



INTELLIGENT VEHICLE DAMAGE ASSESSMENT AND COST ESTIMATOR FOR INSURANCE COMPANIES

IBM PROJECT REPORT

SUBMITTED BY

ATSHAYAA V - 962319104028

CHITHAMBARA SELVI G - 962319104035

HARIHARAN S V - 962319104044

GOKUL B - 962319104040

in partial fulfillment for the award of the degree

of

BACHELOR OF ENGINEERING

in

COMPUTER SCIENCE AND ENGINEERING

AMRITA COLLEGE OF ENGINEERING AND TECHNOLOGY

ERACHAKULAM, NAGERCOIL.

ANNA UNIVERSITY::CHENNAI 600 025

JUNE 2022

CERTIFICATE OF EVALUATION

COLLEGE NAME : AMRITA COLLEG OF ENGINEERING AND
TECHNOLOGY

BRANCH : COMPUTER SCIENCE AND ENGINEERING

SEMESTER : VII

TITLE : INTELLIGENT VEHICLE DAMAGE ASSESSMENT
AND COST ESTIMATOR FOR INSURANCE COMPANIES

TEAM ID : PNT2022TMID51924

STUDENT NAMES	REGISTRATION NUMBER	SUPERVISOR
ATSHAYAA V	962319104028	Mrs S L JOTHI LAKSHMI
CHITHAMBARA SELVI G	962319104035	
HARIHARAN S V	962319104044	
GOKUL B	962319104040	

The report of this project is submitted by the above students in partial fulfillment for the award of Bachelor of Engineering Degree, in Computer Science and Engineering of Anna University are evaluated and confirmed to the reports of the work done by the above students.

MENTOR

EVALUATOR

ACKNOWLEDGEMENT

First and foremost we would like to express our sincere gratitude to our respected Founder Amma, Mata Amritanandamayi Devi and Chairman Mr. K S Ramasubban IAS(Retd) for their blessings and grace in making our project great success.

We would like to place a record with deep sense of gratitude to our Honorable Principal Dr. T Kannan for having given us the opportunity to pursue B.E., course in this prestigious institution.

We would like to express our sincere thanks to our beloved Head of the Department Dr. P M Siva Raja and Project Coordinator Mrs. S L Jothi Lakshmi, for creating a supportive and a model environment for us to work and build up our innovative skills. We also thank our project Mentor Mr. Anant Raj I V for the kind encouragement and moral support, who has been a constant source of inspiration to us.

We wish to express our sincere sense of gratitude to Dr. P M Siva Raja, Head, Department of Computer Science and Engineering and to our Project Guide who enabled us to complete our project successfully.

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.

1.2 Purpose

The aim of this 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 be it dent 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 a used car.

2. LITERATURE SURVEY

2.1 Existing problem

2.1.1. TITLE: Convolutional Neural Networks for vehicle damage detection, 2021

AUTHOR NAME: R.E. Ruitenbeek Vehicle damage is becoming an increasing liability for shared mobility providers. The high number of driver handovers necessitates the use of an accurate and quick inspection system capable of detecting minor damage and categorising it. To address this, a damage detection model is created that locates vehicle damages and categorises them into twelve groups. To improve detection performance, multiple deep learning algorithms are used, and the effect of various transfer learning and training strategies is evaluated. The final model, which was trained on over 10,000 damage photos, can detect minor defects in a variety of environments, including water and dirt. A performance evaluation using domain

experts reveals that the model performs comparably. Furthermore, the model is tested in a specially designed light street, demonstrating how strong reflections complicate detection performance.

2.1.2. TITLE: Deep Learning Based Car Damage Detection, Classification and Severity

AUTHOR NAME: Ritik Gandhi¹, 2021 Because it is a manual procedure, resolving a claim in the accident insurance sector takes time, and there is a gap between the ideal and real settlement. We are using deep learning models to not only speed up the process, but also to deliver better customer service and boost insurance company profitability. In this paper, we use multiple pre trained models such as VGG 16, VGG 19, Resnet50, and DENSENET to choose the top performing models. We first use the Resnet50 model to determine whether or not the automobile is damaged, and if it is, we utilise the WPOD-net model to identify the licence plate. The YOLO model is used to detect the affected region. Finally, the damage severity is implemented using the DENSENET model. We discovered that transfer learning outperforms fine-tuning after applying multiple models. Furthermore, we present a framework that incorporates all of this into a single application, assisting in the automation of the insurance sector.

2.2 References

[1]. R.E. Ruitenbeek, Convolutional Neural Networks for vehicle damage detection, 2021

[2]. Ritik Gandhi¹Deep Learning Based Car Damage Detection, Classification and Severity, 2021

2.3 Problem Statement Definition

Nowadays, insurance companies use AI to analyze the image and to estimate the cost. Let us consider a customer who was met with an accident, and his car has been damaged and he tries to claim insurance from the insurance company in which he holds insurance for his car, He took picture of his car and posts it on the company's site, the site analyses the picture and estimates the cost for the insurance

(The amount to be claimed from the insurance company by the customer), Here is the problem, but the estimated amount is not accurate, the amount is too low or high, this may cause loss to either the customer or the insurance company

Example:



Problem Statement (PS)	I am	I'm trying to	But	Because	Which makes me feel
PS-1	Customer (Insurance holder)	To claim insurance for my car	I'm not getting the accurate amount for the damage caused	Image recognition is poor	Insurance is worthless
PS-2	Employee (insurance company)	Estimate the damage by analyzing the given image and	It's difficult to estimate the cost with the picture	Image recognition is not efficient	Bad and frustrated

3.IDEATION & PROPOSED SOLUTION

3.1 Empathy Map Canvas


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

Example:



3.2 Ideation & Brainstorming

Step-1: Team Gathering, Collaboration and Select the Problem Statement



Brainstorm & idea prioritization

Use this template in your own brainstorming sessions so your team can unleash their imagination and start shaping concepts even if you're not sitting in the same room.

🕒 10 minutes to prepare
🕒 1 hour to collaborate
👤 2-8 people recommended

➔

Before you collaborate

A little bit of preparation goes a long way with this session. Here's what you need to do to get going.

🕒 10 minutes

A **Team gathering**
Define who should participate in the session and send an invite. Share relevant information or pre-work ahead.

B **Set the goal**
Think about the problem you'll be focusing on solving in the brainstorming session.

C **Learn how to use the facilitation tools**
Use the Facilitation Superpowers to run a happy and productive session.

[Open article](#) ➔

1


Define your problem statement

What problem are you trying to solve? Frame your problem as a How Might We statement. This will be the focus of your brainstorm.

🕒 5 minutes

PROBLEM STATEMENT

- Major problem faced by the consumer on an insurance company is not having the idea about the cost of damage.
- Insurance companies are failing to provide the right amount for the car damage and the customer is not able to claim amount for the damage.
- Developing the solution that can able to identify the right cost for the damage would be beneficial for many consumers.



Key rules of brainstorming

To run an smooth and productive session

➤ Stay in topic.

➤ Defer judgment.

➤ Go for volume.

💡 Encourage wild ideas.

👂 Listen to others.

👁️ If possible, be visual.

STEP-2: Brainstorm, Idea Listing and Grouping

2

Brainstorm
Write down any ideas that come to mind that address your problem statement.

10 minutes

TIP

You can select a sticky note and hit the pencil (switch to sketch) icon to start drawing!

ATSHAYA V

Computer vision base estimation for car

With image processing the user gets notified severity of the damage

Neutral network extracting image feature.

Machine Learning allows to predict accurate cost for the damage

Chithambara Selvi G

We can estimate the minor, major and severe damage of the car.

With the help of the image we can detect the dent.

We can estimate the labour cost.

Fake image detection

HARI HARAN S V

With the help of AI we provide easy estimation about damage.

Images are clearly detected using VCG model.

With the help of AI we avoid fraud estimations.

Car model detection.

GOKUL B

We have to acquire the correct information about the damage of car

We have to maintain the user database.

Response time should be quick.

Website consists a feedback note.

3

Group ideas
Take turns sharing your ideas while clustering similar or related notes as you go. In the last 10 minutes, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you can break it up into smaller sub-groups.

20 minutes

Damage Detection

Computer vision base estimation for car

Car model detection.

We have to acquire the correct information about the damage of car

Fake image detection

With the help of AI we provide easy estimation about damage.

Damage Severity

With image processing the user gets notified severity of the damage

With the help of the image we can detect the dent.

We can estimate the minor, major and severe damage of the car.

Damage Severity

Machine Learning allows to predict accurate cost for the damage

Images are clearly detected using VCG model.

Damage Severity

With the help of AI we avoid fraud estimations.

We can estimate the labour cost.

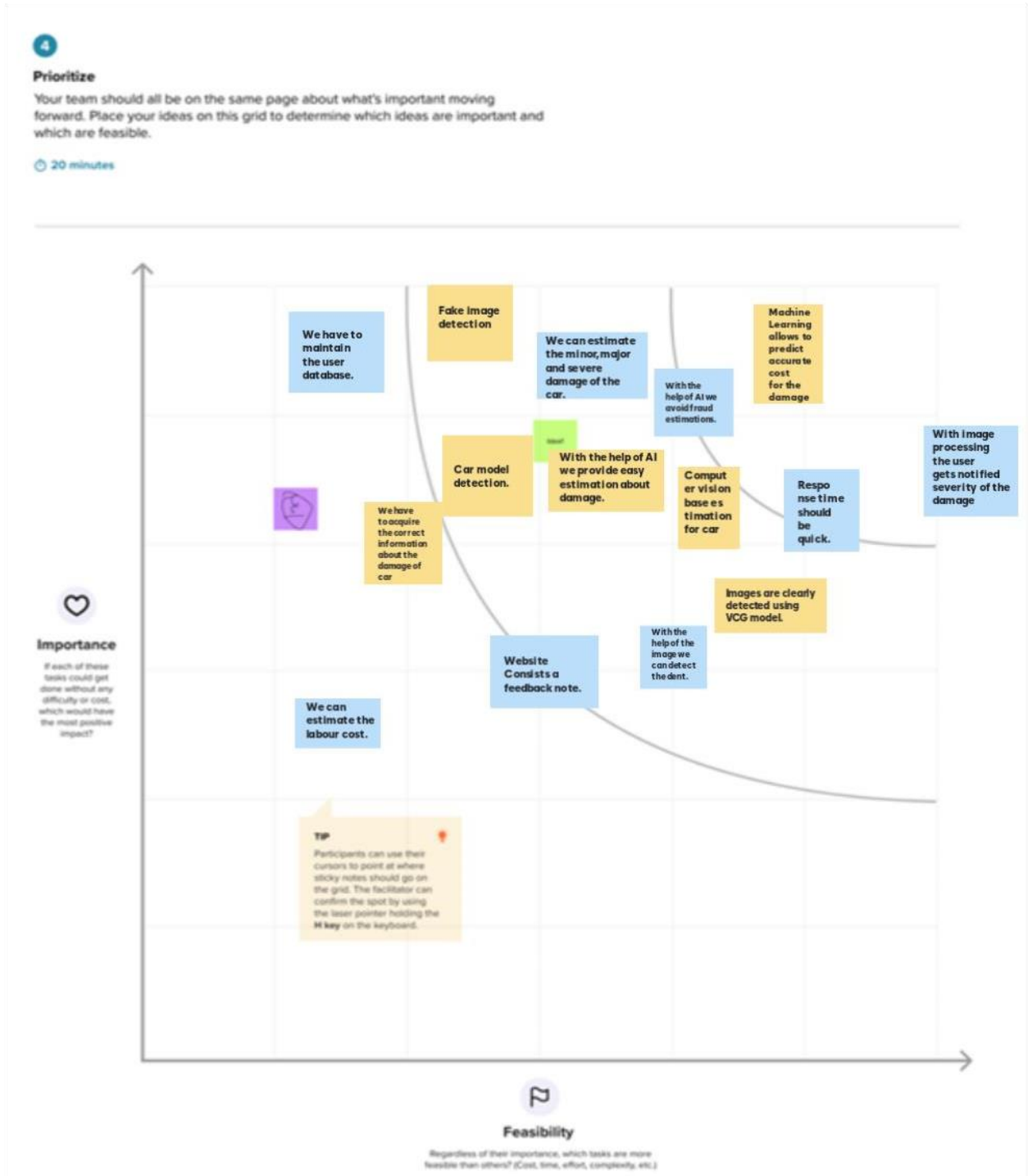
Data & Should have

We have to maintain the user database.

Response time should be quick.

Website consists a feedback note.

Step-3: Idea Prioritization:



3.3 Proposed Solution

S.NO.	PARAMETER	DESCRIPTION
1.	Problem Statement (Problem to be solved)	Nowadays, Insurance Companies faced the greatest problem which is the leakage of the insurance claim. Some of the customers claim an extra amount for the damage to the vehicle through fake bills for the claim. So The Insurance Companies lost most of the amount due to the leakage of the claim.
2.	Idea / Solution description	To solve this problem, we have to develop software that helps to insurance companies. The procedures we design involve creative initiative that will inspire the company has to believe in that software and also the customer..
3.	Novelty / Uniqueness	We applied deep learning-based algorithms, Mask R-CNN , for car damage detection and assessment in real world datasets. The algorithms detect the damaged part of a car and assess its location and then its severity. Initially, it discovers the effect of domain-specific pre-trained CNN models , which are trained by datasets . .
4.	Social Impact / Customer Satisfaction	On the website, customers have to take a photo of the damaged portion of the vehicle and send it to the company to claim the insurance. The process is quicker and they can easily access the website and post the picture on the company's website and estimate the cost for the damage .so we think it is easy for the customers.
5.	Business Model (Revenue Model)	It is more effective than others.It reduces the delay.This helps the customer to get the claim quickly.It has good accuracy.

6.	Scalability of the Solution	Mask R-CNN. Maximum object detection accuracy for training set is approximately 54% (using data augmentation and hyperparameter tuning).
----	-----------------------------	--

3.4 Problem Solution fit

Define CS, fit into CL	1. CUSTOMER SEGMENT(S) CS <div><ul style="list-style-type: none">• People who owns a vehicle• Insurance company</div>	6. CUSTOMER LIMITATIONS <small>EG. BUDGET, DEVICES</small> CL <div><ul style="list-style-type: none">• Not good about Insurance knowledge.• Lack of information</div>	5. AVAILABLE SOLUTIONS <small>PLUSES & MINUSES</small> AS <div>Cost estimation done manually and slow Pros: Confidentiality , less fraud (Trustable) Cons: Process takes huge time span</div>	Explore AS, differentiate
	2. PROBLEMS / PAINS <small>+ ITS FREQUENCY</small> PR <div><ul style="list-style-type: none">• The cost estimated is not accurate• Consumes more time.• Fails to provide the correct value• Waiting time is high</div>	9. PROBLEM ROOT / CAUSE RC <div><ul style="list-style-type: none">• The customer not getting accurate cost for the damage. In this app accurate estimated value can be obtained.</div>	7. BEHAVIOR <small>+ ITS INTENSITY</small> BE <div><ul style="list-style-type: none">• Customer do not know how to take pictures of the damaged part properly• Machine cannot predict accurately</div>	
Identify strong TR & EM	3. TRIGGERS TO ACT TR <div><ul style="list-style-type: none">• Customer doesn't want to fall for any traps. Society wants instant results</div>	10. YOUR SOLUTION SL <div><ul style="list-style-type: none">• "AI based intelligent vehicle damage assessment and Cost Estimator for Insurance Companies;"• Using Mask R-CNN</div>	8. CHANNELS of BEHAVIOR CH <div>ONLINE: Use the website in an improper way</div> <div>OFFLINE: Customers are unaware and tend to fight</div>	Extract online & offline CH of BE
	4. EMOTIONS <small>BEFORE / AFTER</small> EM <div>BEFORE: Customer can't get the exact damage value insurance. AFTER: Customer easily get the exact value for insurance within 24 hours</div>			

4.REQUIREMENT ANALYSIS

4.1Functional requirement

Following are the functional requirements of the proposed solution.

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	User registration	Go to the website Register via email Set a password Reset the password
FR-2	User Confirmation	Confirmation via Email Confirmation via OTP
FR-3	Interface	Good Interface to user to operate
FR-4	Accessing datasets	Details about user Details about vehicle Details about insurance companies
FR-5	Mobile application	AI and camera sensors in the field can be accessed by mobile applications.

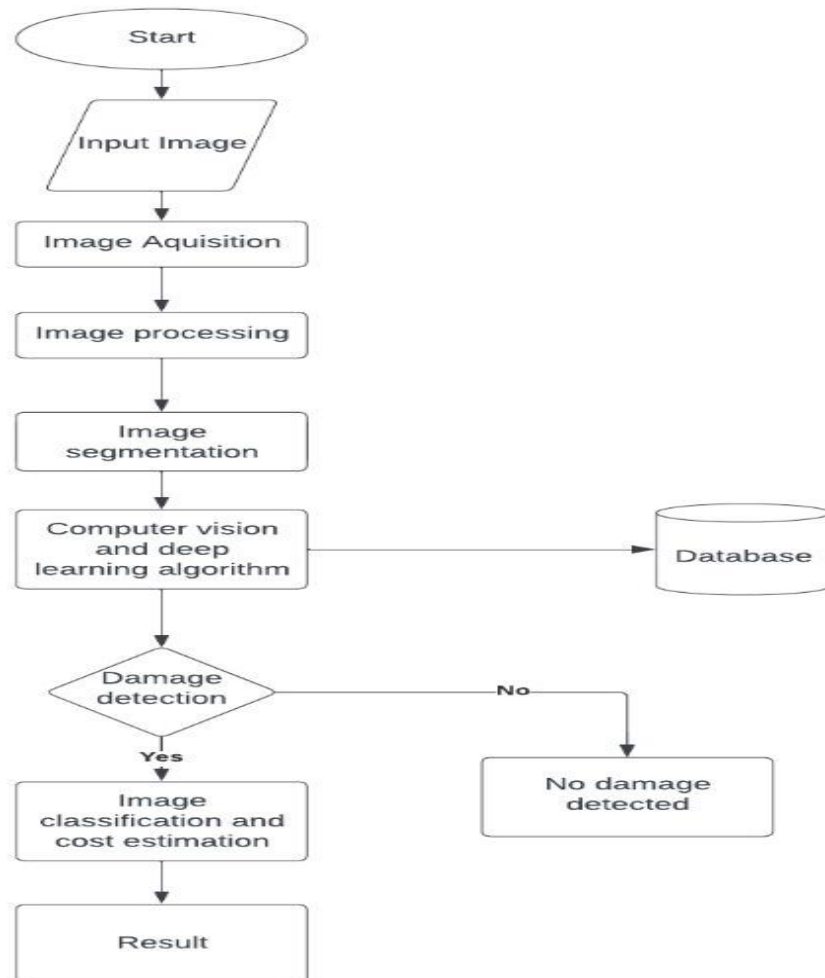
4.2 Non-Functional requirements

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

FR No.	Non-Functional Requirement	Description
NFR-1	Usability	The smart claiming system for vehicle damage insurance in insurance companies
NFR-2	Security	We have designed this project for users to claim insurance faster.
NFR-3	Reliability	This project will help the user to claim the insurance cost based on vehicle damage. It gives the exact value to the user. This helps the user to get the accurate cost without any failure.
NFR-4	Performance	AI devices and sensors are used to indicate to the user to estimate the cost of the vehicle. AI camera scans the damaged vehicle and gives exact cost insurance to the user.
NFR-5	Availability	This application can be accessed from any device browser.
NFR-6	Scalability	This project is more scalability in our present and future uses to estimate the cost accurately to the user.

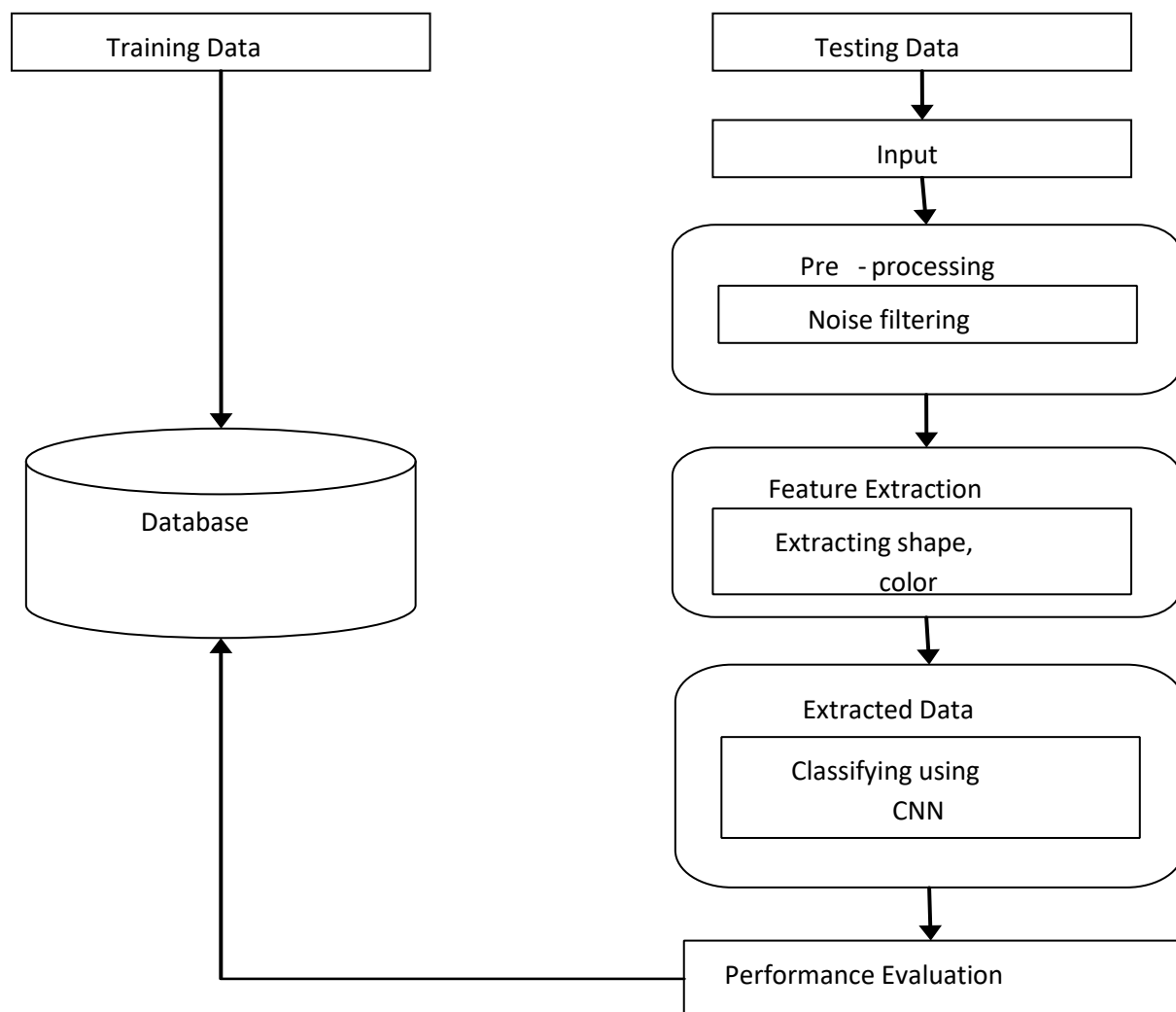
5.PROJECT DESIGN

5.1Data Flow Diagrams



5.1 Solution & Technical Architecture

A system architecture or systems architecture is the conceptual model that defines the structure, behavior, and more views of a system. An architecture description is a formal description and representation of a system, organized in a way that supports reasoning about the structures and behaviors of the system. System architecture can comprise system components, the externally visible properties of those components, the relationships (e.g. the behavior) between them. It can provide a plan from which products can be procured, and systems developed, that will work together to implement the overall system. There have been efforts to formalize languages to describe system architecture, collectively these are called architecture description languages (ADLs).



5.2 User Stories

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	Atshayaa
Sprint-1	Login	USN-2	As a user, I will receive a confirmation email once I have registered for the application	1	High	Gokul
Sprint-1	Dashboard	USN-3	As a user, I can register for the application through Face book	2	High	Hari Haran
Sprint-2	Details about insurance company	USN-4	As a user, I can register for the application through Gmail	2	Low	Chithambara Selvi
Sprint-1	repeated logins and logout	USN-5	As a user, I can log into the application by entering email & password	1	Medium	Atshayaa
Sprint-2	Webpage	USN-6	As a user I must capture images of my vehicle and upload it into the web portal.	2	High	Gokul
Sprint-3	Details about estimated cost based on damage	USN-7	As a user I must receive a detailed report of the damages present in the vehicle and the cost-estimated	2	High	Hari Haran
Sprint-4	Provide friendly and efficient customersupport and sort out the queries.	USN-8	As a user, I need to get support from developers in case of queries and failure of service provided	2	High	Chithambara Selvi
Sprint-4	overview the entire process and act as a bridge between user and developer	USN-9	We need to satisfy the customer needs in an efficient way and make sure any sort of errors is fixed	2	High	Atshayaa

6 PROJECT PLANNING & SCHEDULING

6.1 Sprint Planning & Estimation

Sprint	Total Story Points	Duration	Sprint Start Date	Sprint End Date (Planned)	Story Points Completed (as on Planned End Date)	Sprint Release Date (Actual)
Sprint 1	20	6 days	24 Oct 2022	29 Oct 2022	20	29 Oct 2022
Sprint 2	20	6 days	31 Oct 2022	05 Nov 2022	20	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 202	19 Nov 2022	20	19 Nov 2022

6.2 Sprint Delivery Schedule

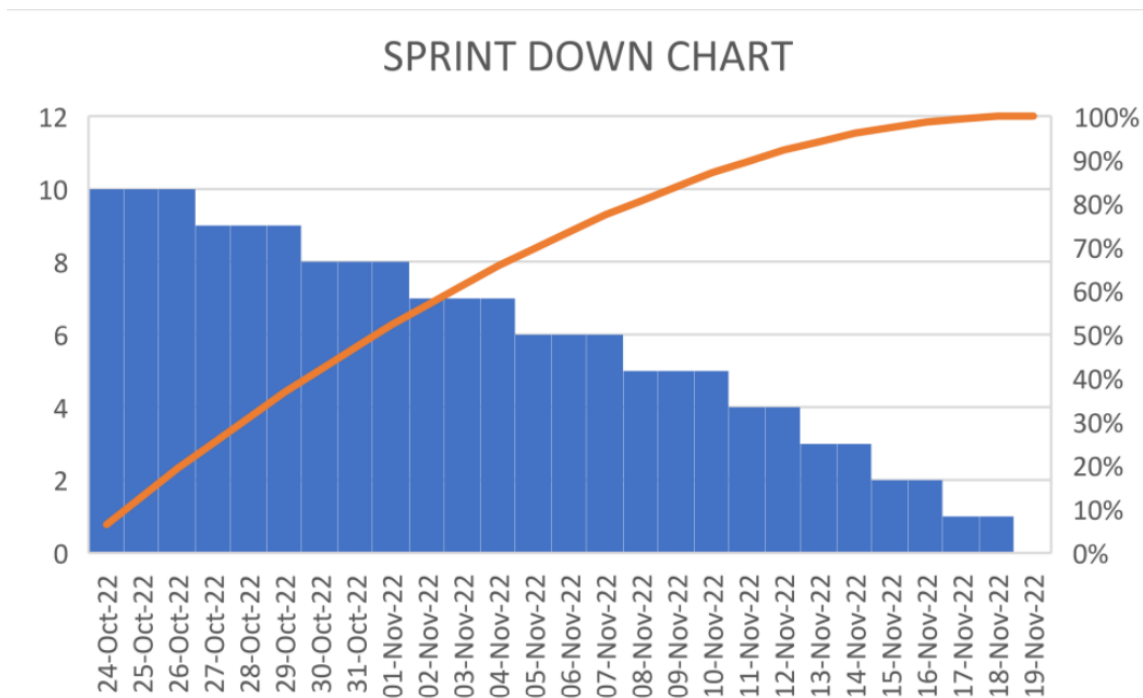
Sprint	Duration	Sprint Start Date	Sprint End Date (Planned)	Sprint Release Date (Actual)
Sprint 1	6 days	24 Oct 2022	29 Oct 2022	29 Oct 2022
Sprint 2	6 days	31 Oct 2022	05 Nov 2022	05 Nov 2022
Sprint 3	6 days	07 Nov 2022	12 Nov 2022	12 Nov 2022
Sprint 4	6 days	14 Nov 202	19 Nov 2022	19 Nov 2022

6.3 Reports from JIRA

Velocity: We have a 10-day sprint duration, and the velocity of the team is 20 (points per sprint). Let's calculate the team's average velocity (AV) per iteration unit (story points per day) $AV = \text{Sprint duration} / \text{Velocity} = 20 / 6 = 3$ Burn down Chart: A burn down chart is a graphical representation of work left to do versus time. It is often used in agile software development methodologies such as Scrum. However, burn down charts can be applied to any project containing measurable progress over time.

	OCT							NOV							NOV							NOV						
	24	25	26	27	28	29	30	31	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	
Sprints	IVDAACEF Sprint 1							IVDAACEF Sprint 2							IVDAACEF Sprint 3							IVDAACEF Sprint 4						
> IVDAACEFIC-10 Registration																												
> IVDAACEFIC-11 Login																												
> IVDAACEFIC-12 Dashboard																												
> IVDAACEFIC-13 Details about Insurance companies																												
> IVDAACEFIC-14 repeated logins and logout																												
> IVDAACEFIC-15 Webpage																												
> IVDAACEFIC-16 Details about estimated cost based...																												
> IVDAACEFIC-17 Provide friendly and efficient custo...																												
> IVDAACEFIC-18 overview the entire process and act...																												

Sprint Down Chart:



7 CODING & SOLUTIONING (Explain the features added in the project along with code)

7.1 Feature 1

```
8 app = Flask(__name__)
9 run_with_ngrok(app)
10
11 @app.route('/register', methods = ['POST','GET'])
12 def signUp():
13     username = request.args.get('username')
14     carName = request.args.get('carName')
15     regNo = request.args.get('regNo')
16     regDate = request.args.get('regDate')
17     password = request.args.get('password')
18
19     sql_stmt = "INSERT INTO users VALUES(?,?,?,?,?,?,?)"
20     stmt = ibm_db.prepare(conn, sql_stmt)
21     uid = uuid.uuid4().hex[:8]
22     ibm_db.bind_param(stmt, 1, uid)
23     ibm_db.bind_param(stmt, 2, username)
24     ibm_db.bind_param(stmt, 3, password)
25     ibm_db.bind_param(stmt, 4, carName)
26     ibm_db.bind_param(stmt, 5, regNo)
27     ibm_db.bind_param(stmt, 6, regDate)
28     ibm_db.bind_param(stmt, 7, "")
29
30     try:
31         resp = ibm_db.execute(stmt)
32         if resp:
33             try:
34                 resp = jsonify({"success": uid})
35                 resp.headers.add('Access-Control-Allow-
Origin', '*')
36                 return resp
37             except:
38                 resp = jsonify({"success": False})
39                 resp.headers.add('Access-Control-Allow-
Origin', '*')
40                 return resp
41
42     except:
43         print(ibm_db.stmt_error)
44
```

```

45 @app.route('/login', methods = ['POST','GET'])
46 def signIn():
47
48     username = request.args.get('username')
49     password = request.args.get('password')
50
51     sql_stmt = f"SELECT * FROM users WHERE Username='{username}' A
ND Password='{password}'"
52     stmt = ibm_db.prepare(conn, sql_stmt)
53
54     print(stmt)
55
56     try:
57         resp = ibm_db.execute(stmt)
58         row = ibm_db.fetch_both(stmt)
59         if row == False:
60             resp = jsonify({"uid":False})
61             resp.headers.add('Access-Control-Allow-Origin', '*')
62             return resp
63         else:
64             listData = {
65                 "uid": row['UID'],
66                 "username": row['USERNAME'],
67                 "carname": row['CARNAME'],
68                 "regno": row['REGNO'],
69                 "regdate": row['REGDATE'],
70                 "claim": row['CLAIMAMOUNT']
71             }
72             resp = jsonify(listData)
73             resp.headers.add('Access-Control-Allow-Origin', '*')
74             return resp
75
76     except:
77         print(ibm_db.stmt_error)
78
79 @app.route('/postImage', methods = ['POST'])
80 def setImg():
81
82     index = 1
83     # print(request.form)
84     data = request.form
85     vals = list(data.values())
86     vals.pop(0)
87

```

```

88     analyzed = []
89     insertValues = []
90     claimAmount = 0
91
92
93     if not os.path.exists(ROOT_DIR + '/server/public/' + data['uid
94         ']):
95         os.makedirs(ROOT_DIR + '/server/public/' + data['uid'])
96
97     for x in vals:
98         fileName = data['uid']+'_'+str(index)
99
100         with open(ROOT_DIR + '/server/public/' + data['uid'] + '/'
101             + fileName + ".jpg", "wb") as f:
102             f.write(base64.decodebytes(str.encode(x)))
103
104         index += 1
105
106     onlyfiles = [f for f in listdir(ROOT_DIR + '/server/public/'+d
107         ata['uid']+'/') if isfile(join(ROOT_DIR + '/server/public/'+data['
108         uid']+'/', f))]
109
110     print(onlyfiles)
111
112     # prediction(data['uid'],onlyfiles[0])
113     # prediction('das231564a','das231564a_1.jpg')
114
115     # time.sleep(20)
116
117     for x in onlyfiles:
118         listData = {
119             "filename": str(x),
120             "area": "",
121             "amount": 0
122         }
123         # tmpData = {
124         #     "filename": str(x),
125         #     "area": "",
126         #     "amount": 0
127         # }
128         result = prediction(str(data['uid']),str(x))
129         print(x)
130         listData['area'] = (result / 1000)

```

```

128     listData['amount'] = round((listData['area'] * 200),2)
129     claimAmount += listData['amount']
130     analyzed.append(listData)
131     insertValues.append((data['uid'],str(listData['area']),str(l
istData['amount'])))
132     print(analyzed)
133     print(insertValues)
134
135     tuple_of_tuples = tuple([tuple(x) for x in insertValues])
136
137     sql_stmt = f"INSERT INTO images VALUES(?,?,?)"
138     stmt = ibm_db.prepare(conn, sql_stmt)
139
140     try:
141         exe = ibm_db.execute_many(stmt,tuple_of_tuples)
142         print(exe)
143         sql_stmt_2 = f"update users set claimamount='{str(claimAmo
unt)}' where uid='{data['uid']}'"
144         res = ibm_db.exec_immediate(conn, sql_stmt_2)
145         resp = jsonify(analyzed)
146         resp.headers.add('Access-Control-Allow-Origin', '*')
147         return resp
148     except:
149         resp = jsonify({"uid": 'error'})
150         resp.headers.add('Access-Control-Allow-Origin', '*')
151         return resp
152         print(ibm_db.stmt_error)
153
154
155 @app.route('/viewImage', methods = ['POST','GET'])
156 def viewImg():
157     uid = request.args['uid']
158     filename = request.args['filename']
159     # uid = request.args['uid']
160     # print(list(uid))
161     return send_from_directory(ROOT_DIR + '/server/public/'+uid+'/'
, filename)
162
163 @app.route('/getImage', methods = ['POST'])
164 def getImg():
165     uid = request.args['uid']
166     imageData = []
167     fileCount = 0

```

```

168     onlyfiles = [f for f in listdir(ROOT_DIR + '/server/public/'+u
    id+'/') if isfile(join(ROOT_DIR + '/server/public/'+uid+'/'+', f))]
169     print(onlyfiles)
170     sql_stmt = f"SELECT * FROM images WHERE uid='{uid}'"
171     stmt = ibm_db.prepare(conn, sql_stmt)
172
173     print(stmt)
174
175     try:
176         exe = ibm_db.execute(stmt)
177         print(exe)
178         row = ibm_db.fetch_both(stmt)
179         while(row):
180             listData = {
181                 "filename": str(onlyfiles[fileCount]),
182                 "area": row[1],
183                 "amount": row[2]
184             }
185             imageData.append(listData)
186             fileCount += 1
187             # print(row[0])
188             # print()
189             # print()
190             row = ibm_db.fetch_both(stmt)
191         print(imageData)
192         resp = jsonify(imageData)
193         resp.headers.add('Access-Control-Allow-Origin', '*')
194         return resp
195
196     except:
197         print(ibm_db.stmt_error)
198
199 app.run()

```

9.2 Feature 2

```

10 def prediction(uid,filename):
11     # print(uid)
12     area = 0
13     # path_to_new_image = '/content/drive/MyDrive/pythoncode/insuran
    ceApp/dataset/test/test3.jpg'
14     path_to_new_image = os.path.join(ROOT_DIR + '/server/public/' +
    uid + '/' + filename)
15     print(path_to_new_image)

```



```

16  image1 = mpimg.imread(path_to_new_image)
17  print(np.shape(image1.shape))
18  if np.shape(image1.shape)[0] < 3:
19      image1 = image1.reshape((*image1.shape, 1))
20
21  # Run object detection
22  #print(len([image1]))
23  results1 = model.detect([image1], verbose=0)
24
25  # Display results
26  # ax = get_ax(1)
27  r1 = results1[0]
28
29  def save_co_ordinates(image, boxes, masks, class_ids, class_name
s):
30      image = image.split("/")[-1]
31      image_data = []
32
33      for i in range(boxes.shape[0]):
34
35          mask = masks[:, :, i]
36          padded_mask = np.zeros(
37              (mask.shape[0] + 2, mask.shape[1] + 2), dtype=np.uint8)
38          padded_mask[1:-1, 1:-1] = mask
39          contours = find_contours(padded_mask, 0.5)
40          for verts in contours:
41              verts = np.fliplr(verts) - 1
42              list_co_ordinates = np.moveaxis(verts, 1, 0).tolist(
43              )
44              region = {"shape_attributes": {"all_points_x": list_
co_ordinates[0],
45                                          "all_points_y": list_c
o_ordinates[1]},
46                      }
47              image_data.append(region)
48          data = {"filename": image, "regions": image_data, "count": b
oxes.shape[0]}
49          return data
50
51  data = save_co_ordinates(path_to_new_image, r1['rois'], r1['mask
s'], r1['class_ids'], ['scratch', 'dent'])
52

```

```

53  for i in range(data['count']):
54      x = data['regions'][i]['shape_attributes']['all_points_x']
55      y = data['regions'][i]['shape_attributes']['all_points_y']
56      pgon = Polygon(zip(x, y))
57      area = pgon.area + area
58      print(pgon.area)
59
60  print(data['count'])
61
62  #####
63
64  visualize.save_image(image1, path_to_new_image, r1['rois'], r1['
  masks'],
65      r1['class_ids'], r1['scores'], dataset.class_names,
66      filter_class_names=['scratch', 'dent'], scores_thresh=0.9, mode
  =0)
67
68  #####
  #####
69
70  return round(area, 2)
71 prediction('2cbd9bd1', '2cbd9bd1_1.jpg')
72

```

8.TESTING

8.1 Test Cases

A test case has components that describe input, action and an expected response, in order to determine if a feature of an application is working correctly. A test case is a set of instructions on “HOW” to validate a particular test objective/target, which when followed will tell us if the expected behavior of the system is satisfied or not.

Characteristics of a good test case:

- Accurate: Exacts the purpose.
- Economical: No unnecessary steps or words.
- Traceable: Capable of being traced to requirements.
- Repeatable: Can be used to perform the test over and over.
- Reusable: Can be reused if necessary.

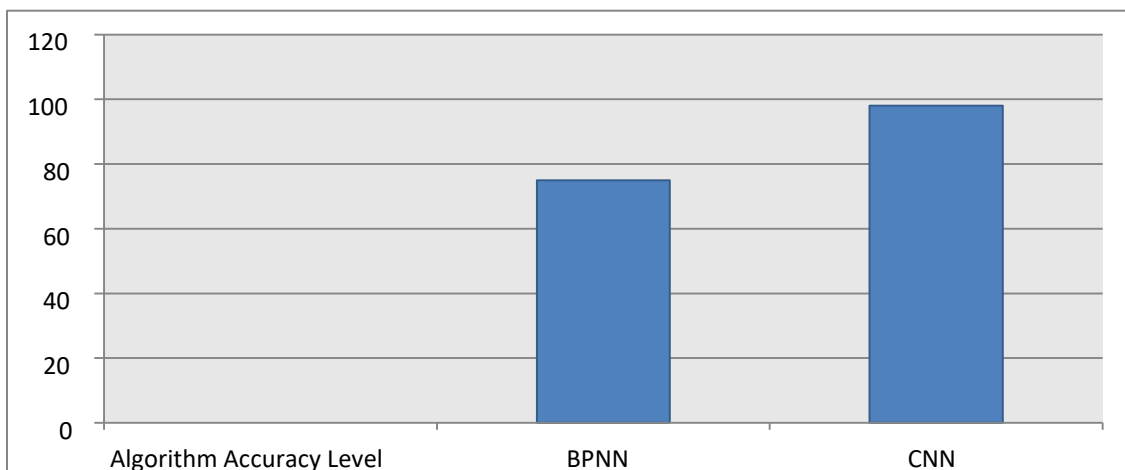
S.NO	Scenario	Input	Excepted output	Actual output
1	User login	User name and password	Login	Login success.
2	Upload Image	Upload damaged vehicle image as a input	Detecting object and analyze for claim insurance	Details are stored in a database.

8.2 User Acceptance Testing

This sort of testing is carried out by users, clients, or other authorised bodies to identify the requirements and operational procedures of an application or piece of software. The most crucial stage of testing is acceptance testing since it determines whether or not the customer will accept the application or programme. It could entail the application's U.I., performance, usability, and usefulness. It is also referred to as end-user testing, operational acceptance testing, and user acceptance testing (UAT).

9. RESULTS

9.1 Performance Metrics



10. ADVANTAGES & DISADVANTAGES

ADVANTAGE :

- Digitalized claim process makes easy to use
- Give the accurate result of the damaged vehicle
- Helps the insurance company to analyze the damaged vehicle and also payment process.

DISADVANTAGE

- It will take more time to claim the insurance in manual process
- Because of incorrect claims, the company behaves badly and doesn't make payments currently.
- Poor customer support

11. CONCLUSION

In this research proposal, a neural network-based solution for automobile detection will be used to address the issues of automotive damage analysis and position and severity prediction. This project does several tasks in one bundle. The method will unquestionably assist the insurance firms in conducting far more thorough and systematic analyses of the vehicle damage. Simply sending the system a photograph of the vehicle, it will evaluate it and determine whether there is damage of any type, where it is located, and how severe it is.

12.FUTURE SCOPE

In future work, need to use several regularisation methods with a big dataset in our next work. Anticipate the cost of a car damaged component more accurately and reliably if we have higher quality datasets that include the attributes of a car (make, model, and year of production), location data, kind of damaged part, and repair cost. This study makes it possible to work together on picture recognition projects in the future, with a focus on the auto insurance industry. The study was able to accurately validate the presence of damage, its location, and its degree while eliminating human bias. These can be further enhanced by adding the on the fly data augmentation approaches.

13.APPENDIX

Source Code

Index:

```
<template>
  <div class="flex h-screen bg-screen">
    <div class="m-auto">
      <div class="w-96 h-3/4 bg-white rounded-lg px-8 py-8">
        <div v-if="page == 'login'">
          <LoginPage />
        </div>
        <div v-if="page == 'reg'">
          <RegPage />
        </div>
      </div>
    </div>
  </div>
  <div v-if="isLoading">
```

```
    <div class="flex justify-center items-center h-screen fixed top-0 left-
0 right-0 bottom-0 w-full z-50 overflow-hidden bg-gray-700 opacity-75">
      <LoginPage />
    </div>
  </div>
</div>
</template>
```

```
<script>
```

```
import LoginPage from './login.vue'
import RegPage from './register.vue'
import LoadingPage from './loading.vue'
import ProfilePage from './profile.vue'
```

```
export default {
  data: () => ({
    isWarning: false,
    isLoading: false,
    page: 'login',
    posts: [],
    isValid: false,
    uid: "",
    username: "",
    password: "",
    carName: "",
```

```
    regNo: "",
    regDate: "",
    url: 'http://127.0.0.1:5000/login?'
  )),
  name: 'IndexPage',
  components: {
    LoginPage,
    RegPage,
    LoadingPage,
    ProfilePage,
  },
  mounted() {

  },
  methods: {
    changePage(page) {
      return this.page = page
    },
    setLoading(action) {
      return this.isLoading = action
    },

    changeAfterReg(uid,username,carname,regno,regdate) {
      this.uid = uid
      this.username = username
      this.carName = carname
```

```

    this.regNo = regno
    this.regDate = regdate
    var tmpdata = {
      'uid': uid,
      'username': username,
      'carname': carname,
      'regno': regno,
      'regdate': regdate
    }
  },
  created() {
    this.$root.$refs.index = this;
  }
}
</script>

```

Login:

```

<template lang="">
  <div>
    <h1 class="pb-3 font-Montserrat-500">Username</h1>
    <input v-model="username" class="font-Montserrat-400 text-sm
shadow appearance-none border rounded w-full py-2 px-3 text-gray-700

```



```
leading-tight focus:outline-none focus:shadow-outline" id="username"
type="text" placeholder="username">
```

```
<h1 id="errorMsg" class="hidden pt-1 font-Montserrat-400 text-sm
text-red-600">Username or Password is wrong. please try again.</h1>
```

```
<h1 class="pt-4 pb-3 font-Montserrat-500">Password</h1>
```

```
<input v-model="password" class="shadow appearance-none border
rounded w-full py-2 px-3 text-gray-700 mb-3 leading-tight focus:outline-
none focus:shadow-outline" id="password" type="password"
placeholder="password">
```

```
<div class="pt-4 flex items-center justify-around">
```

```
<div v-if="isValid">
```

```
<button @click="login(username,password)" class="bg-blue-500
hover:bg-blue-700 text-white font-bold py-2 px-4 rounded focus:outline-
none focus:shadow-outline" type="button">
```

```
    Sign In
```

```
</button>
```

```
</div>
```

```
<div v-else>
```

```
<div class="cursor-default bg-gray-500 text-gray-200 font-bold
py-2 px-4 rounded focus:outline-none focus:shadow-outline"
type="button">
```

```
    Sign In
```

```
</div>
```

```
</div>
```

```
<div @click="regpage()" class="cursor-pointer text-blue-400
hover:text-blue-600 font-semibold">
```

```
        Register user
      </div>
    </div>
  </div>
</template>
<script>
```

```
export default {
  name: 'LoginPage',
  data: () => ({
    posts: [],
    imgData: [],
    isValid: false,
    username: '',
    password: '',
    url: 'http://d9a5-35-196-119-233.ngrok.io/login'
  }),
  mounted() {

    document.getElementById('username').addEventListener('input', (e) =>
    {
      if(this.username.length > 0 && this.password.length > 0) {
        this.isValid = true
      } else {
        this.isValid = false
      }
    })
  }
}
```

```

        document.getElementById('errmsg').classList.add('hidden')
        console.log(this.isValid)
    })

    document.getElementById('password').addEventListener('input', (e) =>
    {
        if(this.username.length > 0 && this.password.length > 0) {
            this.isValid = true
        } else {
            this.isValid = false
        }
        document.getElementById('errmsg').classList.add('hidden')
        console.log(this.isValid)
    })

    },
    methods: {
        regpage() {
            this.$root.$refs.index.changePage('reg');
        },
        async login(username,password) {
            this.$root.$refs.index.setLoading(true);
            const res = await fetch(this.$server.url + 'login?' + new
URLSearchParams({username: username.toLowerCase(), password:
password}},{method: "POST"}))
            this.posts = await res.json()

```

```
console.log(this.posts)
if(this.posts['uid'] != false) {
  console.log('success')
  var resp = {
    'uid': this.posts['uid'],
    'username': this.posts['username'].toUpperCase(),
    'carname': this.posts['carname'].toUpperCase(),
    'regno': this.posts['regno'].toUpperCase(),
    'regdate': this.posts['regdate'],
    'claim': this.posts['claim']
  }
  this.$root.$refs.index.setLoading(false);
  this.$router.push({ name: 'profile', params: { data: resp } })
} else {
  this.$root.$refs.index.setLoading(false);
  document.getElementById('errmsg').classList.remove('hidden')
  console.log('failed')
}
},
},
}
</script>
<style lang="">

</style>
```

Register:

```
<template lang="">
  <div>
    <h1 class="pb-3 font-Montserrat-500">Username</h1>
    <input v-model="username" class="font-Montserrat-400 text-sm
shadow appearance-none border rounded w-full py-2 px-3 text-gray-700
leading-tight focus:outline-none focus:shadow-outline" id="username"
type="text" placeholder="Username">
    <h1 class="pt-4 pb-3 font-Montserrat-500">Car Name</h1>
    <input v-model="carName" class="font-Montserrat-400 text-sm
shadow appearance-none border rounded w-full py-2 px-3 text-gray-700
leading-tight focus:outline-none focus:shadow-outline" id="carname"
type="text" placeholder="Car Name">
    <h1 class="pt-4 pb-3 font-Montserrat-500">Registration
Number</h1>
    <input v-model="regNo" class="font-Montserrat-400 text-sm
shadow appearance-none border rounded w-full py-2 px-3 text-gray-700
leading-tight focus:outline-none focus:shadow-outline" id="regno"
type="text" placeholder="Registration Number">
    <h1 class="pt-4 pb-3 font-Montserrat-500">Registration Date</h1>
    <div class="shadow appearance-none border rounded w-full py-2
px-3">
      <datepicker v-model="tmpDate"
@input="handleChange(tmpDate)" :placeholder="datePh"
format="dd/MM/yyyy" />
    </div>
```

```
<h1 class="pt-4 pb-3 font-Montserrat-500">Password</h1>
```

```
<input v-model="password" class="shadow appearance-none border
rounded w-full py-2 px-3 text-gray-700 mb-3 leading-tight focus:outline-
none focus:shadow-outline" id="password" type="password"
placeholder="Password">
```

```
<div class="pt-4 flex items-center justify-around">
```

```
<div v-if="isValid">
```

```
<button
```

```
@click="register(username,carName,regNo,regDate,password)"
```

```
class="bg-blue-500 hover:bg-blue-700 text-white font-bold py-2 px-4
rounded focus:outline-none focus:shadow-outline" type="button">
```

```
Sign Up
```

```
</button>
```

```
</div>
```

```
<div v-else>
```

```
<div class="cursor-default bg-gray-500 text-gray-200 font-bold
py-2 px-4 rounded focus:outline-none focus:shadow-outline"
type="button">
```

```
Sign Up
```

```
</div>
```

```
</div>
```

```
<div @click="loginpage()" class="cursor-default bg-red-600
hover:bg-red-800 text-white font-bold py-2 px-4 rounded focus:outline-
none focus:shadow-outline" type="button">
```

```
Cancel
```

```
</div>
```

</div>

</div>

</template>

<script>

import moment from 'moment'

export default {

name: 'RegPage',

components: {

},

data: () => ({

posts: [],

tmpDate: "",

datePh: "Select Date",

isValid: false,

username: "",

password: "",

carName: "",

regNo: "",

regDate: "",

url: 'http://127.0.0.1:5000/register?'

}),

mounted() {

```

        document.getElementById('username').addEventListener('input', (e)
=> {
            if(this.username.length > 0 && this.carName.length > 0 &&
this.regNo.length > 0 && this.regDate.length > 0 &&
this.password.length > 0) {
                this.isValid = true
            } else {
                this.isValid = false
            }
            console.log(this.regDate)
        })
        document.getElementById('carname').addEventListener('input', (e)
=> {
            if(this.username.length > 0 && this.carName.length > 0 &&
this.regNo.length > 0 && this.regDate.length > 0 &&
this.password.length > 0) {
                this.isValid = true
            } else {
                this.isValid = false
            }
            console.log(this.regDate)
        })
        document.getElementById('regno').addEventListener('input', (e) =>
{

```



```

        if(this.username.length > 0 && this.carName.length > 0 &&
this.regNo.length > 0 && this.regDate.length > 0 &&
this.password.length > 0) {
            this.isValid = true
        } else {
            this.isValid = false
        }
        console.log(this.regDate)
    })
    document.getElementById('password').addEventListener('input', (e)
=> {
        if(this.username.length > 0 && this.carName.length > 0 &&
this.regNo.length > 0 && this.regDate.length > 0 &&
this.password.length > 0) {
            this.isValid = true
        } else {
            this.isValid = false
        }
        console.log(this.regDate)
    })
},
methods: {
    loginpage() {
        this.$root.$refs.index.changePage('login');
    },
    handleChange(e) {

```

```

    var date = moment(e).format('MM/DD/YYYY')
    this.regDate = date
    console.log(this.regDate)
    if(this.username.length > 0 && this.carName.length > 0 &&
this.regNo.length > 0 && this.regDate.length > 0 &&
this.password.length > 0) {
        this.isValid = true
    } else {
        this.isValid = false
    }
},
async register(username,carName,regNo,regDate,password) {
    this.$root.$refs.index.setLoading(true);
    const res = await fetch(this.$server.url + 'register?' + new
URLSearchParams({username: username.toLowerCase(), carName:
carName, regNo: regNo, regDate: regDate, password:
password})),{method: "POST"})
    if(res.status == 200) {
        this.posts = await res.json()
        if(this.posts['success'] != false) {
            console.log('success')
            var resp = {
                'uid': this.posts['success'],
                'username': this.username.toUpperCase(),
                'carname': this.carName.toUpperCase(),
                'regno': this.regNo.toUpperCase(),

```

```
        'regdate': this.regDate,  
        'claim': "  
    }  
    this.$root.$refs.index.setLoading(false);  
    this.$router.push({ name: 'profile', params: { data: resp } })  
  } else {  
    console.log('failed')  
  }  
}  
},  
},  
}  
</script>  
<style lang="">  
  
</style>
```

GitHub & Project Demo Link

Github link: <https://github.com/IBM-EPBL/IBM-Project-39308-1660405304>

Project demo: <https://photos.app.goo.gl/KFDAuxboP5tA1xZ4A>