize(frame,(244,244))
    img = cv2.cvtColor(img,cv2.COLOR_BGR2RGB) if(np.max(img)>1):
        img=img/255.0  img = np.array([img])
prediction = model1.predict(img) if(f):
    label= ['front','rear','side'] else:
    label    =['minor','moderate','severe']
    preds = label[np.argmax(prediction)]
return preds
client = Cloudant.iam( '74ea3e6f-04fa-4f93-977a-f90736ba19f7-
bluemix','eI8X3foZgdln6z0hsZEfVOtCAG8bKC39O8nbBWUQnnGx',connect=True) name = 'name'
email = 'a@b.c'  password='123'
user_database = client.create_database('user_database')
user_image_database = client.create_database('user_image_database')
#upload the database to divyasri account
def image_database_updation(name,email,imagestr): global user_image_database
    now = datetime.datetime.now()
    json_image_document={ 'name':name,
        'email':email,'image':imagestr,
        'datetime':now.strftime("%m/%d/%Y, %H:%M:%S")
    }
```

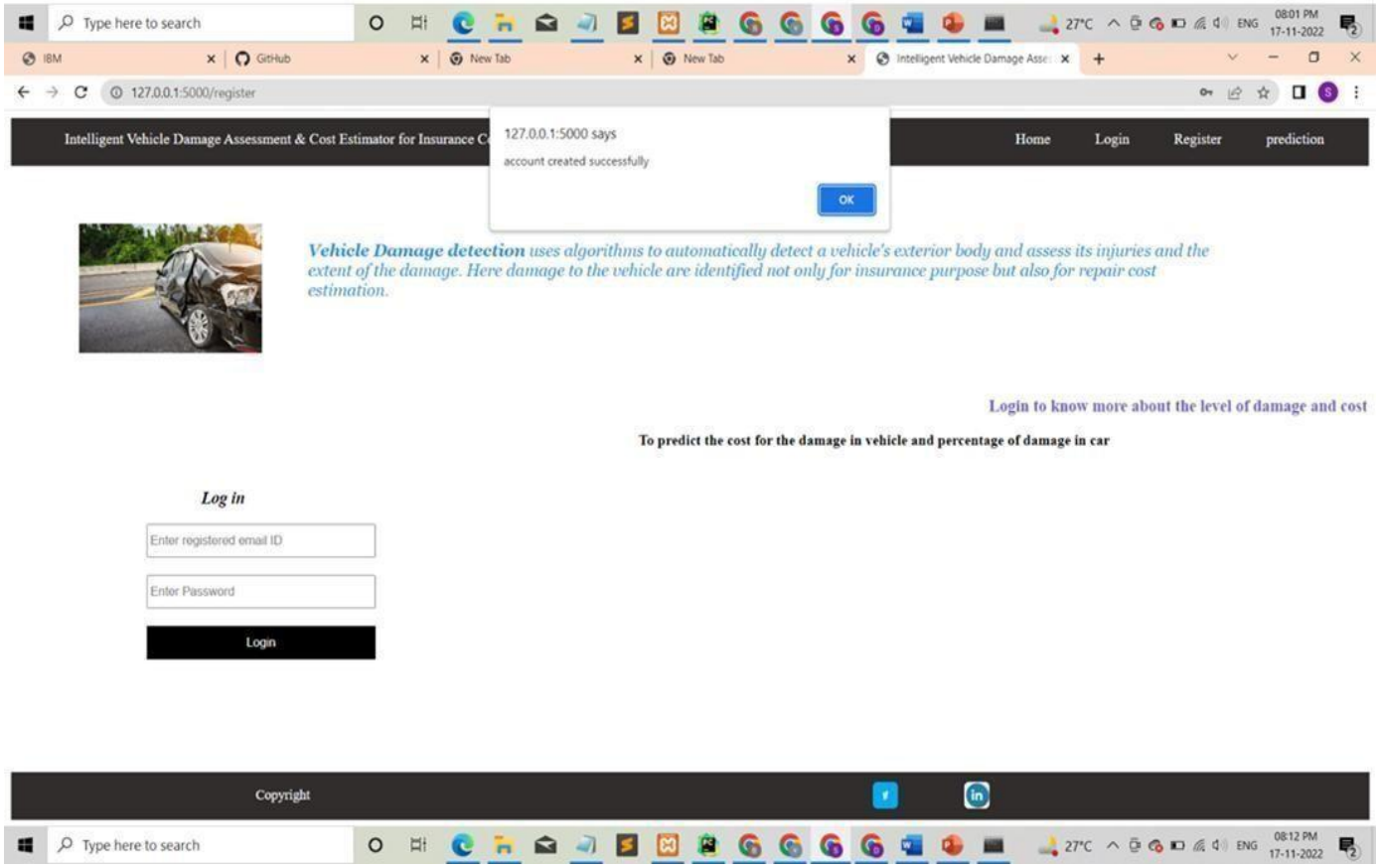
```
new_image_document = user_image_database.create_document(json_image_document)
if(new_image_document.exists()):
print('database updated')
else: print('database couldn\'t be edited')  return
def image_database_retrieval():  global user_image_database
image_result_retrieved = Result(user_image_database.all_docs,include_docs=True) image_result ={}
for i in image_result_retrieved:
if(i['doc']['email'] in image_result.keys()):# like current date> rx date('str')
n = datetime.datetime.strptime(i['doc']['datetime'],'%m/%d/%Y, %H:%M:%S')
o = datetime.datetime.strptime(image_result[i['doc']['email']]['date'],'%m/%d/%Y, %H:%M:%S') if(n>o):
image_result[i['doc']['email']] = {'name':i['doc']['name'],'image':i['doc']['image'],'date':i['doc']['datetime']}
else:
image_result[i['doc']['email']] = {'name':i['doc']['name'],'image':i['doc']['image'],'date':i['doc']['datetime']}
return(image_result)
def database_updation(name,email,password):
global user_database
jsonDocument = { 'name':name, 'email':email, 'password':password}
newDocument = user_database.create_document(jsonDocument)
if(newDocument.exists()): print('database updated')
else:
print('database couldn\'t be edited')
return
 #database_updation(name,email,password)
def database_retrieval(): global user_database
result_retrieved = Result(user_database.all_docs,include_docs=True) #print(list(result_retrieved))
result = {}
for i in list(result_retrieved): result[i['doc']['email']]={'name':i['doc']['name'],'password':i['doc']['password']}
return result #print(database_retrieval())
app = Flask( name ) app.secret_key = 'apple'
login_manager = flask_login.LoginManager()
login_manager.init_app(app)
users = {'a@b.c': {'password': '123'}} class User(flask_login.UserMixin):pass
@login_manager.user_loader def user_loader(email):
data = database_retrieval()
if email not in data:
return
user = User()  user.id = email
user.name = data[email]['name'] return user
@login_manager.request_loader def request_loader(request):
email = request.form.get('email')
data = database_retrieval() if email not in data:
return
user = User() user.id = email
user.name = data[email]['name']
return user @app.route('/')
```

```
def index(): if(flask_login.current_user.is_authenticated):
return render_template('dashboard.html')
else:
return flask.redirect(flask.url_for('login'))

@app.route('/register',methods = ['GET','POST']) def register():
data = database_retrieval() if(flask.request.method == 'GET'):
return render_template('register.html') email = flask.request.form['email']
if(email in data):
return render_template('register.html',flash_message='True') else:
database_updation(flask.request.form['name'],email,flask.request.form['password'])
#users[email]={'password':flask.request.form['password']}
user = User() user.id = email
user.name = flask.request.form['name']
flask_login.login_user(user)
return render_template('dashboard.html',flash_message='True')
@app.route('/login',methods =['GET','POST']) def login():
data = database_retrieval() if(flask.request.method == 'GET'):
return render_template('login.html',flash_message='False') email = flask.request.form['email']
if(email in data and flask.request.form['password']==data[email]['password']): user = User()
user.id = email flask_login.login_user(user)
return render_template('dashboard.html',flash_message='Fal')
#flask.flash('invalid credentials !!!')
return render_template('login.html',flash_message="True") #error = 'inavlid credentials')
@app.route('/dashboard',methods = ['GET','POST']) @flask_login.login_required
def dashboard(): if(flask.request.method == 'GET'):
return render_template('dashboard.html',flash_message='False')
email = flask.request.form['email']
if(email in users and flask.request.form['password']==users[email]['password']): user = User()
user.id = email flask_login.login_user(user)
return render_template('dashboard.html',flash_message="Fal") return
render_template('dashboard.html',flash_message="Fals")
@app.route('/logout') @flask_login.login_required def logout():
flask_login.logout_user()
return render_template('logout.html')
@app.route('/prediction',methods = ['GET','POST']) @flask_login.login_required
def prediction():
if(flask.request.method=='POST'): img = flask.request.files['myFile'] try:
os.remove('static\imagedata\save.png') except:
Pass imgstr = base64.b64encode(img.read()).decode('utf-8')
image_database_updation(flask_login.current_user.name,flask_login.current_user.id,imgstr)
data = image_database_retrieval()
print(flask_login.current_user.id)
#print(len(base64.b64decode(data[flask_login.current_user.id]['image'].strip())))
image = Image.open(BytesIO(base64.b64decode(data[flask_login.current_user.id]['image'])))
img_retrived = np.array(image)
```

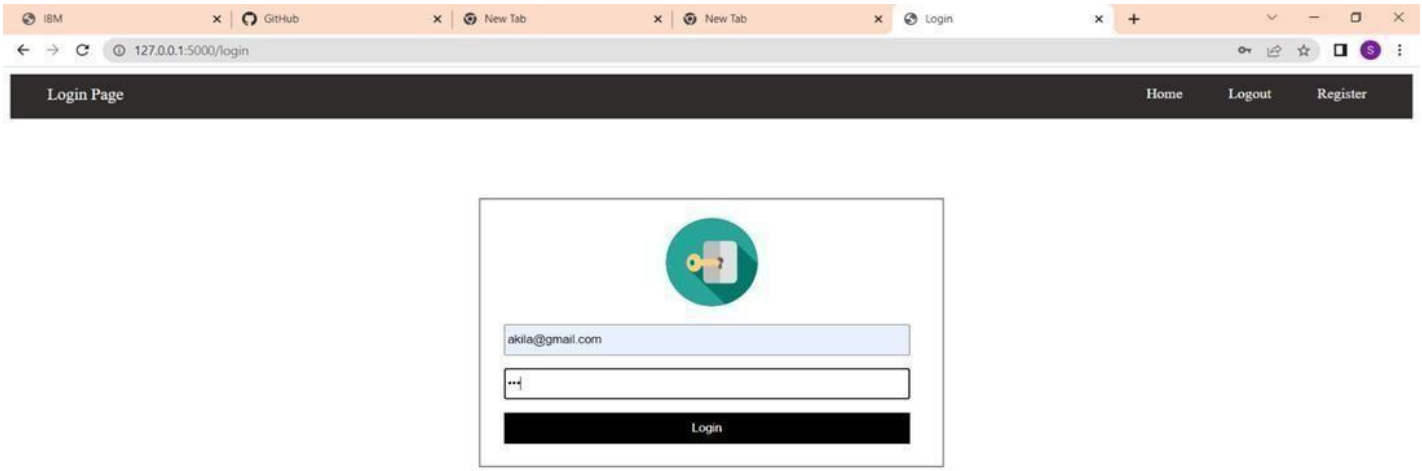
```
'''img_retrived = np.asarray(base64.b64decode(data[flask_login.current_user.id]['image']))
print(data[flask_login.current_user.id]['image'])
print(img_retrived.shape)'''
#img_retrived = np.resize(img_retrived,(244,244)) img_retrive = Image.fromarray(img_retrived)
img_retrive.save('static/image data/sae.png') '''img_retrived = np.frombuffer(
BytesIO(  base64.b64decode(data[flask_login.current_user.id]['image'])
))'''  print('###################################')
result1=detect(img_retrived,model1=model2,f=True) result2 = detect(img_retrived,model1=model1,f=False)
value=''
if(result1 == 'front' and result2 == 'minor'):
value = 'Cost=3000 - 5000 INR , Percentage=10% , Type=car front minor damage’
elif(result1 == 'front' and result2 == 'moderate'):
value = 'Cost=6000 - 8000 INR , Percentage=20% , Type=car front moderate damage'
elif(result1 == 'front' and result2 == 'severe'):
value = 'Cost=9000 - 11000 INR , Percentage=30% , Type=car front severe damage’
elif(result1 == 'rear' and result2 == 'minor'):
value = 'Cost=4000 - 6000 INR , Percentage=40% , Type=car rear minor damage’
elif(result1 == 'rear' and result2 == 'moderate'):
value = 'Cost=7000 - 9000 INR , Percentage=50% , Type=car rear moderate damage’
elif(result1 == 'rear' and result2 == 'severe'):
value = 'Cost=11000 - 13000 INR , Percentage=60% , Type=car rear severe damage’
elif(result1 == 'side' and result2 == 'minor'):
value = 'Cost=6000 - 8000 INR , Percentage=70% , Type=car side minor damage'
elif(result1 == 'side' and result2 == 'moderate'):
value = 'Cost=9000- 11000 INR , Percentage=80% , Type=car side moderate damage’
elif(result1 == 'side' and result2 == 'severe'):
value = 'Cost=12000 - 15000 INR , Percentage=90% , Type=car side severe damage’
else:
value = '16000 - 50000 INR , Percentage=100% ,Type=heavy damage' print(result1,result2,value)
print('###################################')
      img_retrived = Image.fromarray(img_retrived)
      img_retrived.save('static/image data/save.png’)
      print('image uploaded and retrieved')
      return render_template('prediction.html',flash_message='True',value = result1+' '+result2+' '+value)
      #,imag=img_retrived)
      return render_template('prediction.html',flash_message='Flase’)

      if__name== ' main ':
   app.run(debug=True)
```

Register.html

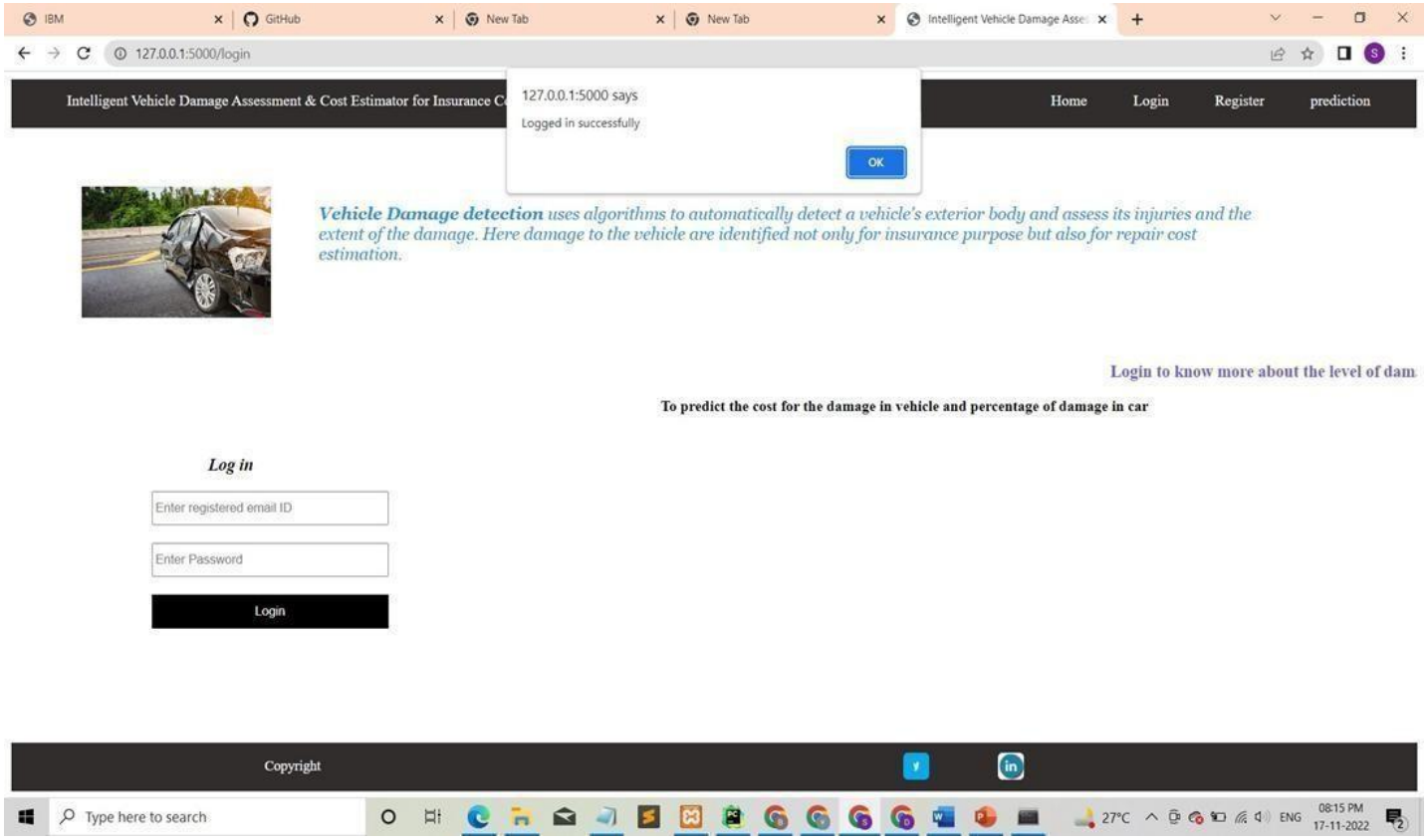




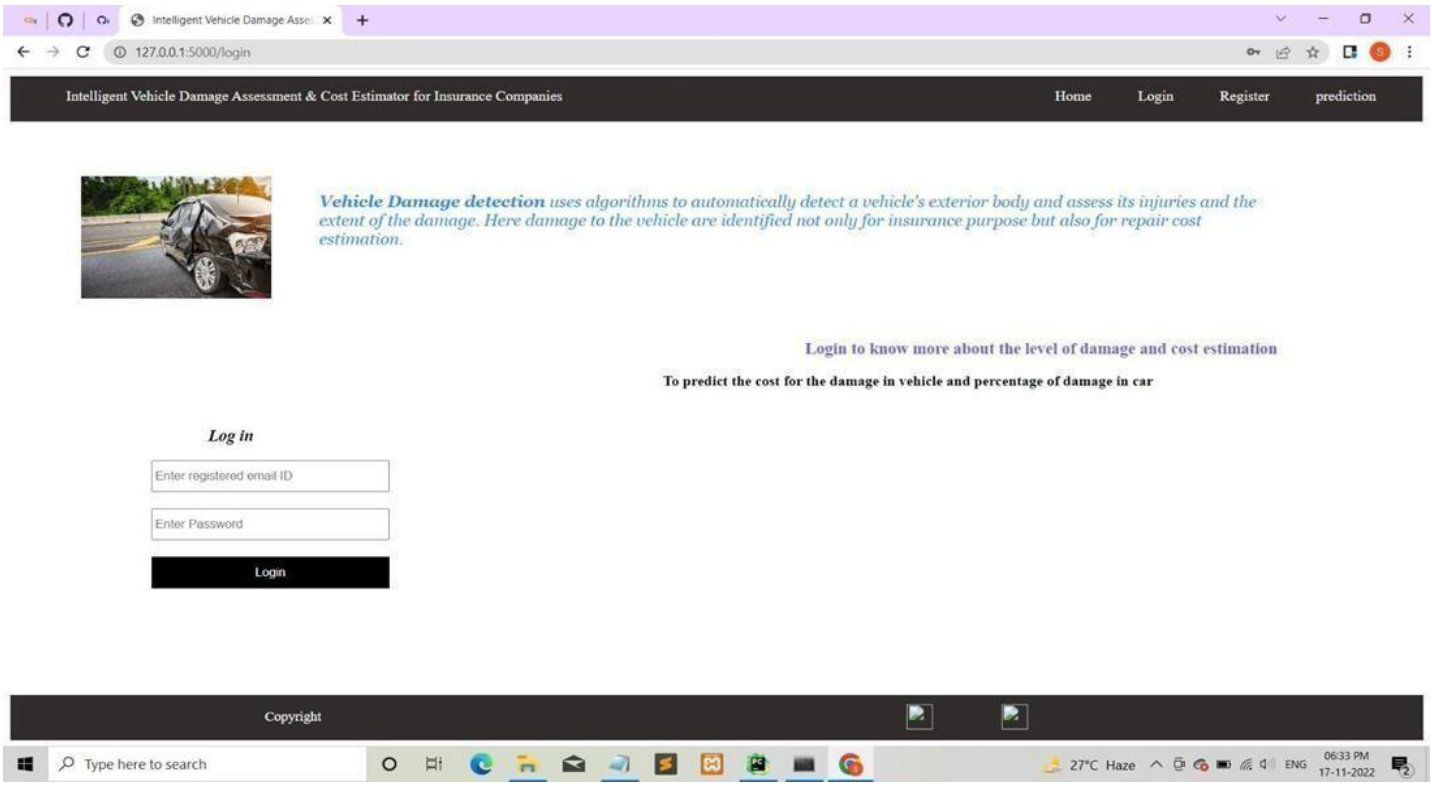
Login.html


IBM
GitHub
New Tab
New Tab
Login
127.0.0.1:5000/login
Login Page
Home
Logout
Register
akila@gmail.com
Login

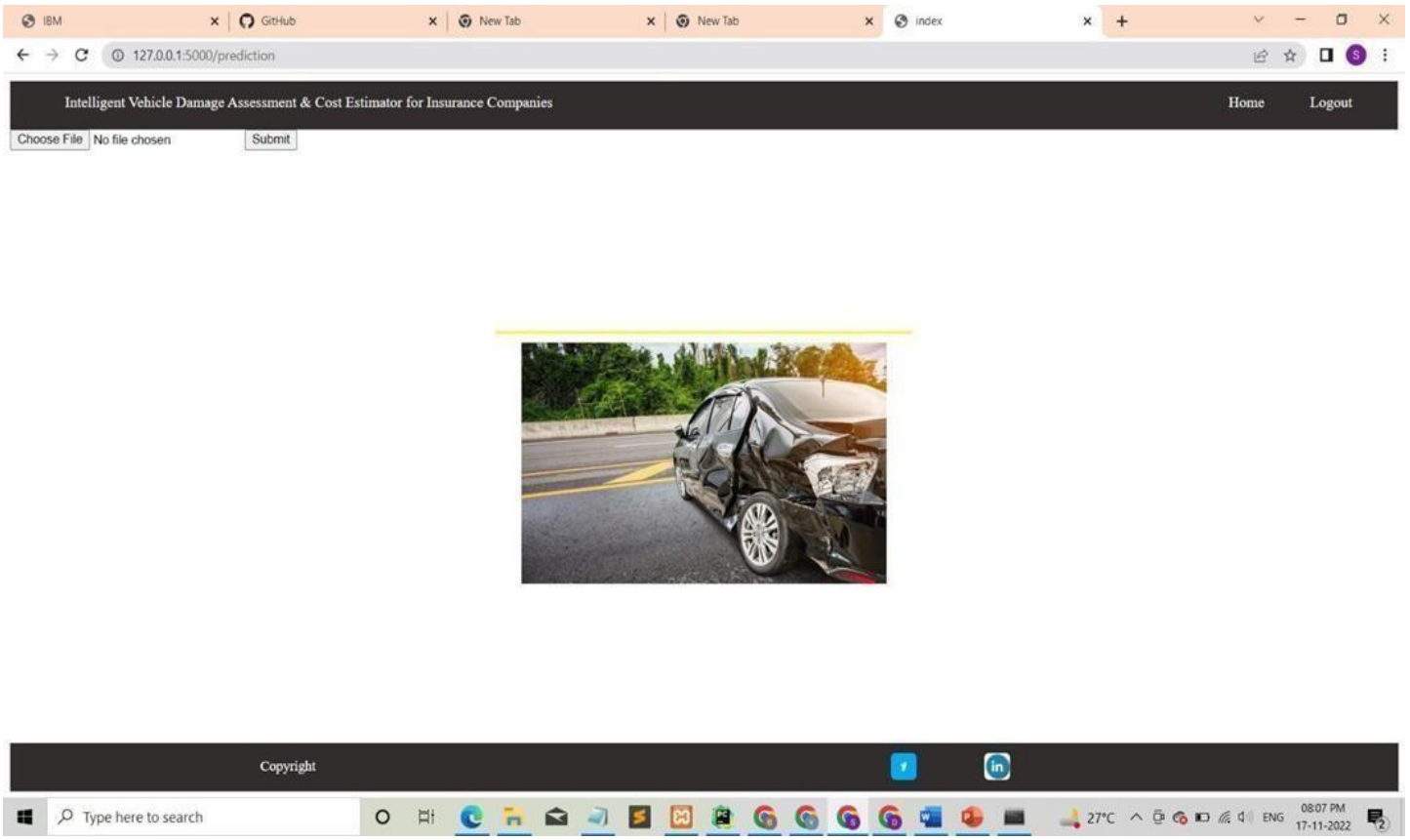

Type here to search
27°C
ENG
08:00 PM
17-11-2022

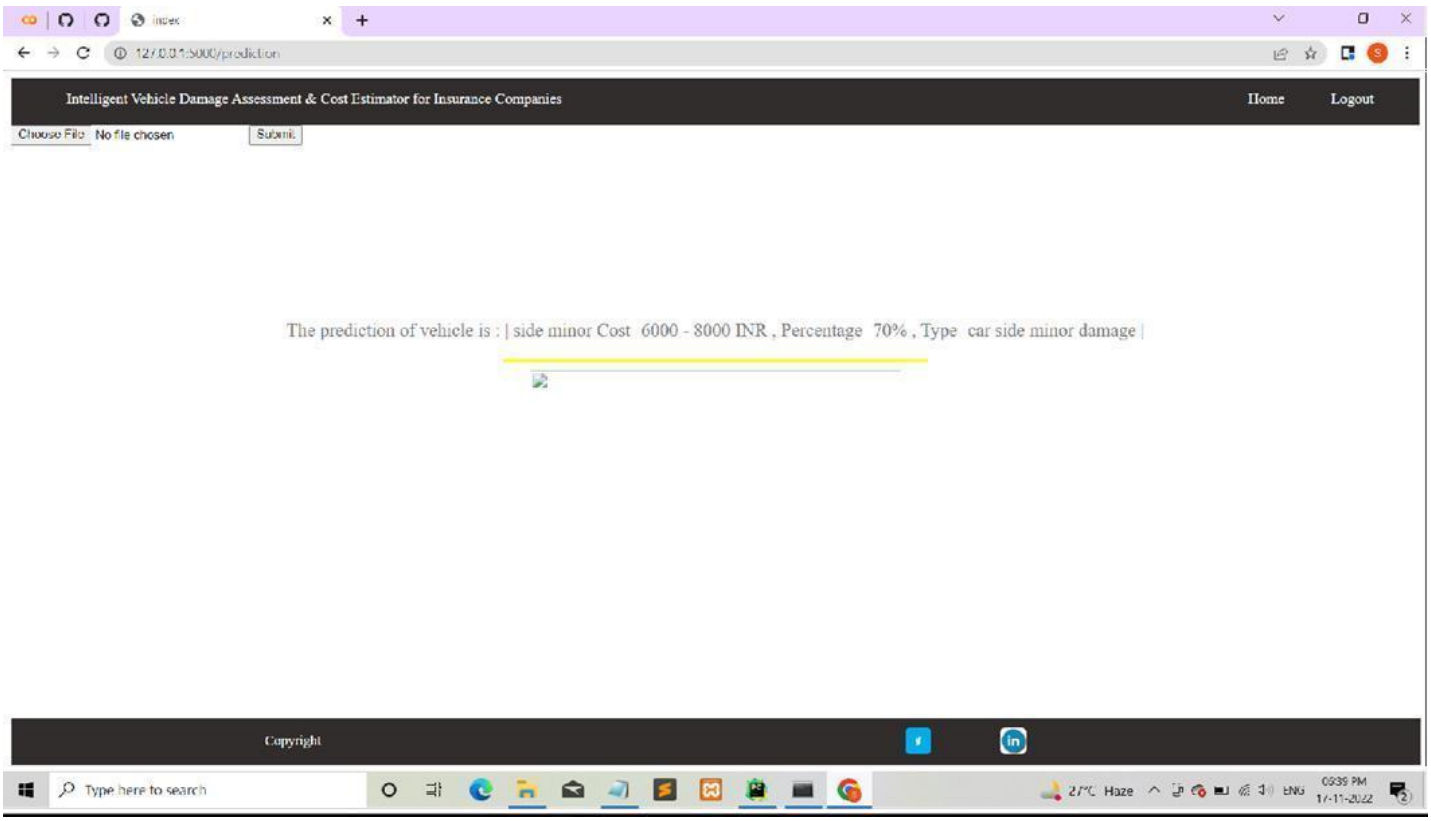

IBM
GitHub
New Tab
New Tab
Intelligent Vehicle Damage Asse
127.0.0.1:5000/login
Intelligent Vehicle Damage Assessment & Cost Estimator for Insurance C
Home
Login
Register
prediction
127.0.0.1:5000 says
Logged in successfully
OK
Vehicle Damage detection uses algorithms to automatically detect a vehicle's exterior body and assess its injuries and the extent of the damage. Here damage to the vehicle are identified not only for insurance purpose but also for repair cost estimation.
Login to know more about the level of dam
To predict the cost for the damage in vehicle and percentage of damage in car
Log in
Enter registered email ID
Enter Password
Login
Copyright
Type here to search
27°C
ENG
08:15 PM
17-11-2022

Dashboard.html

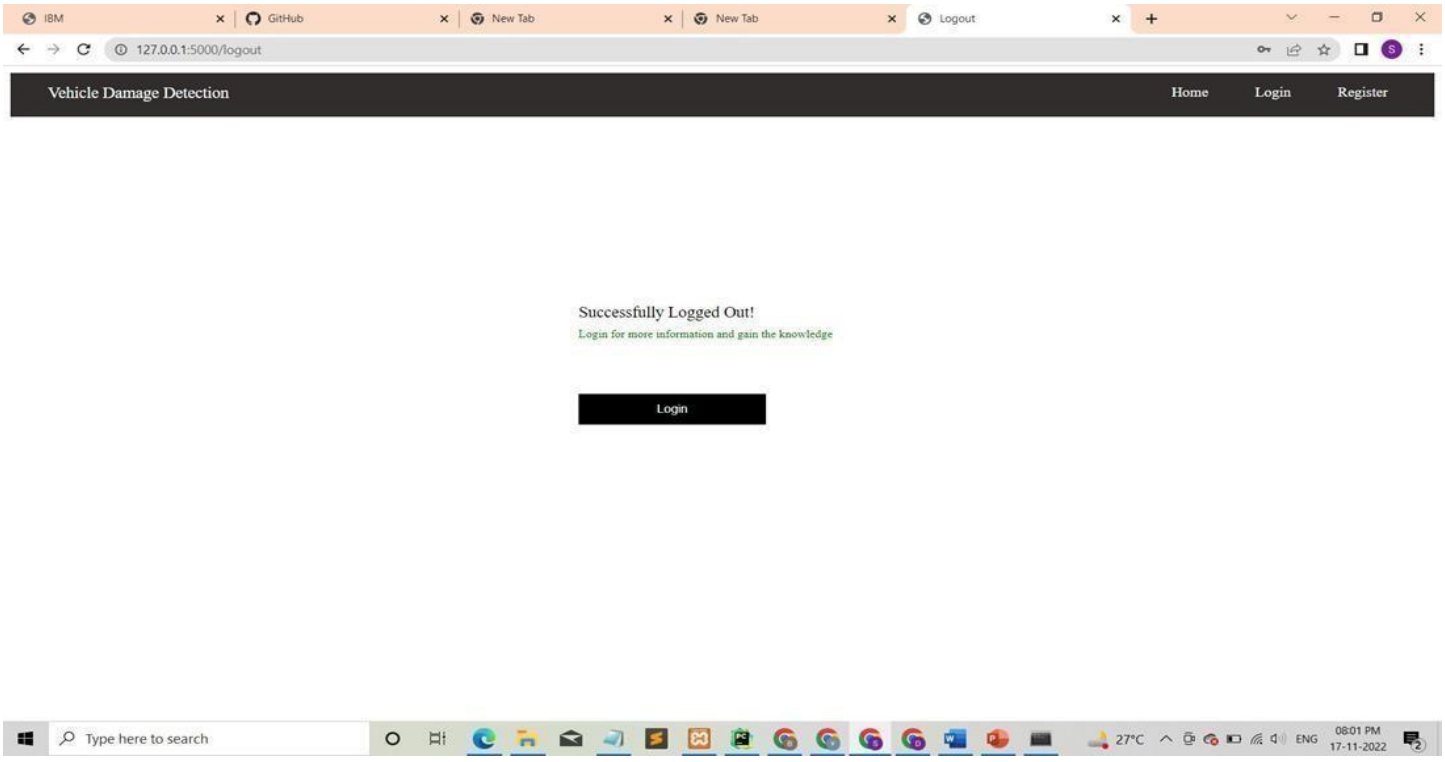


Prediction.html

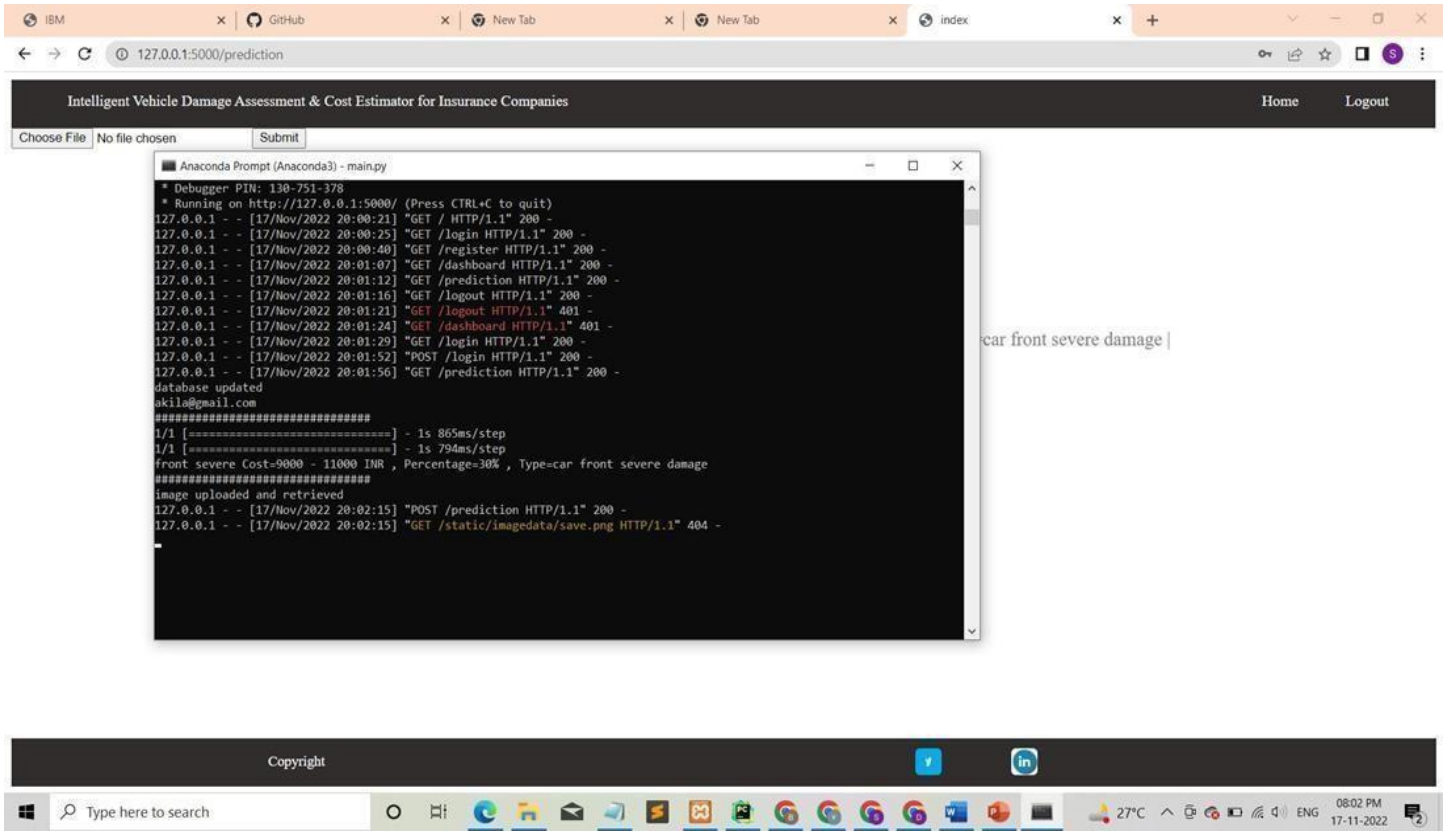



Intelligent Vehicle Damage Assessment & Cost Estimator for Insurance Companies
Home
Logout
Choose File
No file chosen
Submit
The prediction of vehicle is : | side minor Cost 6000 - 8000 INR , Percentage 70% , Type car side minor damage |
Copyright
Type here to search

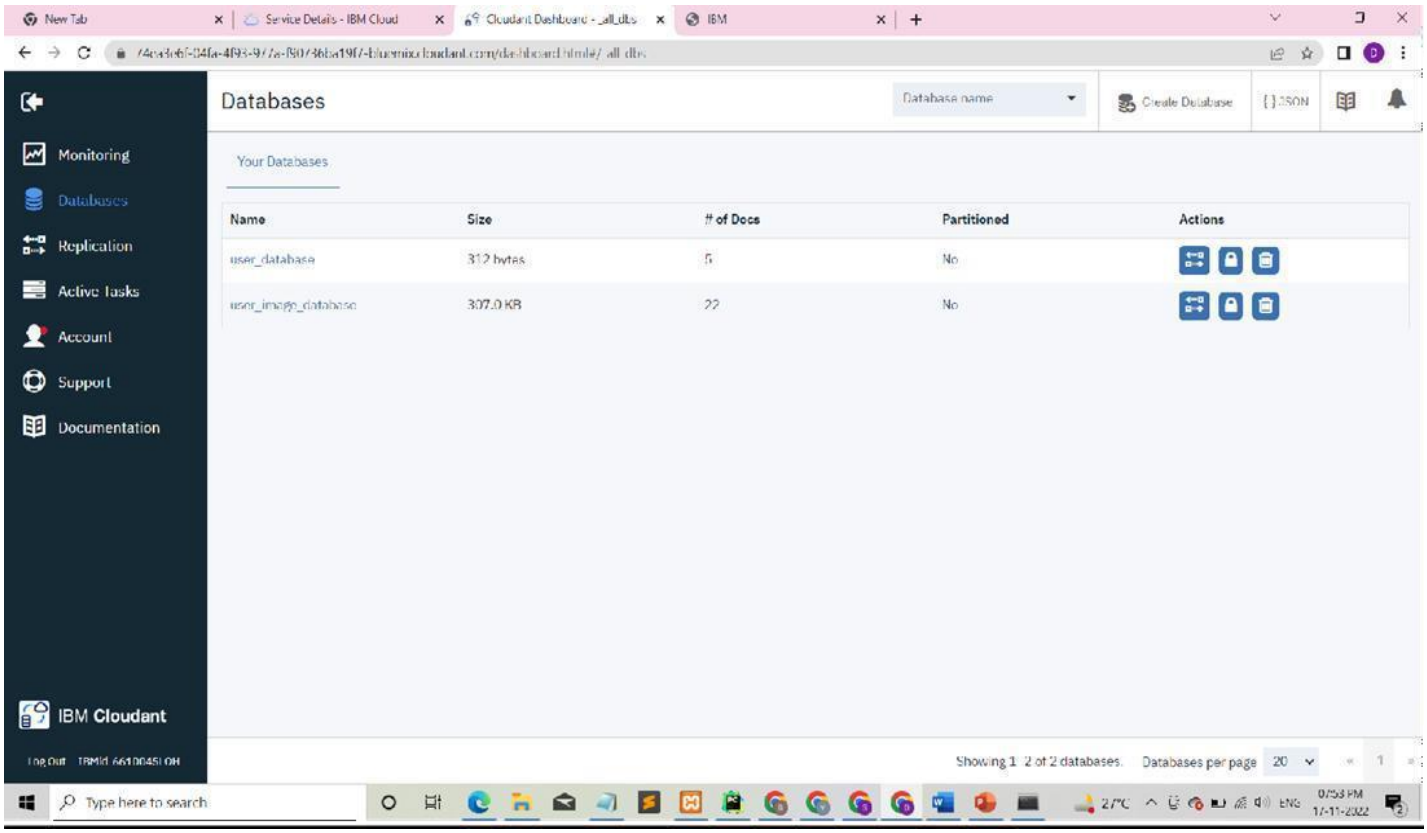
Logout.html


IBM
GitHub
New Tab
New Tab
Logout
127.0.0.1:5000/logout
Vehicle Damage Detection
Home
Login
Register
Successfully Logged Out!
Login for more information and gain the knowledge
Login
Type here to search

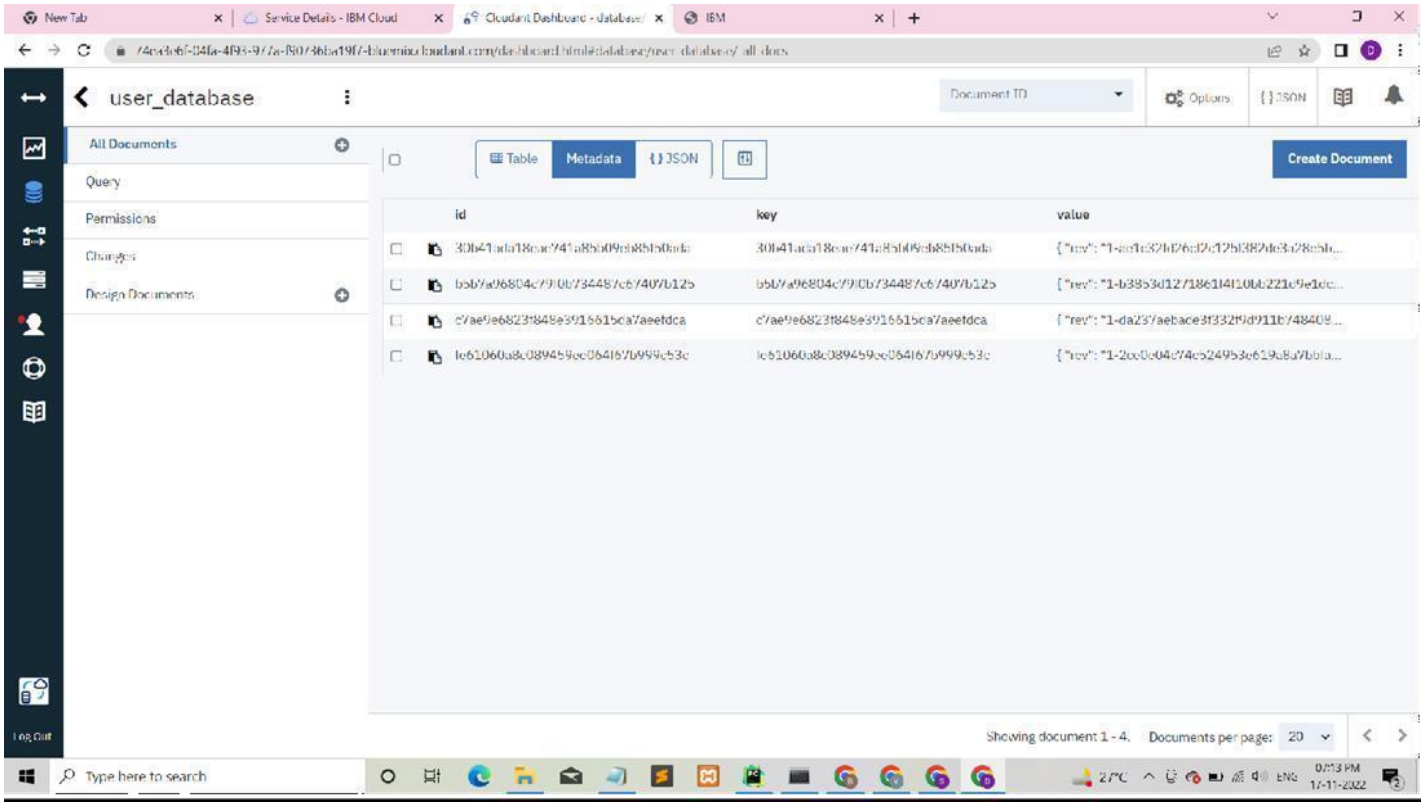
Main.py



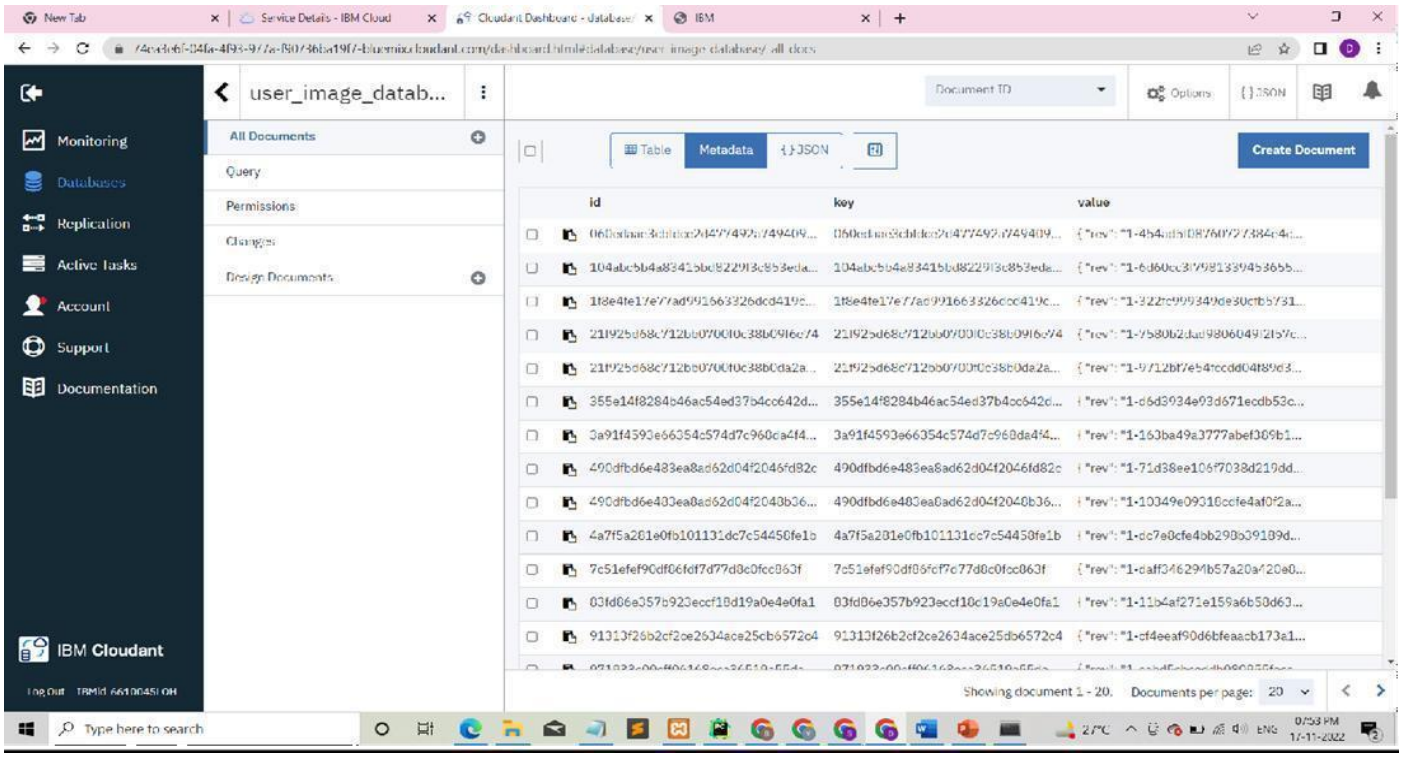
Cloud database



User database:



User image database:



CHAPTER-8

TESTING

8.1. Test Case

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
HomePage_TC_OO1	Functional	Home Page	Verify user is able to see the home page or not.		1.Enter URL andclick go 2. verifywhether the user is able to see the home page.	Enter URL andclick go	User able to see the home page	Working as expected	Pass	Nil	N	_	Swetha B
HomePage_TC_OO2	UI	Home Page	Verify the UI elementsin Home Page		1.Enter URL andclick go 2. Verify the UI elements in Home Page.	Enter URL andclick go	Application should show below UI elements:	Working as expected	pass	Nil	N	_	Swetha M
RegisterPage_TC_OO3	Functional	RegisterPage	A Register page is able to will Inputthe user data.		1.Enter URL andclick go 2.Verify the UI elements in Home Page 3.Click the Register button	Click in register page	Applicationshould show 'Incorrect email or password ' validation message.	Working as expected	pass	Nil	N		Divyasri P
LoginpageTC_OO4	Functional	login page	Verify user is able to redirect to predictpage or not.		1. Enter URL and click go 2.Verify the UI elements in Home Page 3.Click the Loginbutton 4.Click on Predict button 5.Verifywhether the user to redirectto predict page or not.	Click in login home page	Application should show 'Incorrect email or password ' validation message.	Working as expected	pass	Nil	N	_	Akila K
PredictPage_TC_OO5	UI	Predict page	Verify the UI elements in Predict Page		1.Enter URL andclick go 2. Verify the UI elementsin PredictPage.	Click the predict button and redirectto predict page	Application should show below UI elements: Upload file Button, Predictbutton.	Working as expected	pass	Nil	N	_	Swetha B, Swetha M
PredictPage_TC_OO6	Functional	Predict page	Verify user is able to select the predict		1.Enter URL and click go 2.Click on Predictbutton 3.Verifywhether the user to redirectto predict page or not.	vehicle damage images	Application should shows user to choose predict option	Working as expected	pass	Nil	N	_	Swetha B, Divyasri P
PredictPage_TC_OO7	Functional	Predict page	Verify user is able to upload the image or not.		1.Enter URL and click go 2.Click on Predictbutton 3.Verify whether the user to redirect to predict page or not. 4.Verify user is able to select the dropdown value or not. 5.Verify user is able to uploadthe imagesor not	Imagesto be Uploaded	Application should shows the uploaded image.	Working as expected	pass	Nil	N	_	Swetha B, Akila K
PredictPage_TC_OO8	Functional	Predict page	Verify whether the image is predicted correctly or not		1.Enter URL and click go 2.Click on Predictbutton 3.Verify whether the user to redirect to predict page or not. 4.Verify user is able to select the dropdown value or not. 5.Verify user is able to uploadthe images or not 6. Verify whether the image is predictedcorrectly or not	Clickthe Predict Button	Applicationshowsthe predicted output	Working as expected	pass	Nil	N	_	Swetha B

8.2 User Acceptance Testing

1. Purpose of Document

The purpose of this document is to briefly explain the test coverage and open issues of the [Intelligent vehicle damage assessment & cost estimator for insurance companies] project at the time of the release to User Acceptance Testing (UAT).

2. 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	7	7	9	7	30
Duplicate	4	0	2	0	6
External	1	2	0	2	5
Fixed	14	1	6	8	29
Not Reproduced	0	0	1	0	1
Skipped	0	0	2	1	3
Won't Fix	0	4	1	0	5
Totals	26	14	21	19	80

3. Test Case Analysis

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

Section	Total Cases	Not Tested	Fail	Pass
Print Engine	5	0	0	5
Client Application	28	0	0	28
Security	2	0	0	2
Outsource Shipping	1	0	0	1
Exception Reporting	6	0	0	6
Final Report Output	8	0	0	8
Version Control	1	0	0	1

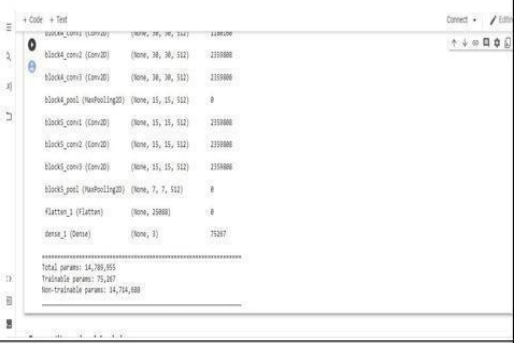
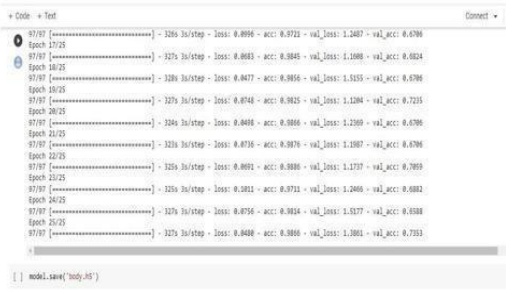
CHAPTER-9

RESULTS

9.1 Performance Metrics

Model Performance Testing:

Project team shall fill the following information in model performance testing template.

S.No.	Parameter	Values	Screenshot
1.	Model Summary	Total params:14,789,955 Trainable params: 75,267 Non-trainable params: 14,714,688	
2.	Accuracy	Training Accuracy – 98.66 Validation Accuracy – 73.53	

Model Building:

```
model=Model(inputs=vgg.input,outputs=prediction)
model.summary()
```

```
Model: "model_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_2 (InputLayer)        [(None, 244, 244, 3)]     0

 block1_conv1 (Conv2D)       (None, 244, 244, 64)      1792

 block1_conv2 (Conv2D)       (None, 244, 244, 64)      36928

 block1_pool (MaxPooling2D)  (None, 122, 122, 64)      0

 block2_conv1 (Conv2D)       (None, 122, 122, 128)     73856

 block2_conv2 (Conv2D)       (None, 122, 122, 128)     147584

 block2_pool (MaxPooling2D)  (None, 61, 61, 128)       0

 block3_conv1 (Conv2D)       (None, 61, 61, 256)       295168

 block3_conv2 (Conv2D)       (None, 61, 61, 256)       590080

 block3_conv3 (Conv2D)       (None, 61, 61, 256)       590080

 block3_pool (MaxPooling2D)  (None, 30, 30, 256)       0
```

```
 block4_conv1 (Conv2D)       (None, 30, 30, 512)       1180160

 block4_conv2 (Conv2D)       (None, 30, 30, 512)       2359808

 block4_conv3 (Conv2D)       (None, 30, 30, 512)       2359808

 block4_pool (MaxPooling2D)  (None, 15, 15, 512)       0

 block5_conv1 (Conv2D)       (None, 15, 15, 512)       2359808

 block5_conv2 (Conv2D)       (None, 15, 15, 512)       2359808

 block5_conv3 (Conv2D)       (None, 15, 15, 512)       2359808

 block5_pool (MaxPooling2D)  (None, 7, 7, 512)         0

 flatten_1 (Flatten)         (None, 25088)             0

 dense_1 (Dense)             (None, 3)                 75267

=================================================================
Total params: 14,789,955
Trainable params: 75,267
Non-trainable params: 14,714,688
_________________________________________________________________
```

Accuracy:

```
r = model.fit_generator(
    training_set,
    validation_data = test_set,
    epochs = 25,
    steps_per_epoch=979//10,
    validation_steps = 171//10
)
```

```
/tmp/wsuser/ipykernel_164/289406290.py:1: UserWarning: `Model.fit_generator` is deprecated and will be removed in a future version. Please use
  r = model.fit_generator(
Epoch 1/25
97/97 [==============================] - 339s 3s/step - loss: 1.1511 - acc: 0.5459 - val_loss: 0.9324 - val_acc: 0.6294
Epoch 2/25
97/97 [==============================] - 328s 3s/step - loss: 0.6237 - acc: 0.7534 - val_loss: 0.7954 - val_acc: 0.6941
Epoch 3/25
97/97 [==============================] - 331s 3s/step - loss: 0.4937 - acc: 0.8070 - val_loss: 1.1732 - val_acc: 0.6176
Epoch 4/25
97/97 [==============================] - 326s 3s/step - loss: 0.4349 - acc: 0.8411 - val_loss: 0.9766 - val_acc: 0.6824
Epoch 5/25
97/97 [==============================] - 326s 3s/step - loss: 0.3661 - acc: 0.8617 - val_loss: 1.1987 - val_acc: 0.6529
Epoch 6/25
97/97 [==============================] - 325s 3s/step - loss: 0.2681 - acc: 0.8875 - val_loss: 0.9087 - val_acc: 0.6941
Epoch 7/25
97/97 [==============================] - 325s 3s/step - loss: 0.2292 - acc: 0.9195 - val_loss: 1.0251 - val_acc: 0.6647
Epoch 8/25
97/97 [==============================] - 326s 3s/step - loss: 0.1248 - acc: 0.9659 - val_loss: 1.0597 - val_acc: 0.6706
Epoch 9/25
97/97 [==============================] - 323s 3s/step - loss: 0.1315 - acc: 0.9639 - val_loss: 1.0529 - val_acc: 0.6647
Epoch 10/25
```

```
97/97 [==============================] - 322s 3s/step - loss: 0.0922 - acc: 0.9752 - val_loss: 0.9898 - val_acc: 0.6588
Epoch 11/25
97/97 [==============================] - 323s 3s/step - loss: 0.0913 - acc: 0.9825 - val_loss: 1.5796 - val_acc: 0.6529
Epoch 12/25
97/97 [==============================] - 322s 3s/step - loss: 0.1447 - acc: 0.9536 - val_loss: 1.1999 - val_acc: 0.6706
Epoch 13/25
97/97 [==============================] - 325s 3s/step - loss: 0.0746 - acc: 0.9763 - val_loss: 1.1819 - val_acc: 0.6647
Epoch 14/25
97/97 [==============================] - 325s 3s/step - loss: 0.1078 - acc: 0.9711 - val_loss: 1.0919 - val_acc: 0.7176
Epoch 15/25
97/97 [==============================] - 327s 3s/step - loss: 0.0659 - acc: 0.9866 - val_loss: 1.0925 - val_acc: 0.6824
Epoch 16/25
97/97 [==============================] - 326s 3s/step - loss: 0.0996 - acc: 0.9721 - val_loss: 1.2487 - val_acc: 0.6706
Epoch 17/25
97/97 [==============================] - 327s 3s/step - loss: 0.0683 - acc: 0.9845 - val_loss: 1.1608 - val_acc: 0.6824
Epoch 18/25
97/97 [==============================] - 328s 3s/step - loss: 0.0477 - acc: 0.9856 - val_loss: 1.5155 - val_acc: 0.6706
Epoch 19/25
97/97 [==============================] - 327s 3s/step - loss: 0.0748 - acc: 0.9825 - val_loss: 1.1204 - val_acc: 0.7235
Epoch 20/25
97/97 [==============================] - 324s 3s/step - loss: 0.0498 - acc: 0.9866 - val_loss: 1.2369 - val_acc: 0.6706
Epoch 21/25
97/97 [==============================] - 323s 3s/step - loss: 0.0736 - acc: 0.9876 - val_loss: 1.1987 - val_acc: 0.6706
Epoch 22/25
97/97 [==============================] - 325s 3s/step - loss: 0.0691 - acc: 0.9886 - val_loss: 1.1737 - val_acc: 0.7059
Epoch 23/25
97/97 [==============================] - 325s 3s/step - loss: 0.1011 - acc: 0.9711 - val_loss: 1.2466 - val_acc: 0.6882
Epoch 24/25
97/97 [==============================] - 327s 3s/step - loss: 0.0756 - acc: 0.9814 - val_loss: 1.5177 - val_acc: 0.6588
Epoch 25/25
97/97 [==============================] - 327s 3s/step - loss: 0.0480 - acc: 0.9866 - val_loss: 1.3861 - val_acc: 0.7353
```

CHAPTER-10

ADVANTAGES & DISADVANTAGES

10.1 ADVANTAGE:

- ✓ We detect the damage and estimate the cost with percentage and type in it.
- ✓ We provide the solution more accurate.
- ✓ Assured for financial liability.

10.2 DISADVANTAGE:

- ✓ Internet needed.
- ✓ It is not accurate when the image clarity is low.

CHAPTER-11

CONCLUSION

In this proposed system 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. In this proposed project a VGG16 based solution for damage detection; manage the problem of vehicle damage analysis, prediction of car damage location , severity of the damage and estimate the cost. This project carries out lot of functions in a one package. The system will definitely help the insurance companies to analyze the car damage a lot more successful and well organized. Simply by send the image of the car, the system will analyze the given image and show if there is any kind of damage to the vehicle along with the location of the damage , the severity of the damage and cost of damage is displayed as a output.

CHAPTER-12

FUTURE SCOPE

In this project, we proposed a method for efficient damage detection and cost estimation for vehicle used for insurance companies. With the application of AI the data can be stored and retrieved from anywhere. In this proposed work, the damage detection is for the outer surface of the vehicle. hence in future it can be automated for detection of interior damage in vehicle and also about the service provided for the vehicle. In the future, AI could be used to create self-driving cars, as well as cars that can communicate with each other and with other road users . Using computer vision, natural language processing, and robotic automation, manufacturers are producing vehicles that are safer and more comfortable. These vehicles come equipped with computer technology and connectivity that can better understand road and weather conditions, behavior of other drivers, and traffic.

CHAPTER-13

APPENDIX

Github : http://bitly.ws/x24A

Demo Link : http://bitly.ws/x25g