DHANALAKSHMI SRINIVASAN INTITUTE OF TECHNOLOGY, SAMAYAPURAM, TRICHY.

Project Report

Date	18 NOVEMBER 2022
Team ID	PNT2022TMID46154
Project Name	Project - Intelligent Vehicle Damage Assessment and Cost Estimator for Insurance Companies

Intelligent Vehicle Damage Assessment & Cost Estimator for Insurance Companies

1. INTRODUCTION

1.1Project Overview

Use computer vision and deep learning techniques to accuratelyclassify vehicle damage to facilitate claims triage by training convolution neural networks.

1.2Purpose

The rapidly expanding automobile industry highly backs the equally fast-growing auto insurance market. Although until now this industry has been solely based on traditional ways to make repair claims. In case of an unfortunate accident, the claims for the car damage needs to be filed manually. An inspector is required to physically analyse the vehicles to assess the damage and obtain a cost estimate. In such situation, there is also the possibility of inaccurate settlements due to human errors. Automating such a process with the help of machine learning and remote usage would make the process a lot more convenient for both sides of the damage, increasing productivity of the insurance carrier and satisfaction of the customer.

While the technology is yet to achieve the highest possible levels of accuracy, above is a proof of concept of the application of Deep Learning and Computer Vision into automating the damage assessments by building and training Convolution Neural Networks.

Solution

To automate such a system, the easiest method would be to build a Convolution Neural Network model capable of accepting images from the user and determining the location and severity of the damage. The model is required to pass through multiple checks that would first ensure that given image is that of a car and then to ensure that it is in fact damaged. These are the gate checks before the analysis begins. Once all the gate checks have been validated, the damage check will commence. 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.
The model accepts an input image from the user and processes it across 4 stages:
1. Validates that given image is of a car.

2. Validates that the car is damaged.

3. Finds location of damage as front, rear, or side

4. Determines severity of damage as minor, moderate, or severe

The model can also further be improved to:
1. Obtain a cost estimate

2. Send assessment to insurance carrier

3. Print documentation

2.LITERATURE SURVEY

2.1 Existing problem

1. The field of Computer Vision is yet developing and not mature enough to deal with modular phone camera quality images. Angle, lighting, resolution are factors that can easily cause major disruptions in image classification.

2. Car insurance settlement claims require near perfect accuracy to ensure the customer is not frauded in the process. Such models would be required to be trained on humongous datasets which are highly difficult to procure.

3. To run such heavy datasets to ensure maximum accuracy would be imposed by hardware restriction. Storing, training, and deploying such heavy datasets over the cloud would require expensive architecture.

4. While the computer can avoid human errors, there are often situation that would require such a model to flag for human assistance.

5. Systems running on the Cloud, especially those dealing monetary data are also heavily susceptible to cyber risks and require heavily structured frameworks to ensure customer data security.

6. Such a process will require a certain level of manual control and filter to avoid flooding of fraudulent insurance claims.

2.2 Ideation & Brainstorming

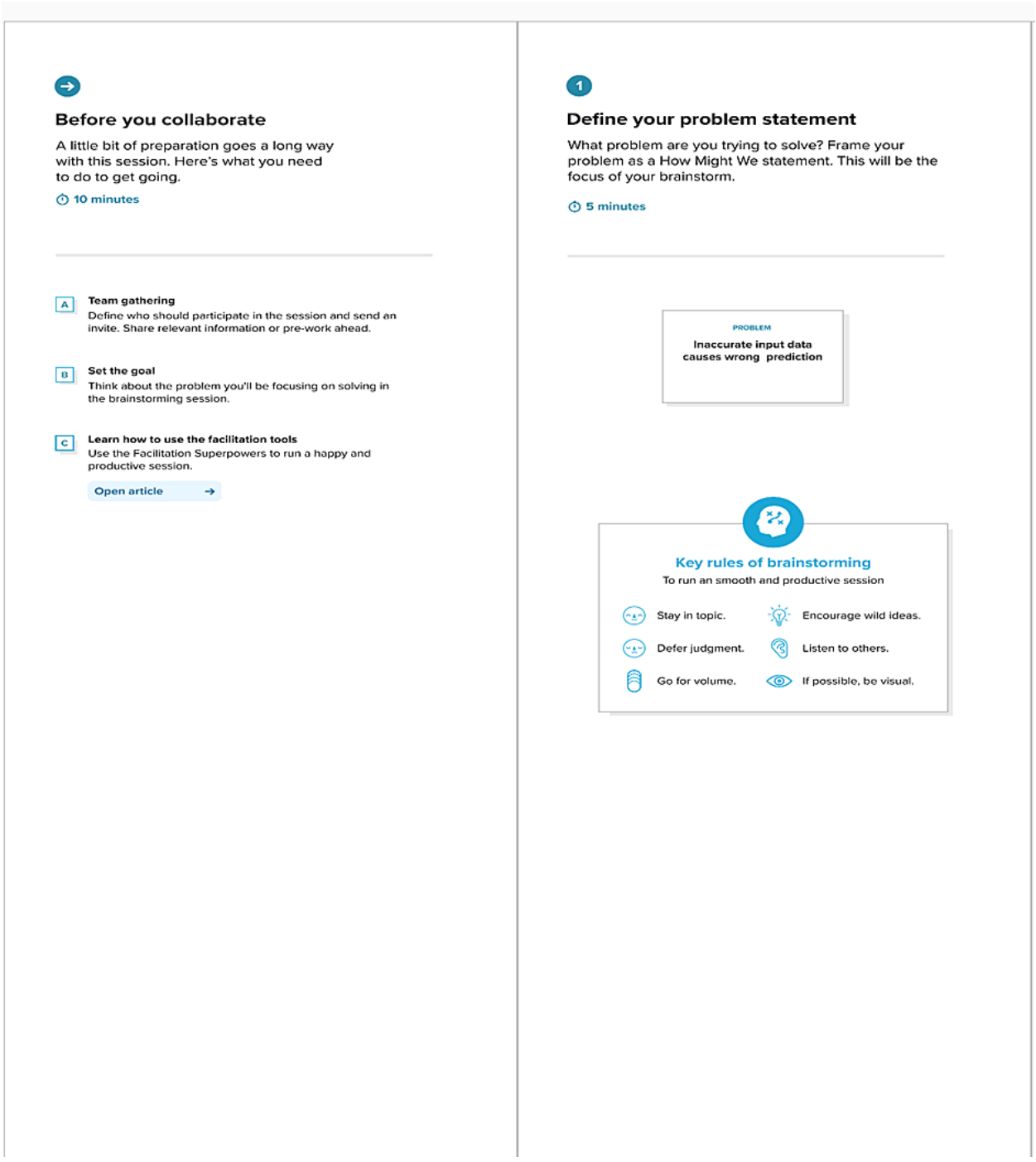
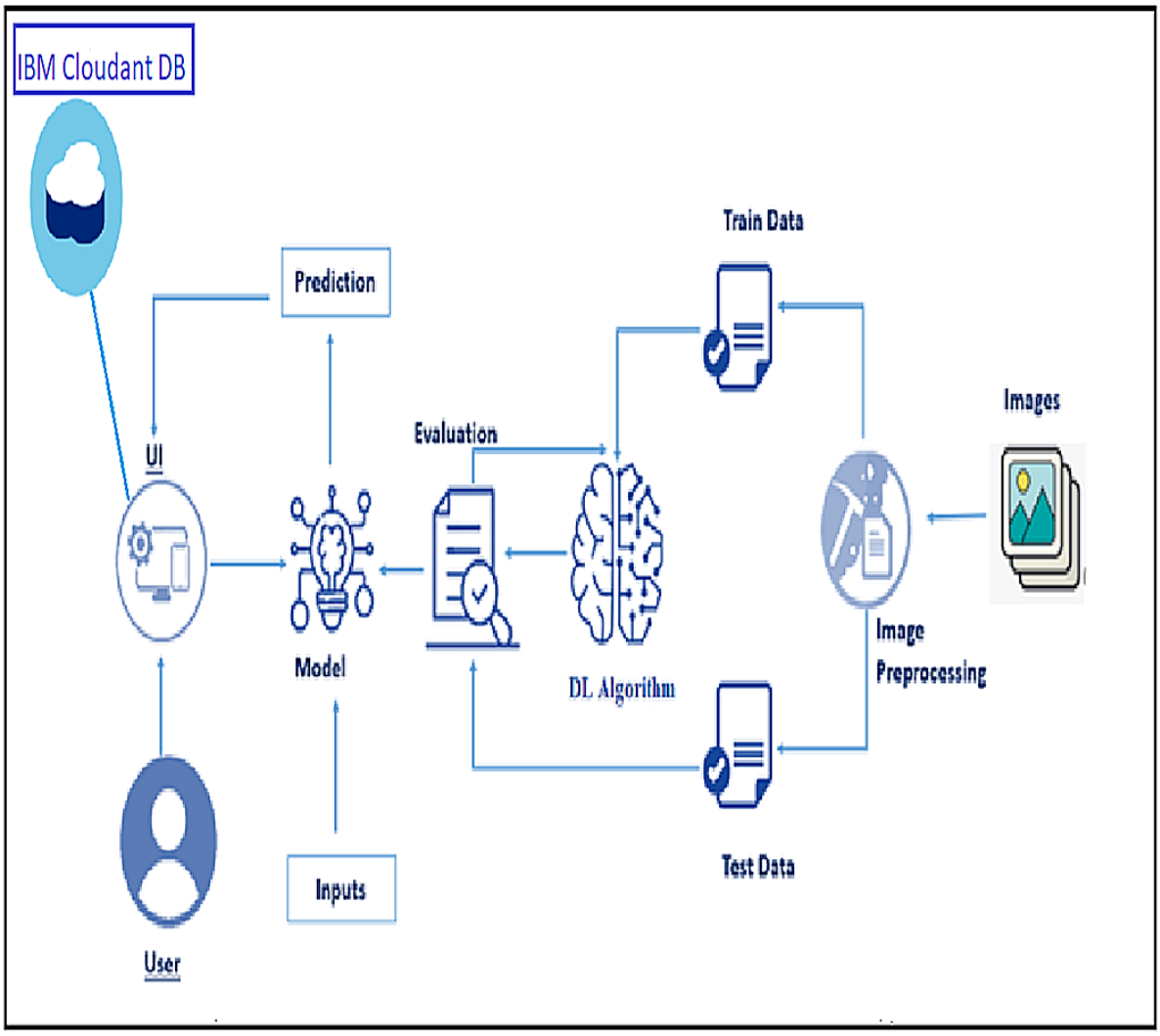


Fig.Team Gathering, Collaboration and Select the Problem Statement

2.3 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	To estimate the cost of the damaged vehicles for insurance companies
1.	Idea / Solution description	● A machine learning application helps to find the cost of the damaged vehicles by using the data sets feed to the program ● Consumer can easily identify the state of the vehicle that with cost price to required
1.	Novelty / Uniqueness	● Easier recording and reporting ● Increased work efficiency ● Easy of use
1.	Social Impact / Customer Satisfaction	● Increased customer satisfaction ● Remote monitoring ● Help to make on spot estimation
1.	Business Model (Revenue Model)	● It is efficient method that provides quick results to the customers
1.	Scalability of the Solution	● It estimated calculations depends on the number of data sets used in the program

2.4 Problem Solution fit



Our system architecture is built around the following modules:

1. User Input: User submits image containing the damage.

2. Gate 1: Checks to ensure the submitted image contains a car.

3. Gate 2: Checks to ensure the submitted image of car is damaged avoiding fraudulent claims.

4. Location Assessment: Tests image against the pre-trained model to locate damage

5. Severity Assessment: Tests image against pre-trained models to determine the severity of damage.

6. Results: The results are sent back to the user and third party

Tools and Frameworks Used

Data Set Collection:

1. Google Images – data source

2. Stanford Car Image Dataset – data source

3. Import.io – online web data scraper

Model Development:

1. TensorFlow – Deep Learning Library

2. Keras – Deep Learning Library

3. NumPy – Scientific numerical calculations library

4. Scikit-learn – Machine learning algorithms tools

Web Development:

1. Flask – Python web framework

2. Bootstrap – HTML, CSS, JavaScript framework

Development Environment:

1. PyCharm IDE – Python program development environment

2. Jupyter Notebooks – web application for interactive data science and scientific computing

3. Anaconda Virtual Environments – python virtual environment application

4. REQUIREMENT ANALYSIS

4.1 Functional requirement:

Libraries Used:

1. numpy

2. pandas

3. matplotlib

4. sklearn

5. seaborn

6. pickle

7. IPython

8. collections

9. h5py

10. json

11. Urllib

Improving The Model

1. The data set used in this application consisted of around 1500 images for the first gate check, while the classification models were trained on only 400 images per class,

while the validation dataset had approximately 75 to 100 images each class. Such a model will have low accuracy.

2. With a wider range of data set featuring multiple components of the car, the model can also be trained to identify what components are damaged, also classifying the varying degree of damage of each.

3. With a highly expansive dataset containing the make, model, year of the car and the possible cost estimates for the varying degrees of damage, the model can also predict the value for the user, before he submits the more advanced and detailed assessment for evaluation.

4. Using more secure and durable hardware, the entire system can be built on the Cloud to run remotely and from the user's cellular device itself.

5. The application can also be updated to recommend the user of policies pertaining to the specific accounts and other insurance benefits.

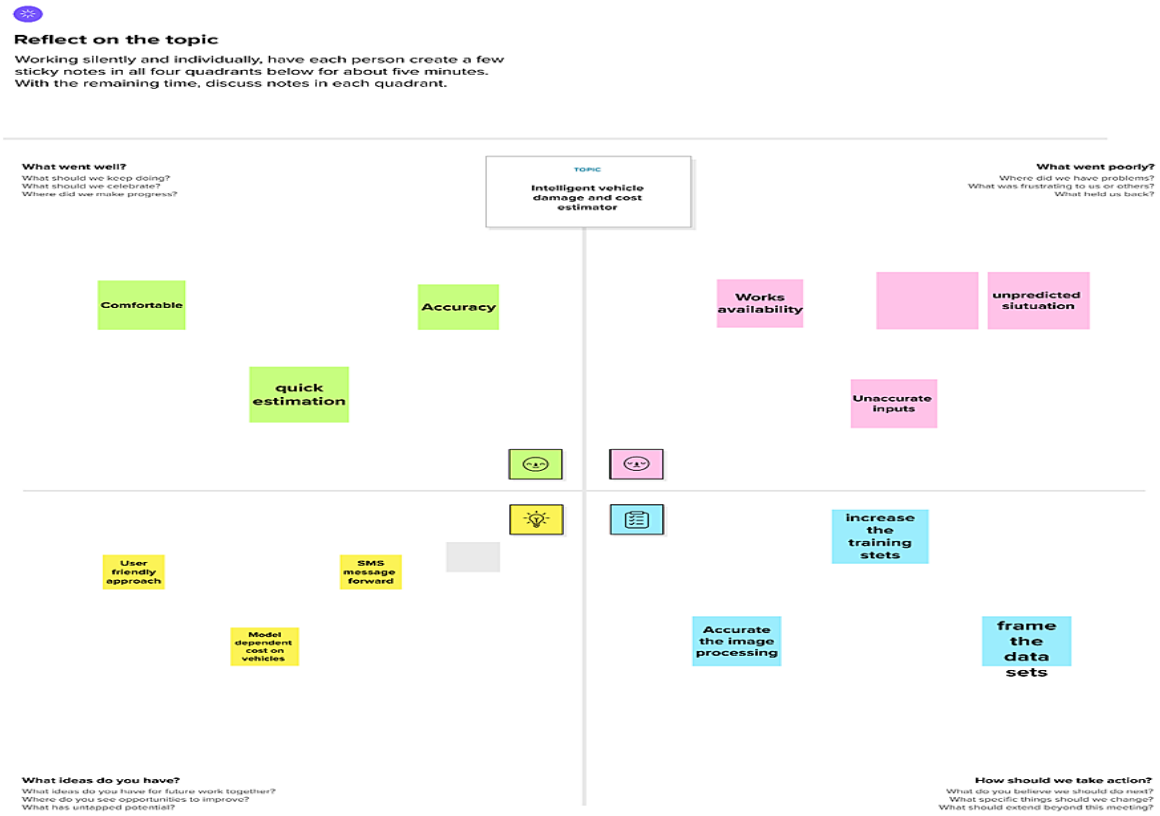
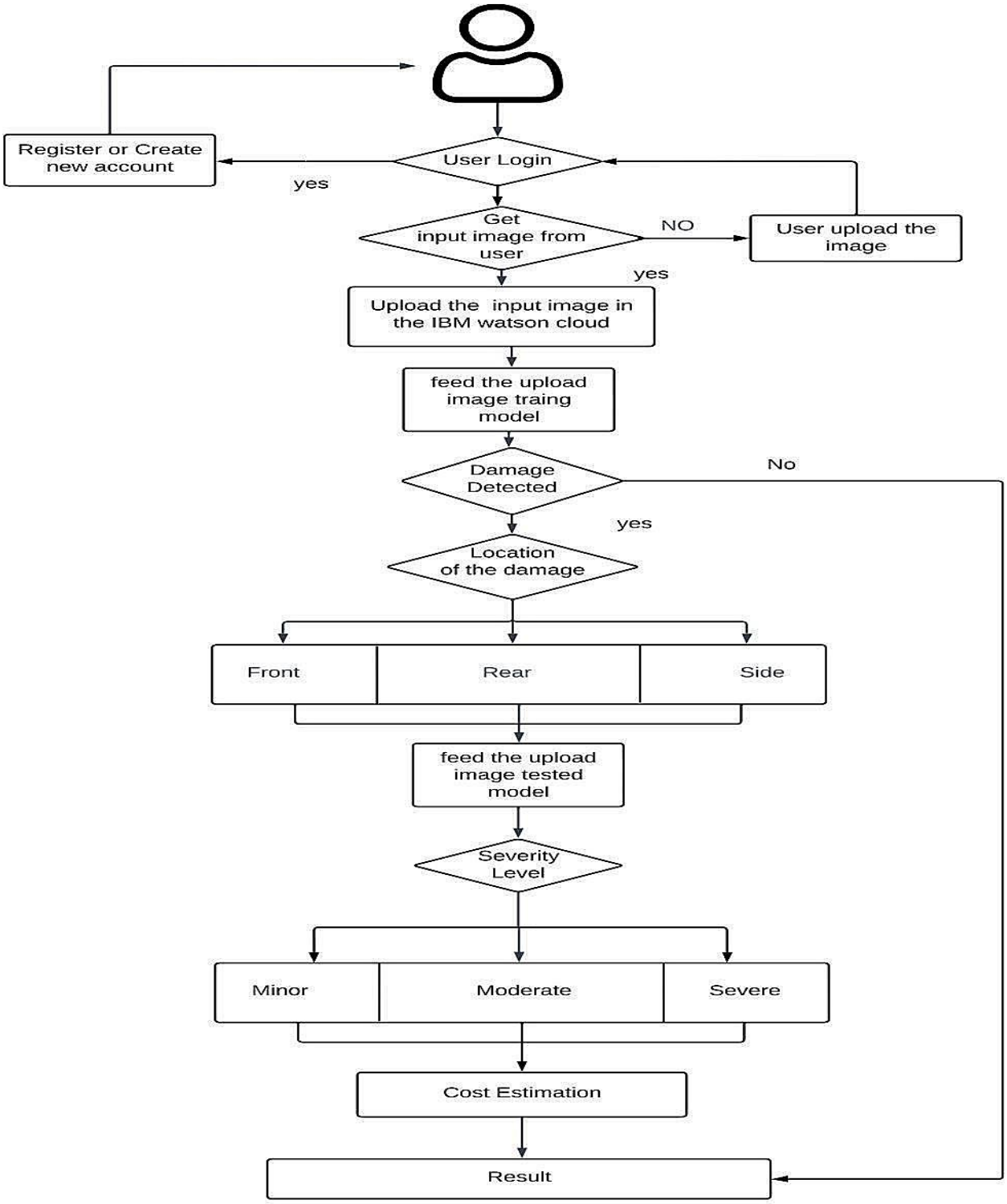


Fig.Empathy Map Canvas

5. PROJECT DESIGN:

5.1 Data Flow Diagrams:



5.2 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Release

Customer (Mobileuser)	Registration	USN-1	As a user, I can register for the application by entering my email, password, and confirmingmy password.	I can accessmy account / dashboard by entering valid credentail s	High	Sprint-1
CustomerDetails	Login	USN-2	As a user,I will receive confirmatio n email once I have registered for the application	I can receive confirmation email & click confirm	High	Sprint-1
Customer Uses	Dashboard	USN-3	As a user,I can register for the application through Facebook	I can register & access the dashboard with Facebook Login	Low	Sprint-4
CustomerOption s	Details about insurance companie s	USN-4	As a user,I can register for the application through Gmail	I can register & access the dashboard with Facebook Gmail	Mediu m	Sprint-1
Customer usage	Login	USN-5	As a user, I can log into the application byentering email & password	I can log in and view my dashboard at my demand onany time	High	Sprint-1
Customer needs to do	Dashboard	USN-6	As a user I must capture images of my vehicle and upload it into theweb portal	I can capture the entire vehicle and upload	High	Sprint-2
Customer (Webuser)	Details about estimated cost basedon damage	USN-7	As a user I must receive a detailed report of the damages present in the vehicle and the cost estimated	I can get the estimated insurance cost	High	Sprint-3

Customer Care Executive	Details about Estimated cost Basedon damage	USN-8	As a user, I need to get support from developers in case of queries and failure of service provided	I can have smooth user experience s and all the issues raisedis sorted	Mediu m	Sprint-4
Administrator	Details about Estimated cost Basedon damage	USN-9	We need to satisfy the customer needs in an efficient way and make sure any sort of errorsare fixed	I can finish the work without any problems	High	Sprint-4

6. PROJECT PLANNING & SCHEDULING

Product Backlog, Sprint Schedule, and Estimation :

Use the below template to create productbacklog and sprint schedule

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, and password, and confirming my password.	2	High	team member 2, team member 4
Sprint-1	Login	USN-2	As a user, I will receivea confirmation emailonce I haveregistered for the application	1	High	team member 3 and team member 4
Sprint-1	Dashboard	USN-3	As a user, I can register for the application through Facebook	1	High	team member1 and team member 3

Sprint-2	Details about insurance company	USN-4	As a user,I can register for the application through Gmail	1	low	team member 3
Sprint-1	repeated logins and logout	USN-5	As a user, I can log intothe application by enteringemail & password	2	medium	team member 1 and team member 2
Sprint-2	Webpage	USN-6	As a user I must capture imagesof my vehicleand upload it into the web portal.	1	high	team member 4
Sprint-3	Details about estimated cost based on damage	USN-7	As a userI must receivea detailed reportof the damages present in the vehicle and the costestimated	1	high	team member 1, team member 2,teammembe r

Sprint	Functional Requirement (Epic)	User Story Number	User Story / Task	Story Points	Priority	Team Member s
Sprint-4	Provide friendly and efficient customer support and sort outthe queries.	USN-8	As a user, I need to get supportfrom developersin case of queries and failure of service provided	1	high	team member1and team member 3
Sprint-4	overviewthe entire process and act as abridge betweenuser and developer	USN-9	We need to satisfy the customer needsin an efficient way and make sure any sort of errorsare fixed	2	high	team member 2 and team member 4

Project Tracker, Velocity & Burndown Chart:

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

Velocity:

Imagine wehave a 10-day sprint duration, and the velocityof the team is 20 (points per sprint). Let's calculate the team's averagevelocity (AV) per iteration unit (story points per day)

$$AV = \frac{sprint\ duration}{velocity} = \frac{20}{10} = 2$$

7. CODING & SOLUTIONING:

7.1 Feature 1

Dashboard.html

```
<html>
<head>
<title>
Intelligent Vehicle Damage Assessment and Cost Estimator for insurance Companies
</title>
<style type="text/css"> #topmenu {
width: 100%;
background-color: 312D2D; height: 50px;
}
#hedder {
color: white; padding-top: 13px; padding-left: 60px;
}
#home { float: right;
padding-top: 13px; padding-right: 50px;
color: rgb(222, 216, 216);
font-size: medium;
}
#login { float: right;
padding-top: 13px; padding-right: 50px; color: rgb(222, 216, 216);
font-size: medium;
}
#register { float: right;
padding-top: 13px; padding-right: 50px; color: rgb(222, 216, 216); font-size: medium;
}
```

```
#prediction { float: right;
padding-top: 13px; padding-right: 50px; color: rgb(222, 216, 216); font-size: medium;
}
#about {
text-align: center; padding-top: 10%; color: gray;
font-size: 20px;
}
#footer { width: 99%;
background-color: 312D2D;
height: 50px; position: absolute; bottom: 1%;
}
#textcontent { color: white; font-size: 15px;
padding-left: 18%;
padding-top: 1%;
}
#logo {
margin-top: -1.5%;
margin-right: 28%; float: right;
}
.container {
display: flex;
}
#vehicle_img { margin-top: 4%;
margin-left: 5%;
}
#topic_content {
font-family: Georgia; font-size: large; padding-top: 4%; color: DodgerBlue; padding-right: 10%;
}
.pname1 {
margin-top: 3%;
font-weight: 600 !important; font-size: large;
color: SlateBlue !important;
}
.login_prediction { display: flex;
}
#login_details { padding-left: 10%;
```

```
}
#signin {
text-align: center; padding-bottom: 10%;
font-size: large;
}
#predict {
text-align: right;
}
#blink {
color: red;
animation: blinker 0.9s linear infinite; font-weight: bold;
}
@keyframes blinker { 50% {
opacity: 0;
}
}
</style>
</head>
<body onload="flashMessage()">
<script>
function flashMessage(){ if("{{flash_message}}" == "True"){
alert("account created successfully")
}
if("{{flash_message}}" == "Fals"){ alert("invalid credentials")
}
if("{{flash_message}}" == "Fal"){ alert("Logged in successfully")
}
}
</script>
<div id="topmenu">
<div id="prediction">
<a href="{{ url_for('prediction') }}" style="color: white;text-decoration:
none;">prediction</a>
</div>
<div id="register">
<a href="{{ url_for('register') }}" style="color: white;text-decoration:
none;">Register</a>
</div>
```

Login

Home

Intelligent Vehicle Damage Assessment & Cost Estimator for Insurance Companies

<img src="/static/images/bsi1.png" alt="Damaged Vehicle" width="80%"
height="auto"
/>

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.

<marquee class="pname1" direction="left" behavior="scroll" scrollamount="10"
>Login to know more about the level of damage and cost estimation</marquee
>

```
<div class="login_prediction">
<div id="login_details" style="padding-top: 5%">
<div id="signin">
<b><i>Log in</i></b>
</div>
<form action="dashboard" method="POST">
<input type="text"
name="email" id="email"
placeholder="Enter registered email ID" style="width: 150%; height: 35px"
/><br />
<br />
<input type="password"
name="password" id="password" placeholder="Enter Password"
style="width: 150%; height: 35px"
/><br />
<br />
<input type="submit" name="submit" id="submit" value="Login" style="
width: 150%; height: 35px;
text-align: center;
background-color: black;
color: white;
"
/>
</form>
</div>
<div id="predict" style="text-align: center;margin-left: 25%;">
<p>
<b>To predict the cost for the damage in vehicle and percentage of damage in car </b>
</p>
<!--<p id="blink">Click Here!</p>-->
</div>
</div>
<div id="footer">
<div id="textcontent">Copyright</div>
<div id="logo">
```

<img src="/static/images/twitter.jpg"
height="28px" width="28px"
style="border-radius: 18%; margin-right: 40px"
/>
<img src="/static/images/linkedin.jpg" height="28px"
width="28px"
style="margin-left: 30px; border-radius: 18%"
/>

</body>
</html>
Prediction.html
<html>
<head>
<title>index</title>
<style type="text/css"> #topmenu {
width: 100%;
background-color: 312D2D;
height: 50px;
}
#hedder { color: white;
padding-top: 13px; padding-left: 60px;
}
#home { float: right;
padding-top: 13px; padding-right: 50px; color: rgb(222, 216, 216); font-size: medium;
}
#login { float: right;
padding-top: 13px;
padding-right: 50px;
color: rgb(222, 216, 216);
font-size: medium;
}
#register { float: right;
padding-top: 13px; padding-right: 50px;
color: rgb(222, 216, 216);
font-size: medium;

```
}
#prediction { float: right;
padding-top: 13px; padding-right: 50px; color: rgb(222, 216, 216); font-size: medium;
}
#about {
text-align: center; padding-top: 10%; color: gray;
font-size: 20px;
}
#content {
padding-top: 50px; padding-left: 40px; padding-right: 40px; font-size: large;
}
#footer {
width: 99%;
background-color: 312D2D; height: 50px;
position: absolute; bottom: 1%;
}
#textcontent { color: white; font-size: 15px;
padding-left: 18%;
padding-top: 1%;
}
#logo {
margin-top: -1.5%;
margin-right: 28%;
float: right;
}
</style>
</head>
<body onload="flashMessage()">
<div id="topmenu">
<div id="login">
<a href="{{ url_for('logout') }}" style="color: white;text-decoration: none;">Logout</a>
</div>
<div id="home">
<a href="{{ url_for('dashboard') }}" style="color: white;text-decoration: none;">Home</a>
```

```
        </div>
        <div id="hedder">
        Intelligent Vehicle Damage Assessment & Cost Estimator for Insurance
        Companies
        </div>
        </div>
        <form action="prediction" method="POST" enctype="multipart/form-
data">
        <input type="file" id="myFile" name="myFile">
        <input type="submit">
        <script>
        function flashMessage(){
        if("{{flash_message}}" == "True"){
        // alert("invalid credentials")
        // const im = document.createElement('img');
        // im.src = "{{url_for('static', filename='imagedata/save.png')}}";
        // im.height = "200px";
        // im.width = '200px';
        // im.alt = 'hello world'
        //                 document.getElementById('about').appendChild(im);
document.getElementById('image').src = 'static/imagedata/save.png'; const e =
document.getElementById("qwerty");
        const para = document.createElement("p");
        const node = document.createTextNode("The prediction of vehicle is : |
{{value}} |"); para.appendChild(node);
        e.appendChild(para);
        }
        }
```

7.2 Database Schema :

```
#Importing
Libraries
import re
import numpy as np
import os
from flask import Flask, app, request, render_template
from keras import models
```

```
from keras.models import load_model
from keras.preprocessing import image
from tensorflow.python.ops.gen_array_ops import concat
from keras.applications.inception_v3 import preprocess_input
import requests
from flask import Flask, request, render_template, redirect, url_for
from cloudant.client import Cloudant
#Create Database
client = Cloudant.iam('00cba18f-2150-4961-9102-f29b9aee35debluemix','
ht_ByiEjrGeaitIZJTC-ri5_8Oq-dxTNHLGho1mpt0d5', connect=True)
my_database = client.create_database('my_database')
#Loading the Model
model1 = load_model('Model/level.h5')
model2 = load_model('Model/body.h5')
app = Flask(__name__)
@app.route('/')
def index():
return render_template('index.html')
@app.route('/index.html')
def home():
return render_template('index.html')
@app.route('/register.html')
def register():
return render_template('register.html')
@app.route('/afterreg', methods=['POST'])
def afterreg():
x = [x for x in request.form.values()]
print(x)
data = {
'_id': x[1],
'name': x[0],
'psw': x[2]
}
print(data)
query = {'_id': {'$eq': data['_id']}}
docs = my_database.get_query_result(query)
print(docs)
print(len(docs.all()))
```

```
if(len(docs.all())==0):
url = my_database.create_document(data)
response = request.get(url)
return render_template('login.html', pred="Registration Successful,
Please login using your details")
else:
return render_template('register.html', pred="You are already a
member, Please login using your details")
@app.route('/login.html')
def login():
return render_template('login.html')
@app.route('/afterlogin', methods=['POST'])
def afterlogin():
user = request.form['_id']
passw = request.form['psw']
print(user,passw)
query = {'_id': {'$eq': user}}
docs = my_database.get_query_result(query)
print(docs)
print(len(docs.all()))
if(len(docs.all())==0):
return render_template('login.html', pred="The Username is not
found")
else:
if((user==docs[0][0]['_id'] and passw==docs[0][0]['psw'])):
return redirect(url_for('prediction'))
else:
print('Invalid User')
@app.route('/logout.html')
def logout():
return render_template('logout.html')
@app.route('/prediction.html')
def prediction():
return render_template('prediction.html')
@app.route('/result')
def res():
if request.methods=="POST":
f=request.files['image']
```

```
basepath=os.path.dirname(__file__)
filepath=os.path.join(basepath,'uploads',f.filename)
f.save(filepath)
img=image.load_img(filepath,target_size=(256,256))
x=image.img_to_array(img)
x=np.expand_dims(x,axis=0)
img_data=preprocess_input(x)
prediction1=np.argmax(model1.predict(img_data))
prediction2=np.argmax(model2.predict(img_data))
index1=['front','rear','side']
index2=['minor','moderate','severe']
result1 = index1[prediction1]
result2 = index2[prediction2]
if(result1 == "front" and result2 == "minor"):
value = "3000 - 5000 INR"
elif(result1 == "front" and result2 == "moderate"):
value = "6000 - 8000 INR"
elif(result1 == "front" and result2 == "severe"):
value = "9000 - 11000 INR"
elif(result1 == "rear" and result2 == "minor"):
value = "4000 - 6000 INR"
elif(result1 == "rear" and result2 == "moderate"):
value = "7000 - 9000 INR"
elif(result1 == "rear" and result2 == "severe"):
value = "11000 - 13000 INR"
elif(result1 == "side" and result2 == "minor"):
value = "6000 - 8000 INR"
elif(result1 == "side" and result2 == "moderate"):
value = "9000 - 11000 INR"
elif(result1 == "side" and result2 == "severe"):
value = "12000 - 15000 INR"
else:
value = "16000 - 50000 INR"
return render_template('prediction.html',prediction=value)
if __name__=="__main__":
app.run(debug = False,port = 8080)
```