EV Lifeline

Revolutionizing Vehicle Assistance: AI-Powered Solutions for Breakdown Challenges

Prepared By
TMP-2023-24-117

03rd November, 2023

PROJECT INTRODUCTION

- Currently, the world demand for electric vehicles is increasing, but in countries like Sri Lanka, there is a lack of people turning to electric vehicles.
- The reason for this is the lack of knowledge about electric vehicles, the difficulty of finding mechanics when the vehicle breaks down and the difficulty of finding necessary spare parts.
- The purpose of our grant is to provide a successful solution to avoid the hassle and inconvenience that occurs when an electric vehicle breaks down on the road.

OUR TEAM

PERERA B.N.H.
IT20096434

SIRIMANNA D.J.T.K
IT20038328

MADHUBHASHANA A.G.K
IT20038182

VIJERATHNA A.G.V.K.M.
IT20644826

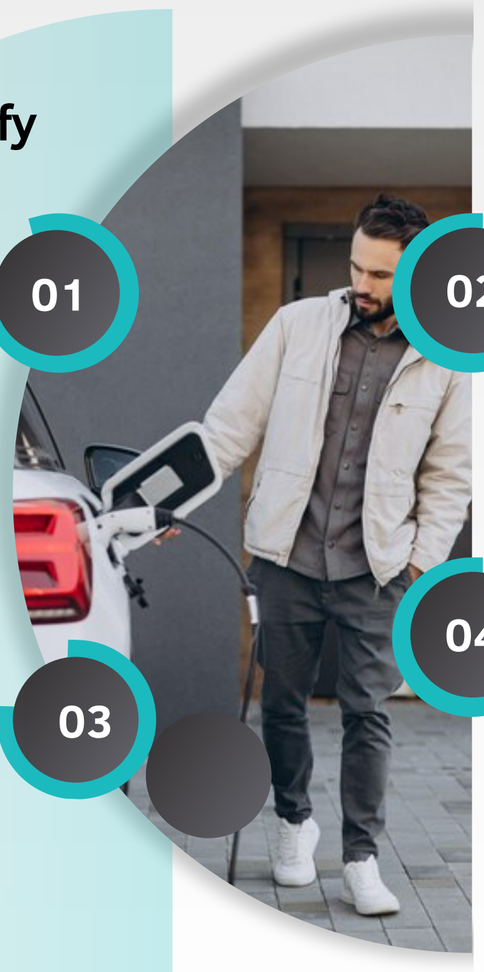
OVERALL PROJECT DESCRIPTION

- Individuals, especially those with limited vehicle knowledge, struggle to identify and address breakdown issues.
- Lack of easily accessible information, mechanics, spare part shops, and maintenance advice.
- Uncertainty about the appropriate actions when a breakdown occurs.
- Difficulty in finding reliable mechanics, particularly in unfamiliar areas.
- No effective mechanisms to inform vehicle owners about essential maintenance tasks, resulting in inconvenience and potential safety risks.

RESEARCH OBJECTIVES

01 Conversational AI chatbot to identify electric vehicle faults using text, dashboard warnings and provide solutions

02 Intelligent Geolocation-based EV Breakdown Assistance with AI based Repair Service Cost Estimation



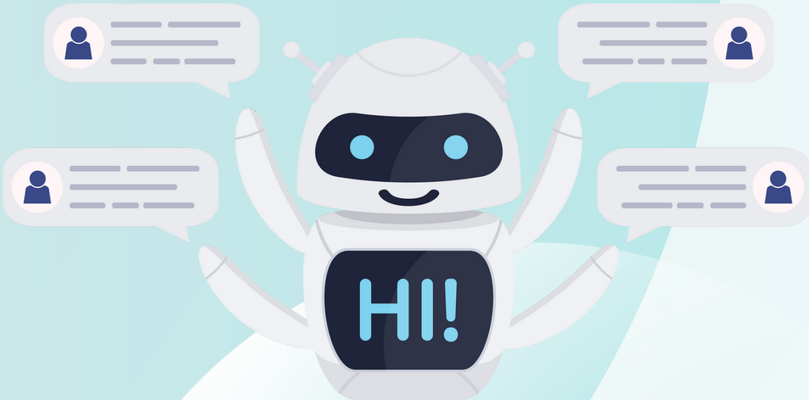
03 Automated Spare Parts Identification and Location System using Image Processing and Computer Vision Techniques

04 Enhancing User Experience in Vehicle Maintenance with a Smart Digital Monitoring Calendar and Reminder System

IT20096434

Perera B.N.H.

Bachelor of Science (Hons) in Information Technology Specializing in Information Technology



INTRODUCTION

Background

- EVs promise reduced emissions and reduced dependence on fossil fuels.
- However, unique challenges arise in managing breakdowns due to EVs' distinct technology.
- EV adoption is growing but still lags behind ICE vehicles.
- Proposed solution: AI-based chatbot for EV breakdown assistance.
- The chatbot would use AI and manufacturer data for real-time support.
- It offers accurate diagnostics, immediate assistance, and regular updates.
- Current breakdown assistance systems mainly focus on ICE vehicles, leaving a gap for specialized EV support.

Research Question

"In the event of an unexpected breakdown of an electric vehicle, How Can Conversational AI Chatbots Enhance Electric Vehicle Breakdown Support? "

This research addresses the need for efficient support during electric vehicle (EV) breakdowns on the road. By harnessing machine learning, AI, and natural language processing, the goal is to create a user-friendly chatbot that can analyze driver-provided text data and dashboard warning images to swiftly identify faults and offer recommendations, ultimately enhancing the assistance experience for EV drivers.

Research Gap

	Research A	Research B	Research C	Proposed System
Specified for Electric Vehicles (EV chatbot)	✗	✗	✗	✓
Availability of Personalized Solutions	✓	✓	✗	✓
Ability to Ask Follow Back Questions	✗	✗	✗	✓
Availability of a Knowledge Base	✓	✗	✓	✓
Update Knowledgebase Easily	✗	✗	✗	✓
Using ML and NLP for Real-Time Breakdown Identification and Assistance	✗	✗	✗	✓

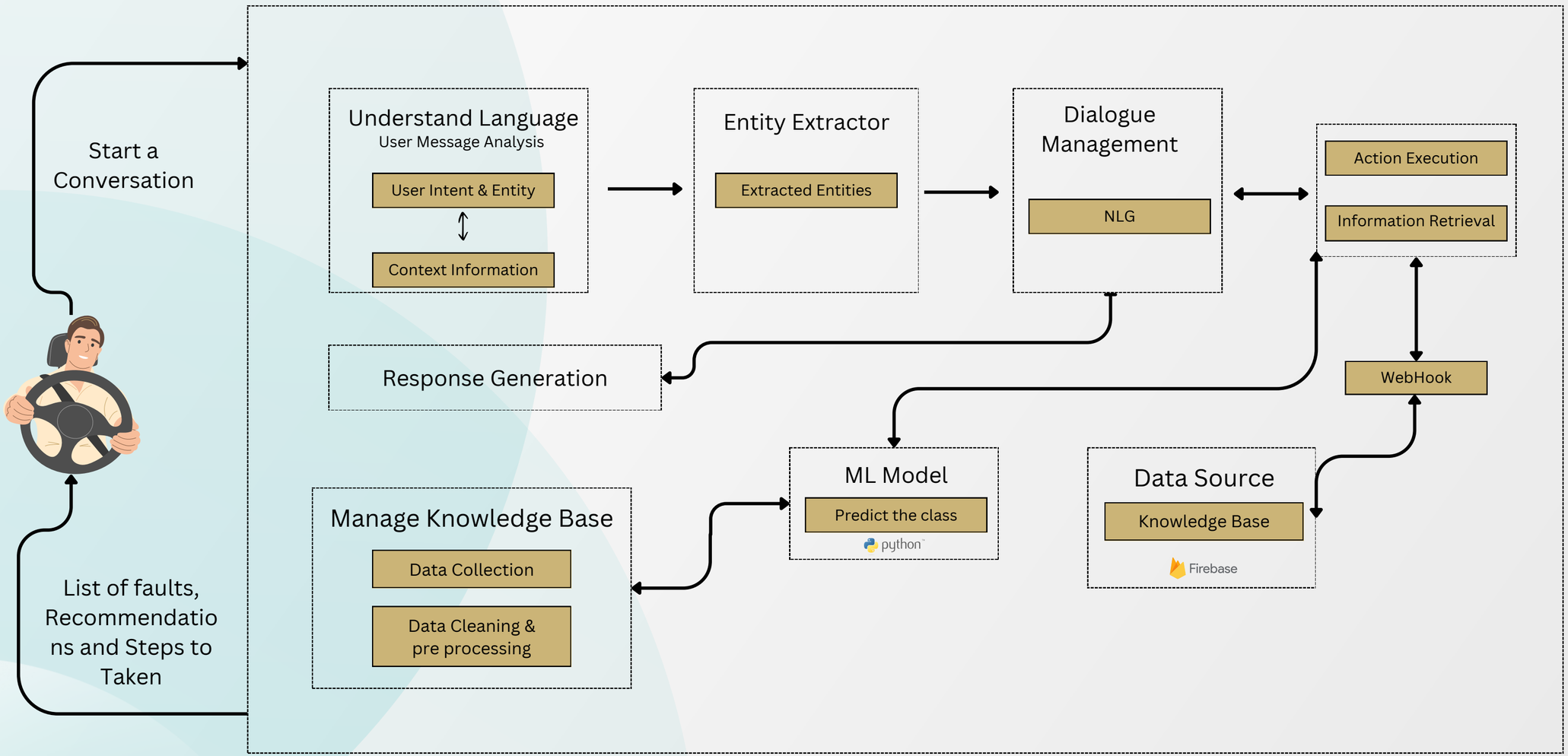
Specific Objective

IMPLEMENT A SMART AI CHATBOT TO RECEIVE TEXT AND DASHBOARD WARNING IMAGES, PROVIDING TIMELY SOLUTIONS FOR ELECTRIC VEHICLE BREAKDOWN SITUATIONS. THE KNOWLEDGE BASE WILL BE KEPT CURRENT AND ENHANCED TO ENSURE ACCURATE ASSISTANCE.

Sub Objectives

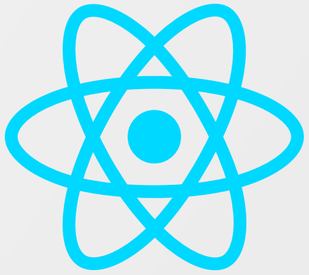
- Data Collection
- Data Preprocessing
- Data Cleaning
- AI Model Development
- Knowledge Base Creation
- Implement NLP techniques
- User Interface
- Testing and Validation

System Architecture Diagram



TECHNIQUES & TECHNOLOGIES

Technology	• React Native • Python • Flask • FireBase • Node Server
Techniques	• Machine Learning model • Recurrent Neural Network (RNN) • Intent Recognition • Entity Extraction • Dialogue Management
Algorithm	• Long Short-Term Memory



METHODOLOGY

DATA COLLECTION

DATA PRE-PROCESSING

CREATING ACTIONS BASED ON THE EV FAULTS

VALIDATE ANSWERS

CREATE EV FAULT-BASED KNOWLEDGE BASERE

DATA VISUALISATION

KNOWLEDGE BASE UPDATED

VERIFY RECOMENDATIONS WITH EXPERTS

COMPARISON

FRAMEWORK	EV BOT	MICROSOFT BOT FRAMEWORK	IBM WATSON ASSISTANCE	Dialogflow (formerly API.ai)
NLU	Highly customizable NLU component with support for complex intents and entities	Limited NLU capabilities with predefined intents and entities	Provides NLU capabilities with predefined intents and entities	Provides NLU capabilities with predefined intents and entities
DIALOGUE MANAGEMENT	Customizable dialogue management options	Rule-based dialogue management	Rule-based dialogue management options	Rule-based dialogue management options
CUSTOMIZABLITY	Highly customizable	Limited customizability due to closed-source nature	Limited Customizability	Limited Customizability
MACHINE LEARNING CAPABILITIES	Support integration of ML models	Limited ML capabilities	Provides machine learning capabilities	Provides ML capabilities.
OPEN-SOURCE	YES	NO	NO	NO

DATA COLLECTION AND PRE-PROCESSING

```
_=train_data_iterator.next()
print(f'Number of train batches: {len(train_data)}')
print(f'Number of training data: {len(train_data)*batch_size}')
print(f'Number of validation batches: {len(val_data)}')
print(f'Number of validation data: {len(val_data)*batch_size}')
print(f'Encoder Input shape (with batches): {_[0].shape}')
print(f'Decoder Input shape (with batches): {_[1].shape}')
print(f'Target Output shape (with batches): {_[2].shape}')
```

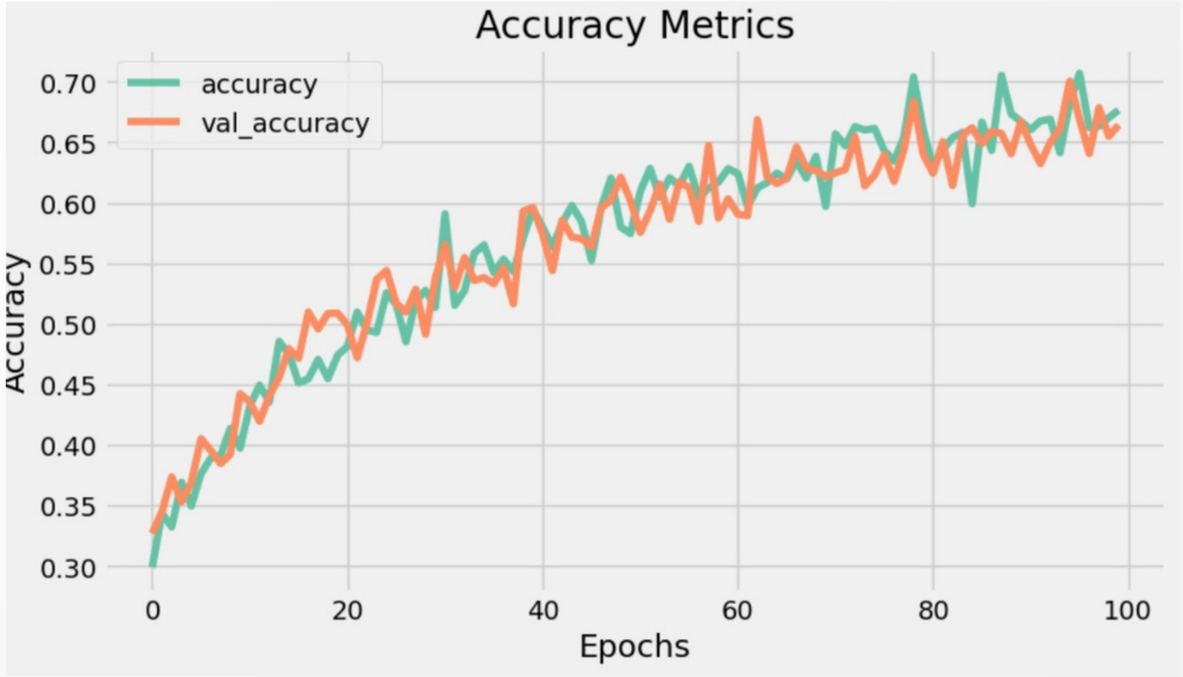
```
Number of train batches: 23
Number of training data: 3427
Number of validation batches: 3
Number of validation data: 447
Encoder Input shape (with batches): (149, 30)
Decoder Input shape (with batches): (149, 30)
```

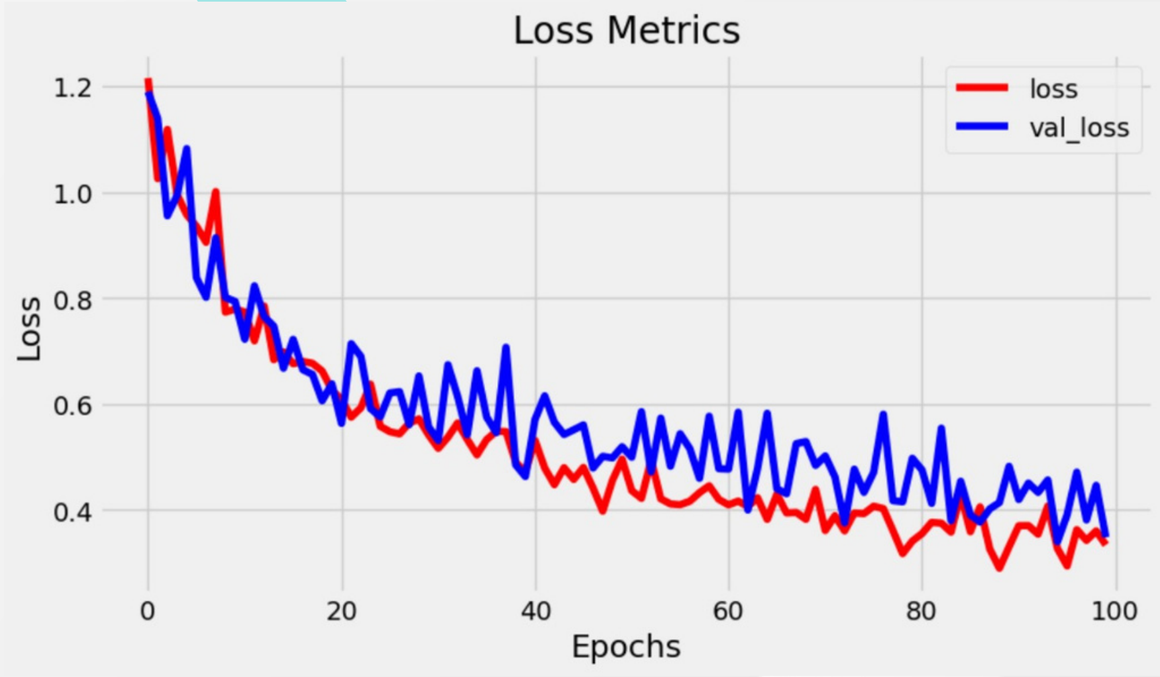
```
[ ]  history=model.fit(
        train_data,
        epochs=100,
        validation_data=val_data,
        callbacks=[
            tf.keras.callbacks.TensorBoard(log_dir='logs'),
            tf.keras.callbacks.ModelCheckpoint('ckpt',verbose=1,save_best_only=True)
        ]
    )
```

ACCURACY AND LOSS

```
Epoch 1/100
23/23 [==============================] - ETA: 0s - loss: 1.6494 - accuracy: 0.2188
Epoch 1: val_loss improved from inf to 1.19003, saving model to ckpt
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
WARNING:absl:Found untraced functions such as _update_step_xla, lstm_cell_layer_call_fn, lstm_cell_layer_call_and_return_conditional_losses, lstm_cell_1_layer_call_fn, lstm_cell_1_layer_call_and_return_cond
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
23/23 [==============================] - 100s 4s/step - loss: 1.6314 - accuracy: 0.2221 - val_loss: 1.1900 - val_accuracy: 0.3270
Epoch 2/100
23/23 [==============================] - ETA: 0s - loss: 1.2275 - accuracy: 0.3114
Epoch 2: val_loss improved from 1.19003 to 1.13934, saving model to ckpt
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
WARNING:absl:Found untraced functions such as _update_step_xla, lstm_cell_layer_call_fn, lstm_cell_layer_call_and_return_conditional_losses, lstm_cell_1_layer_call_fn, lstm_cell_1_layer_call_and_return_cond
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
WARNING:tensorflow:Model's `__init__()` arguments contain non-serializable objects. Please implement a `get_config()` method in the subclassed Model for proper saving and loading. Defaulting to empty config.
23/23 [==============================] - 79s 4s/step - loss: 1.2191 - accuracy: 0.3128 - val_loss: 1.1393 - val_accuracy: 0.3449
```

Accuracy and Loss Metrics





METHODOLOGY

FUNCTIONAL REQUIREMENTS

- Driver able to ask questions and chatbot provide feedback based on their EV faults.
- Able to receive and understand questions related to EV faults from driver.
- Able to identify the fault based on driver provided data and provide recommendations for the identified fault.
- Provide information about the steps to be taken.
- Able to maintain a record of EV fault and provide more adviced as necessary.

NON-FUNCTIONAL REQUIREMENTS

- Performance
- Security
- Reliability
- Scalability
- Usability
- Accessibility
- Compatibility
- Maintainability
- Compliance
- Performance metrics

SYSTEM REQUIREMENTS

- Server infrastructure
- Software development platform
- Natural language processing (NLP) tools
- Machine learning models
- Database management system
- Integration with third-party systems
- Communication channels
- Security measures
- Monitoring and analytics tools

COMPLETION AND FUTURE WORKS

COMPLETION OF COMPONENTS

- DATA GATHERING AND PREPROCESSING
- CREATE FAULT-BASED KNOWLEDGE BASE
- CREATING THE ACTIONS BASED ON THE FAULT
- PROVIDING THE RECOMENDATION BASED ON THE FAULT GIVEN BY THE DRIVER
- VERIFY THE PROVIDED ANSWERS USING ML M

FUTURE IMPLEMENTATIONS

- VERIFY SUGGESTIONS WITH EV EXPERTS
- IMPLEMENTATION OF USER INTERFACES
- HOSTING

IT20038328

Sirimanna D.J.T.K.

Bachelor of Science (Hons) in Information Technology Specializing in Information Technology

Background

- System for Electric Vehicle breakdown solutions using machine learning and geolocation.
- Connects vehicle owners with registered mechanics for emergency breakdown assistance.
- Skilled mechanics are matched to breakdown types and user locations from a database for efficient service.
- Allows users to schedule appointments for non-emergency breakdowns or inspections.
- Mechanic rating and feedback system to help users make informed decisions.

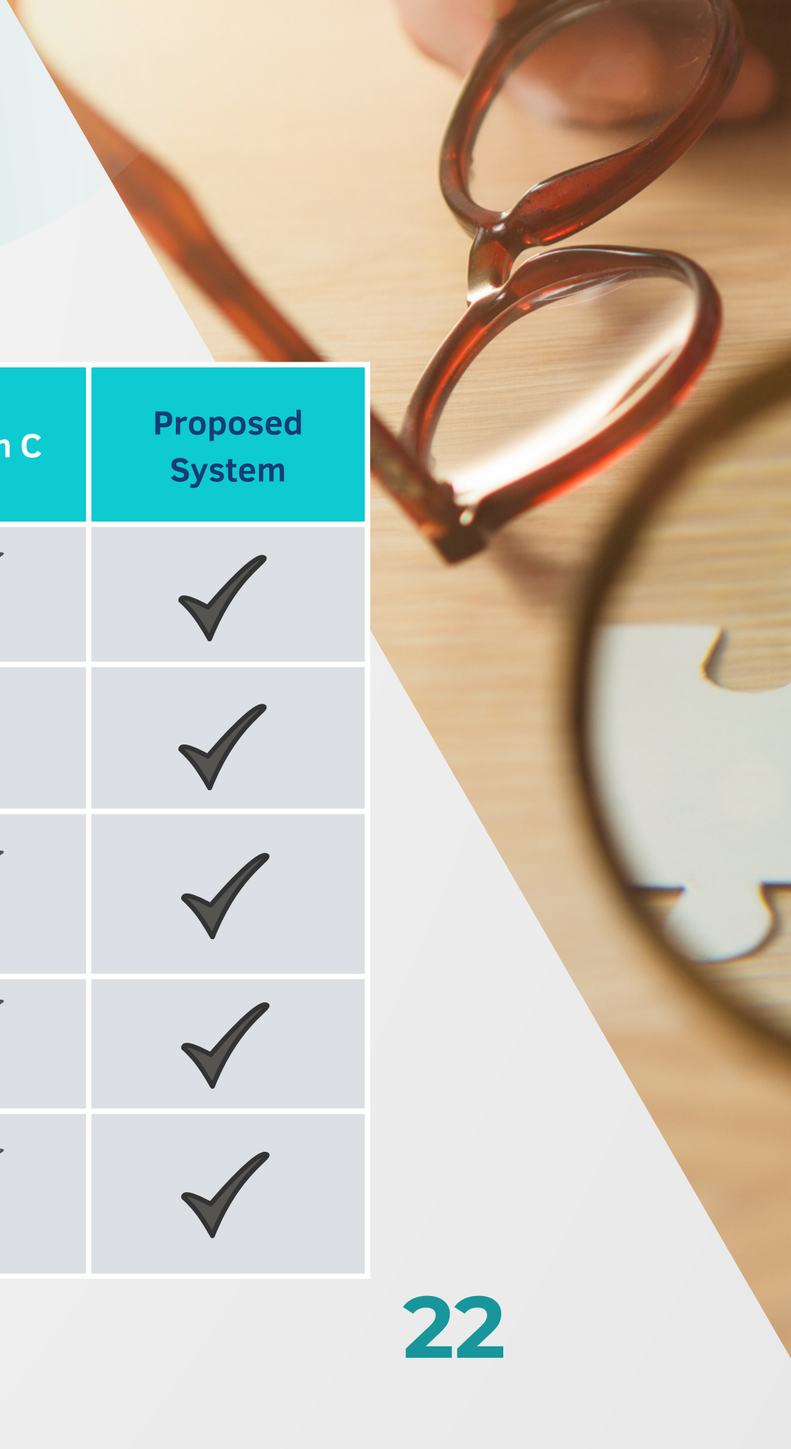
Research Question

"In the event of an unexpected breakdown of an electric vehicle in an unfamiliar location, how can the driver effectively diagnose the problem and locate a qualified mechanic for timely repairs?"

The research identifies a lack of efficient support for electric vehicle (EV) breakdowns in unfamiliar areas. By integrating machine learning, geolocation, and natural language processing, the aim is to develop a user-friendly system that can rapidly diagnose breakdowns, connect drivers with suitable mechanics, and improve the overall assistance experience for EV drivers.

Research Gap

	Research A	Research B	Research C	Proposed System
Specified for Electric Vehicles	✗	✗	✗	✓
Using Geolocation and GPS for locating	✓	✓	✓	✓
Using ML and NLP for Real-Time Breakdown Identification and Assistance	✗	✗	✗	✓
Proactive Repair Service scheduling	✓	✗	✗	✓
Emergency or Priority Service Request Broadcast	✗	✗	✗	✓



Specific Objective

Implement a Machine Learning model and algorithms that analyzes the breakdown description and suggests the most suitable mechanic based on specialization of mechanic and proximity to the driver’s location

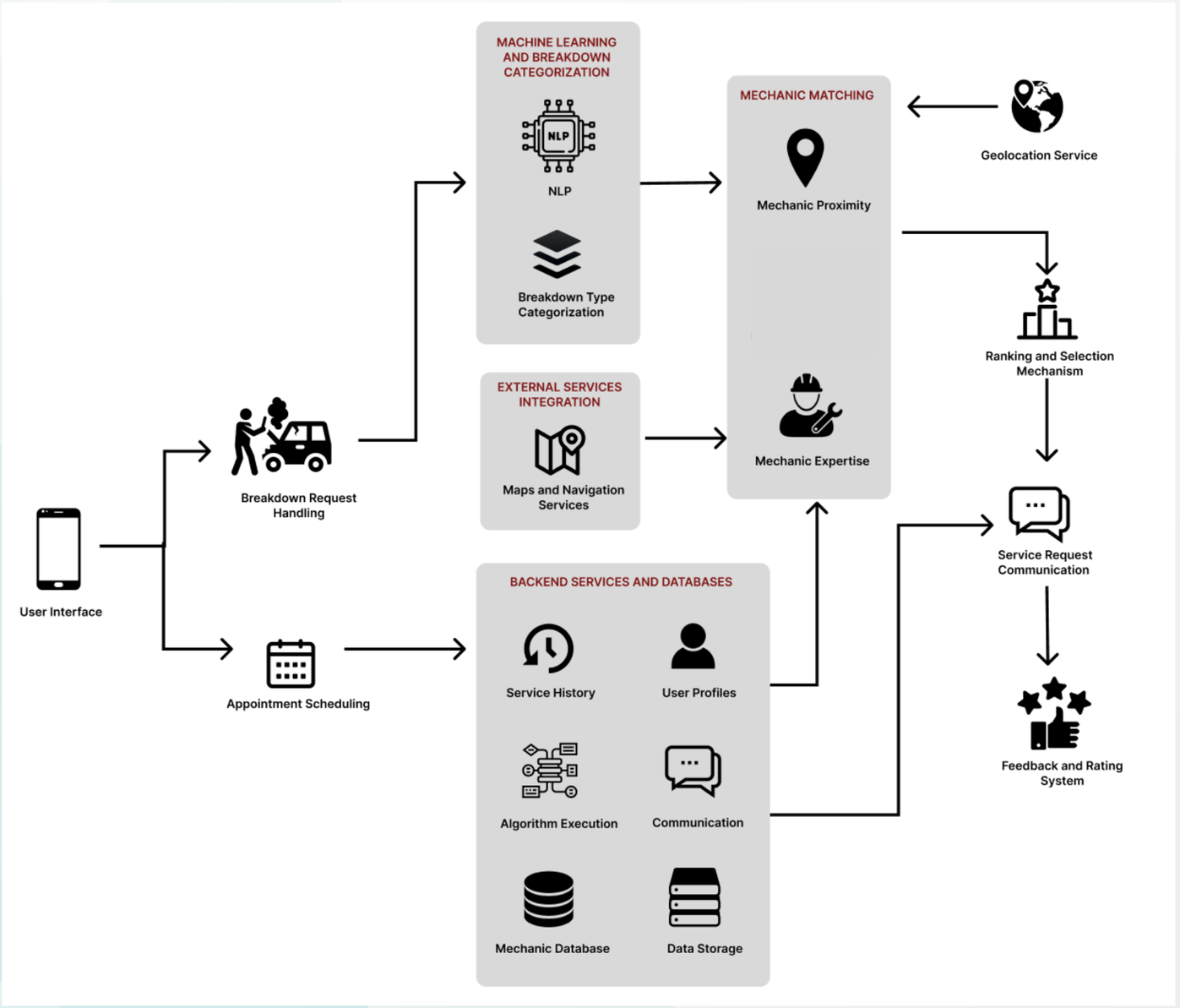
Sub Objectives

- Data Collection
- Data Preprocessing
- ML Model Development
- Implement NLP techniques
- Specialization Matching
- Geolocation Integration
- User Interface
- Testing and Validation
- Scalability and Performance:

Other Objectives

- Implement a private chat feature for Drivers and Mechanics
- Integrating a feedback and rating system for rate mechanics
- Implement a system for repair scheduling feature
- Test the final model on the testing data and implement it in a software application for practical use.

System Architecture Diagram



TECHNIQUES & TECHNOLOGIES

METHODOLOGY

Firebase

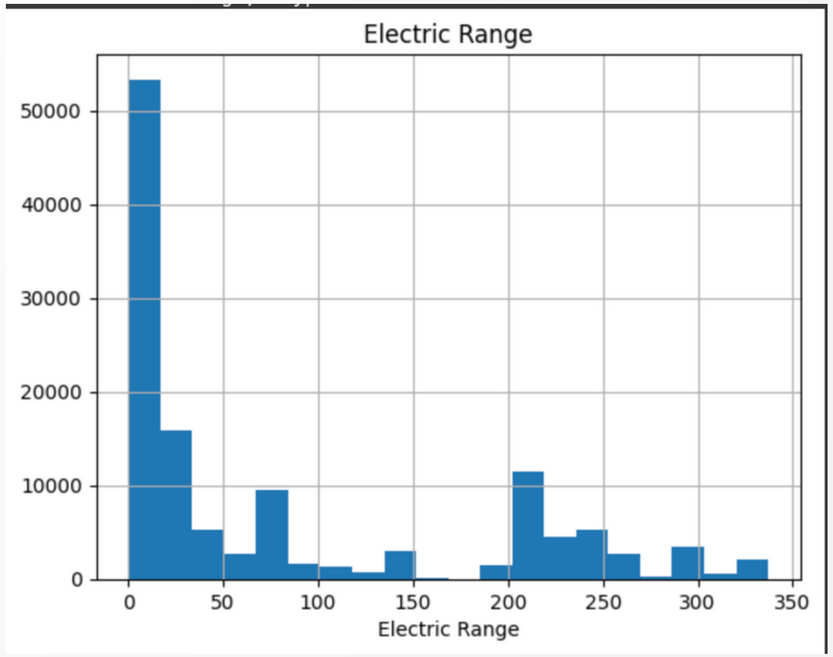
Technology	• React Native • Python • Firebase • Node Server • Geo Location • Flask
Techniques	• Auto Machine Learning (AML) • Sentiment analysis
Model	• Linear regression

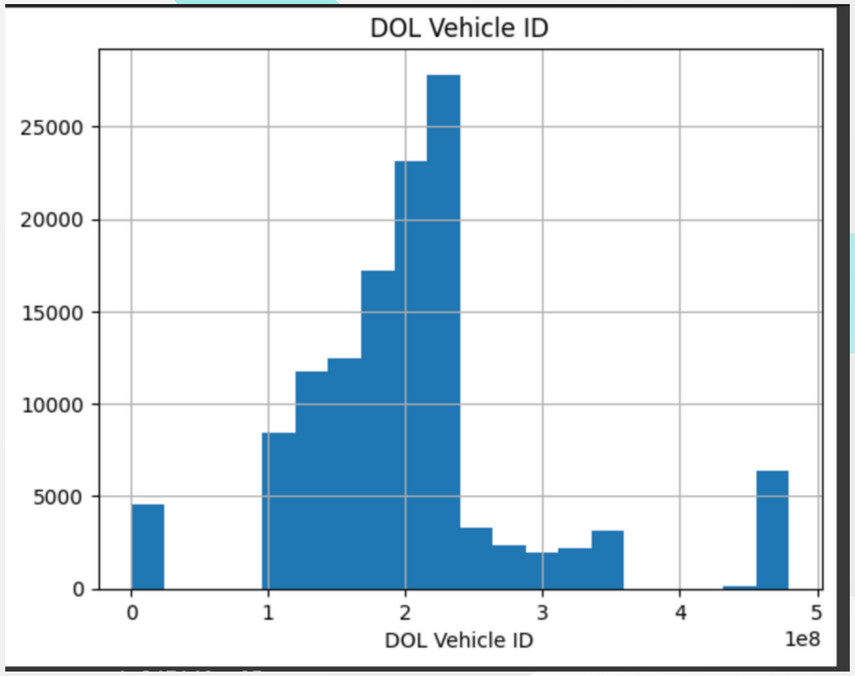
METHODOLOGY

- DATA COLLECTION
- DATA PRE-PROCESSING
- USE TRANSFER LEARNING BASED CNN ARCHITECTURES FOR TRAINING THE MODELS
- DATA AUGMENTATION AND SELECTED THE BEST TECHNIQUE TO RETRAIN MODELS TO ACHIEVE GOOD FITTING
- TUNING HYPERPARAMETERS AND RETRAINED MODELS TO SELECT THE BEST ARCHITECTURE
- DATA VISUALISATION
- HOST THE FINAL MODEL
- DISPLAYED THE RESULTS IN MOBILE APPLICATION

METHODOLOGY- EVIDENCE OF COMPLETION

Data Distribution Graphs





METHODOLOGY- EVIDENCE OF COMPLETION

Data Distribution ratio

```python
def missing_values_table(dataframe, na_name=False):
    na_columns = [col for col in dataframe.columns if dataframe[col].isnull().sum() > 0]

    n_miss = dataframe[na_columns].isnull().sum().sort_values(ascending=False)
    ratio = (dataframe[na_columns].isnull().sum() / dataframe.shape[0] * 100).sort_values(ascending=False)
    missing_df = pd.concat([n_miss, np.round(ratio, 2)], axis=1, keys=['n_miss', 'ratio'])
    print(missing_df, end="\n")

    if na_name:
        return na_columns

missing_values_table(df)
```

```
                        n_miss  ratio
Electric Utility           473   0.38
Legislative District       297   0.24
Model                      181   0.15
Vehicle Location            29   0.02
County                       2   0.00
City                         2   0.00
Postal Code                  2   0.00
2020 Census Tract            2   0.00
```

METHODOLOGY- EVIDENCE OF COMPLETION

Data Distribution ratio

```python
def missing_values_table(dataframe, na_name=False):
    na_columns = [col for col in dataframe.columns if dataframe[col].isnull().sum() > 0]

    n_miss = dataframe[na_columns].isnull().sum().sort_values(ascending=False)
    ratio = (dataframe[na_columns].isnull().sum() / dataframe.shape[0] * 100).sort_values(ascending=False)
    missing_df = pd.concat([n_miss, np.round(ratio, 2)], axis=1, keys=['n_miss', 'ratio'])
    print(missing_df, end="\n")

    if na_name:
        return na_columns

missing_values_table(df)
```

```
                        n_miss  ratio
Electric Utility           473   0.38
Legislative District       297   0.24
Model                      181   0.15
Vehicle Location            29   0.02
County                       2   0.00
City                         2   0.00
Postal Code                  2   0.00
2020 Census Tract            2   0.00
```

EVIDENCE OF COMPLETION

COMPARISON OF NLP Technologies

LSTM

1. **Long-term Dependency Handling**: LSTMs are designed to capture and maintain information over longer sequences, making them well-suited for tasks like sentiment analysis where understanding the context over extended text is crucial.

2. **Memory Cells**: LSTMs have separate memory cells and gates for storing and retrieving information. This architecture is effective at capturing and preserving information over time.

3. **Reduced Vanishing Gradient Problem**: LSTMs mitigate the vanishing gradient problem more effectively compared to simple RNNs, which helps in capturing long-range dependencies in text data.

GRU

1. **Simplified Gating Mechanism**: GRUs have a simplified gating mechanism compared to LSTMs. While this reduces computational complexity, It can limit their ability to capture complex, long-term dependencies effectively. GRUs may struggle to capture nuanced sentiments embedded deep in text sequences.

2. **Limited Capacity for Long Sequences**: GRUs tend to have limited capacity to capture information from very long sequences. In sentiment analysis, where understanding the context and sentiment over a longer text is vital, this can be a weakness.

3. **Less Control Over Memory**: GRUs have fewer parameters and, as a result, offer less control over the memory process. This can make them less adaptable to complex language patterns and sentiment expressions.

SYSTEM REQUIREMNTS

FUNCTIONAL REQUIREMENTS

- Users initiate breakdown assistance requests, providing descriptions and attaching media.
- The system identifies breakdown types using natural language processing and machine learning.
- Mechanics matching is listed by proximity and expertise.
- Real-time communication, mechanic selection, scheduling, feedback, and location tracking enhance user-mechanic interactions.

NON-FUNCTIONAL REQUIREMENTS

- Performance
- Usability
- Reliability
- Security
- Compatibility
- Maintainability
- Scalability
- Accessibility

SYSTEM REQUIREMENTS

- Machine learning and NLP identify breakdown types and match mechanics by expertise and proximity.
- Real-time communication between users and mechanics is facilitated.
- Mechanic selection options and appointment scheduling are available.
- Feedback and ratings for mechanics' services are supported.
- Accurate location tracking via Google Geolocation API and real-time mapping.

COMPLETION AND FUTURE WORKS

COMPLETION OF COMPONENTS

- DATA GATHERING AND PREPROCESSING
- DATASET APPROVAL
- CREATING THE ACTIONS BASED ON BREAKDOWN DESCRIPTION
- PROVIDING THE BREAKDOWN TYPES
- VERIFY THE PROVIDED ANSWERS USING ML MODEL

FUTURE IMPLEMENTATIONS

- OTHER OBJECTIVES
- IMPLEMENTATION OF USER INTERFACES
- HOSTING

IT20038182

Madhubhashana A.G.K.

Bachelor of Science (Hons) in Information Technology Specializing in Information Technology

Background

- In today's fast-paced world, vehicle ownership is essential, but breakdowns pose challenges in identifying parts and finding reliable repair shops, especially for those with limited automotive knowledge.
- Traditional methods often lead to inefficiencies.
- To tackle this, we propose an innovative solution using AI, image processing, and real-time data.
- Our image-based system accurately recognizes parts, provides real-time compatibility information, and lists nearby shops with the required parts.
- Privacy measures protect user data, and our user-friendly interface empowers vehicle owners to confidently address breakdowns.

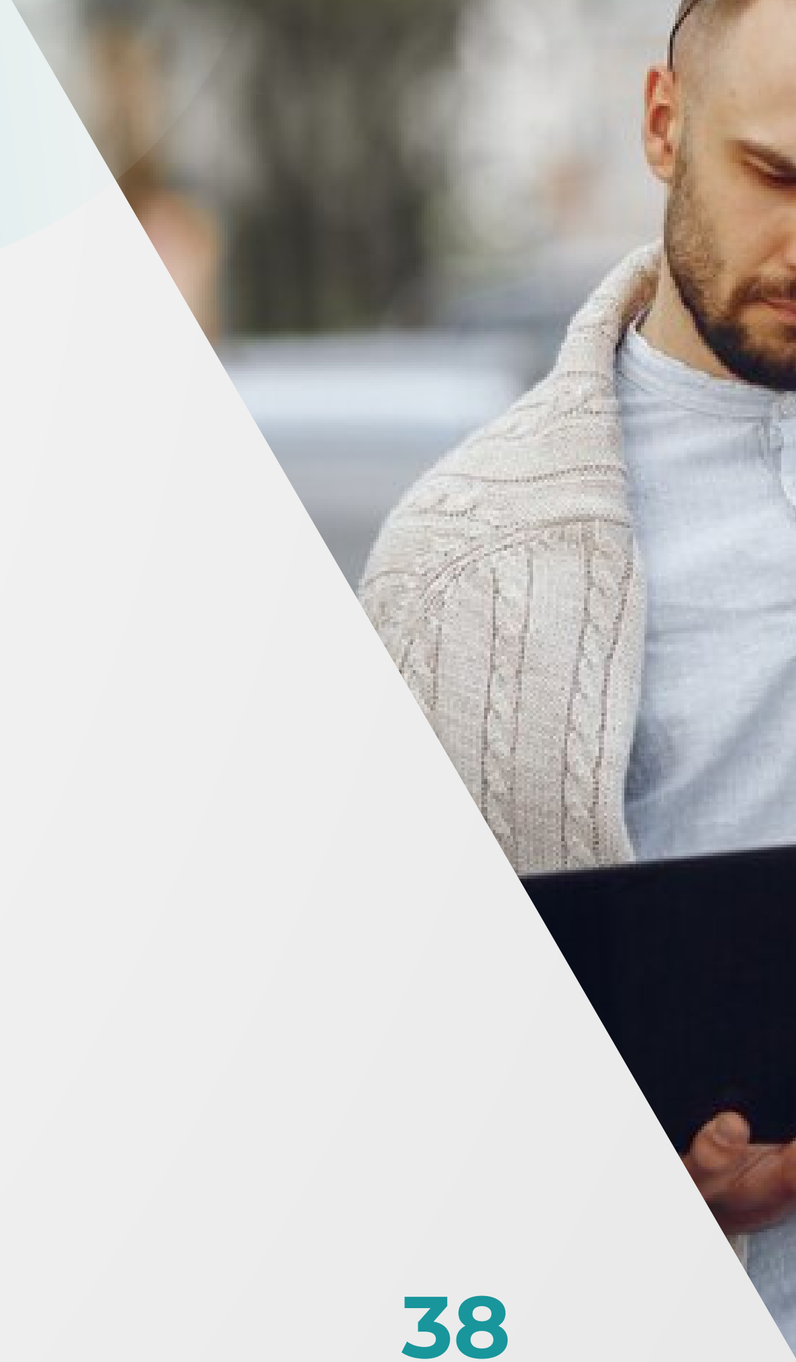
Research Question

“The current issue involves the absence of an efficient and accurate solution for electric vehicle owners to identify spare parts in the event of an unexpected breakdown of an electric vehicle in an unfamiliar location”

There's a need for an AI-based system to identify spare parts and sellers' locations accurately using image processing and machine learning. Continuous learning, transfer learning, and Auto Machine Learning will enhance accessibility and accuracy. Developing such a solution is essential

Research Gap

	Research A	Research B	Research C	Proposed System
Spare Parts Identification	✗	✗	✗	✓
Usage of image processing	✓	✓	✓	✓
Spare parts Identification	✗	✗	✗	✓
Representative datasets	✓	✗	✗	✓
Noise sensitivity	✗	✗	✗	✓



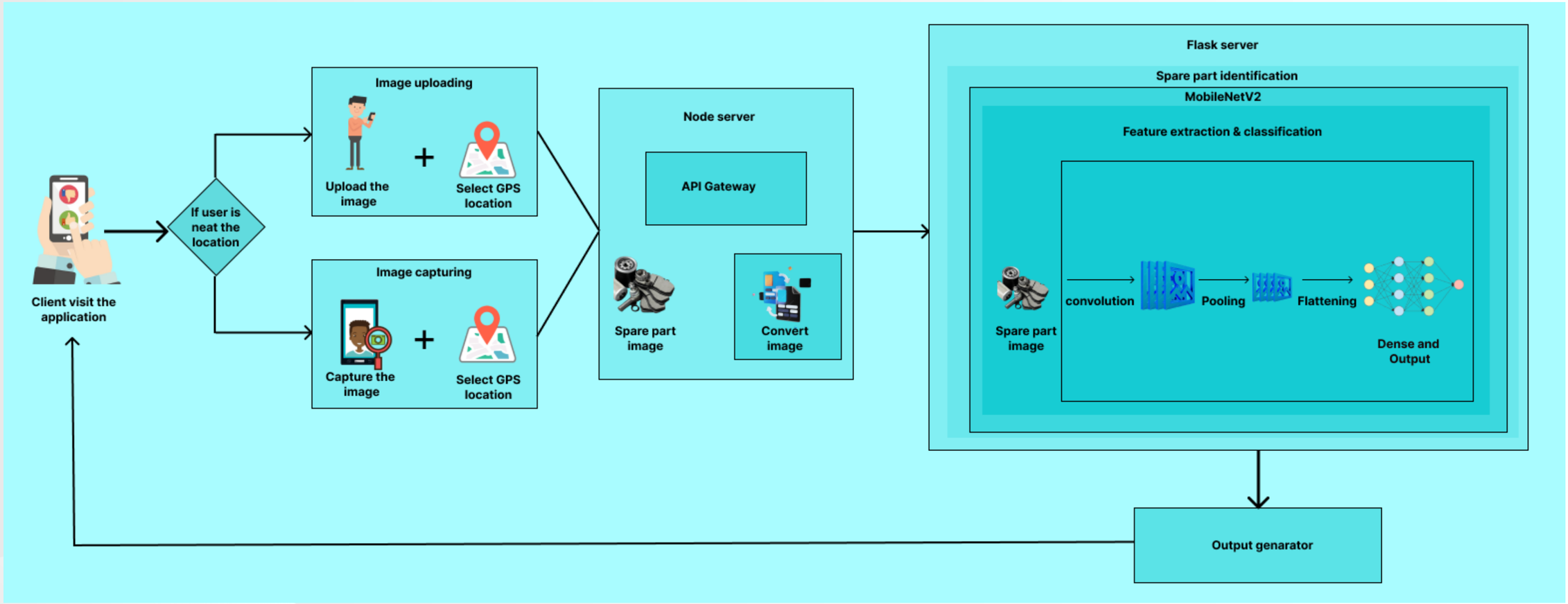
Specific Objective

The research aims to create a real-time image processing system using machine learning to help vehicle owners accurately identify and locate compatible spare parts. It offers convenience through image capture and analysis, enhancing spare parts sourcing

Sub Objectives

- Data Collection
- Data Preprocessing
- ML Model Development
- Implement CNN techniques
- Specialization Matching
- Geolocation Integration
- User Interface
- Testing and Validation
- Scalability and Performance

HIGH LEVEL SYSTEM ARCHITECTURE DIAGRAM



TECHNIQUES & TECHNOLOGIES

METHODOLOGY

Technology	• React Native • Python • Firebase • Node Server • Geo Location • Flask
Techniques	• Image processing • Auto Machine Learning (AML) • CNN_(Conclusion Neural Networks) • ReLU (rectified linear unit)
Model	• MobileNetV2

METHODOLOGY

- DATA COLLECTION
- DATA PRE-PROCESSING
- USE TRANSFER LEARNING BASED CNN ARCHITECTURES FOR TRAINING THE MODELS
- DATA AUGMENTATION AND SELECTED THE BEST TECHNIQUE TO RETRAIN MODELS TO ACHIEVE GOOD FITTING
- TUNING HYPERPARAMETERS AND RETRAINED MODELS TO SELECT THE BEST ARCHITECTURE
- DATA VISUALISATION
- HOST THE FINAL MODEL
- DISPLAYED THE RESULTS IN MOBILE APPLICATION

DATA COLLECTION AND PRE-PROCESSING

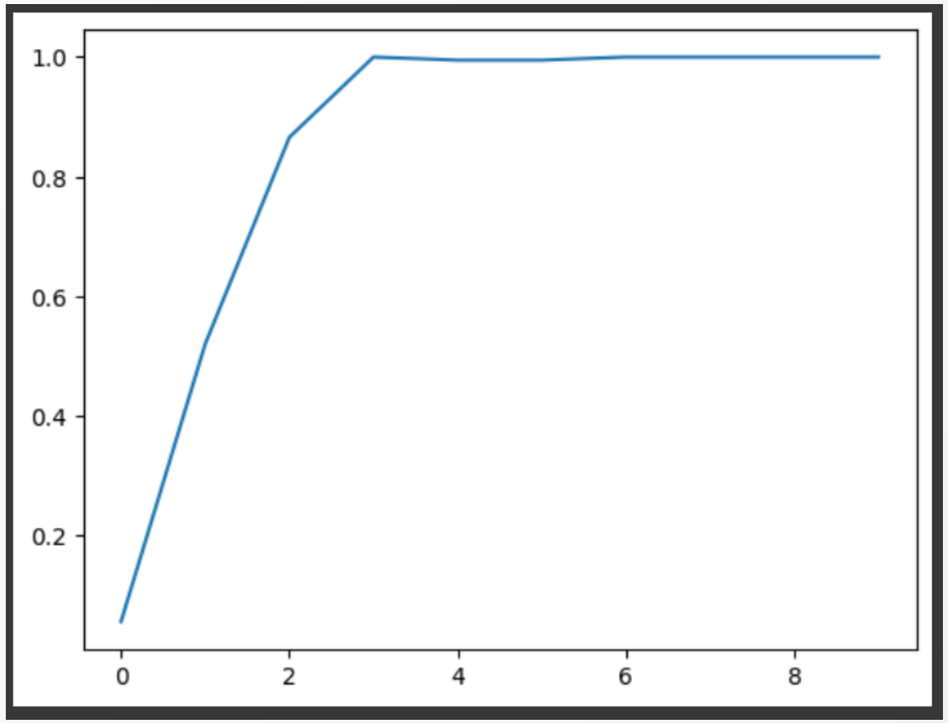
```
Found 194 images belonging to 10 classes.
Found 194 images belonging to 10 classes.
```

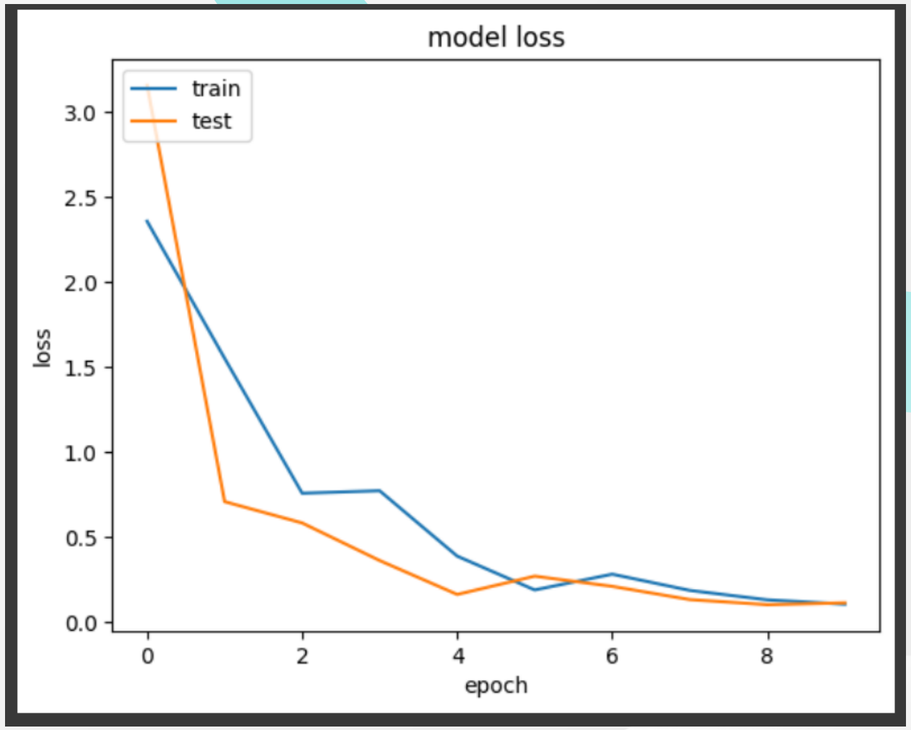
```
Controller
Inventor
BrakingSystem
VehicleInterior
Battery
Transmission
onboardchager
ExteriorComponents
Elecricmotor
CoolingSystem
```

ACCURACY AND LOSS

```
Epoch 1/10
1/1 [==============================] - ETA: 0s - loss: 2.5520 - accuracy: 0.0928/usr/local/lib/python3.10/dist-packages/keras/src/engine/training.py:3079: UserWarning
  saving_api.save_model(
1/1 [==============================] - 90s 90s/step - loss: 2.5520 - accuracy: 0.0928 - val_loss: 3.0553 - val_accuracy: 0.2371
Epoch 2/10
1/1 [==============================] - 16s 16s/step - loss: 1.3238 - accuracy: 0.5773 - val_loss: 3.6684 - val_accuracy: 0.1856
Epoch 3/10
1/1 [==============================] - 16s 16s/step - loss: 0.7348 - accuracy: 0.8299 - val_loss: 4.2476 - val_accuracy: 0.2371
Epoch 4/10
1/1 [==============================] - 17s 17s/step - loss: 0.3178 - accuracy: 0.9278 - val_loss: 4.9126 - val_accuracy: 0.2526
Epoch 5/10
1/1 [==============================] - 12s 12s/step - loss: 0.1250 - accuracy: 0.9845 - val_loss: 5.5901 - val_accuracy: 0.2526
Epoch 6/10
1/1 [==============================] - 13s 13s/step - loss: 0.0703 - accuracy: 0.9897 - val_loss: 6.1774 - val_accuracy: 0.2526
```

SPARE PARTS IDENTIFICATION ACCURACIES





EVIDENCE OF COMPLETION
COMPARISONS OF ARCHITECTURES

VGG19

Complexity and Efficiency
A deep architecture with 19 layers, excels at accurate feature extraction but is computationally intensive and memory-demanding, making it less efficient for real-time tasks.

Model Size
Larger model size, a consequence of its deep layers and parameters, is impractical for resource-constrained devices.

Accuracy
Offers high accuracy, suitable for precision-critical tasks.

Speed and Inference Time
Depth and complexity make it slower for inference, limiting its suitability for real-time tasks.

Deployment Flexibility
Suits scenarios with ample computational resources, like cloud-based applications.

MobileNetV2

Complexity and Efficiency
Designed for mobile and embedded devices, prioritizing efficiency and real-time processing due to its lightweight design.

Model Size
Small model size is ideal for mobile and edge devices with limited memory and storage
Accuracy
Sacrifices a bit of accuracy for efficiency but remains accurate for many applications.
.
Speed and Inference Time
Optimized for low latency and real-time processing.

Deployment Flexibility
Flexible and thrives in resource-constrained environments, catering to efficiency and real-time demands. Your choice should align with your specific application's requirements, resources, and real-time needs

SYSTEM REQUIREMNTS

FUNCTIONAL REQUIREMENTS

Image-Based Spare Parts Identification enables real-time camera use for capturing spare parts, accurately identifying their type and condition, and presenting comprehensive details, including manufacturing specifics and pricing.

Spare Parts Shop Registration offers a mechanism for shops to join, collect and store their details, while the Spare Parts Shop Database maintains current spare part data and allows real-time inventory updates through seamless integration.

NON-FUNCTIONAL REQUIREMENTS

- Performance
- Usability
- Reliability
- Security
- Compatibility
- Maintainability
- Scalability
- Accessibility

SYSTEM REQUIREMENTS

- Image processing capabilities Machine learning algorithms
- Database integration
- Mobile compatibility
- Scalability
- Performance
- Usability
- Security
- Reliability
- Integration with third-party APIs
- System backups and data recovery

COMPLETION AND FUTURE WORKS

COMPLETION OF COMPONENTS

- DATA GATHERING AND PREPROCESSING
- DATASET APPROVAL
- TRAINING THE MACHINE LEARNING MODEL
- INITIAL SPARE PARTS RECOGNITION

FUTURE IMPLEMENTATIONS

- EVALUATING AND FINE TUNING OF THE MODEL
- LOCATION MAPPING
- AUTO MACHINE LEARNING IMPLEMENTATION TO ADD NEW PLANTS EASILY

IT200644826

Vijerathna A.G.V.K.M

Enhancing User Experience in Vehicle Maintenance with a Smart Digital Monitoring Calendar and Reminder System

Bachelor of Science (Hons) in
Information Technology Specializing in Information Technology

INTRODUCTION

Background

- system for electric vehicle maintenance using Machine learning and image OCR.
- **Efficient record management** is important. It organizes vast maintenance data, including service records, parts, and user history, ensuring easy access.
- Streamlining maintenance data, the research integrates Optical Character Recognition **(OCR)** tech.
- Central to this research is the development and deployment of **predictive algorithms**. These algorithms analyze historical service data, usage patterns, and driving conditions to predict ideal spare part replacement times.
- The research proposes an innovative **digital monitoring calendar** for users to input, track, and receive service reminders, promoting proactive maintenance.

Research Question

How can a systematic approach, combining proactive monitoring, streamlined record-keeping, predictive algorithms, and a digital monitoring calendar, revolutionize vehicle maintenance to prevent breakdowns efficiently?

"How to avoid breakdowns of Vehicle through proper maintenance and smart digital monitoring Calendar system ?"

“How to calculate the time of next service after doing a repair to certain part of a vehicle? ”

“How to track average maintenance cost based on vehicle type for a period?”

Research Gap

Features	Research A	Research B	Research C	Proposed System
Smart Digital Monitoring calendar	✗	✗	✗	✓
Predictive algorithms to calculate future repairs	✗	✓	✗	✓
Availability of personalized solutions	✓	✗	✗	✓
Using OCR technology	✗	✗	✗	✓
User Friendly Platform	✓	✗	✓	✓

Specific Objective

Create a system for improved vehicle maintenance with proactive monitoring, OCR-based bill capture, and digital calendars. Empower users with informed decisions for optimal performance and Durability.

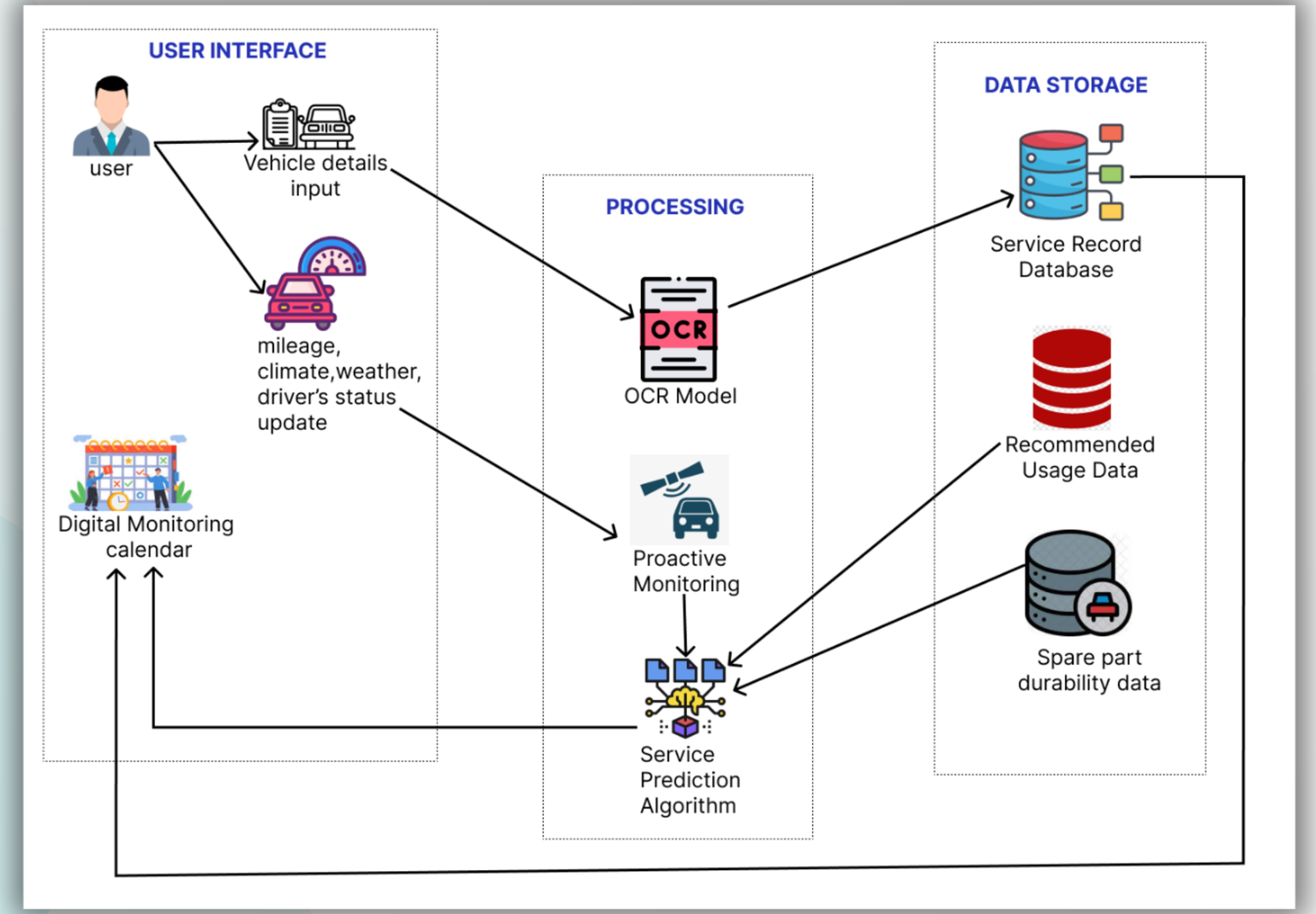
Sub Objectives

- Research and Data Gathering
- Data Preprocessing
- OCR Model Implementation
- Predictive Algorithm Development
- Digital Monitoring Calendar System Design
- Database Creation
- User Interface Development
- Testing and Validation
- Scalability and Performance

METHODOLOGY

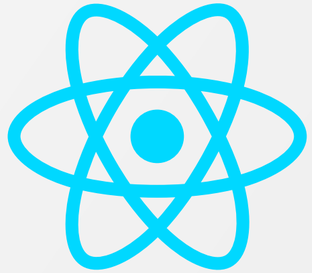
The solution I have suggested is very important for anyone who owns an electric vehicle. With this remedy, the owner of the electric vehicle can do his repairs without missing anything. The system collects the details of the spare parts, information about the vehicle, information about the road the vehicle is driven on, the weather in the area, the nature of the person driving the vehicle, etc. and informs the user about future repairs. System present a user friendly tool through a digital monitoring calendar. By means of an algorithm, system inform the user about possible repairs in the future and stop sudden breakdowns that can happen to the vehicle.

System Architecture Diagram



TECHNIQUES & TECHNOLOGIES

Technology	• React Native • Python • Flask • FireBase • Node Server
Techniques	• Python image OCR • Machine Learning (ML) • sentiment analysis
Algorithm	• Polynomial regression

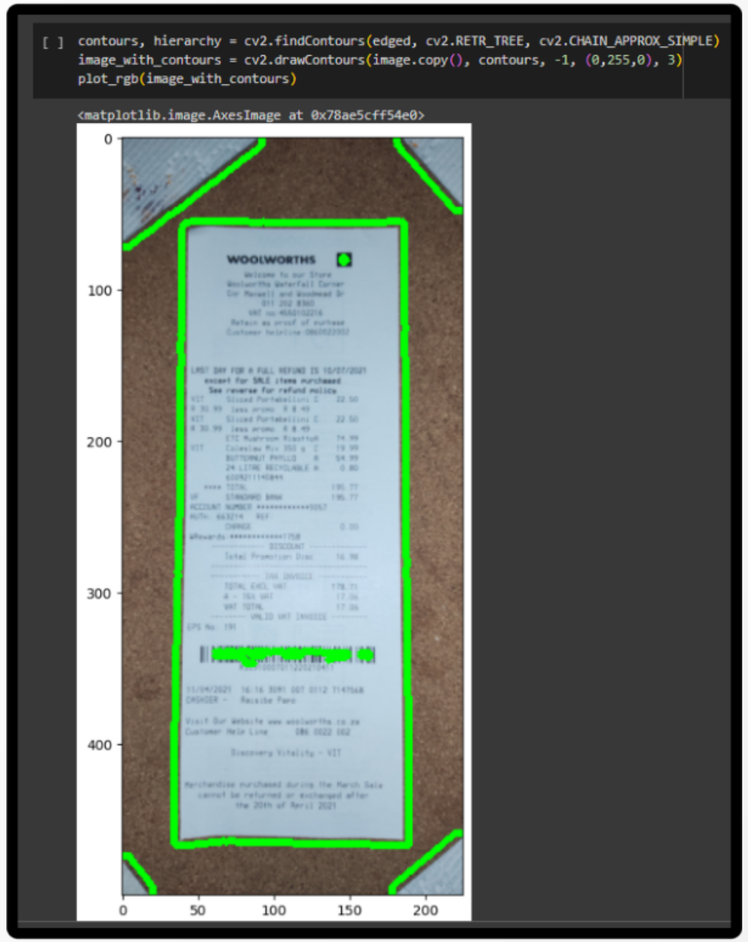


METHODOLOGY

- Research and Data Gathering
- DATA PRE-PROCESSING
- Implementation of OCR Model
- Algorithm Development
- Digital Monitoring Calendar System Design
- Integration of Proactive Monitoring and Record-Keeping Implementation
- HOST THE FINAL MODEL
- DISPLAYED THE RESULTS IN MOBILE APPLICATION

METHODOLOGY

EVIDENCE OF COMPLETION



Identify the edges of coloured Bill

Bill image after removing the background areas of the bill

METHODOLOGY

EVIDENCE OF COMPLETION

```
n_boxes = len(d['level'])
boxes = cv2.cvtColor(image.copy(), cv2.COLOR_BGR2RGB)
for i in range(n_boxes):
    (x, y, w, h) = (d['left'][i], d['top'][i], d['width'][i], d['height'][i])
    boxes = cv2.rectangle(boxes, (x, y), (x + w, y + h), (0, 255, 0), 2)

plot_rgb(boxes)
```

```
<matplotlib.image.AxesImage at 0x78ae5cec4df0>
```

Identify the texts in Bill

```
[ ]  t = PrettyTable(['Item', 'Cost'])
     for counter in range (0,len(food)):
         t.add_row([food[counter], food_item_cost[
     print(t)
```

```
+-----------------------------------+---------+
|                Item               |   Cost  |
+-----------------------------------+---------+
|           less erg RED            |  230.99 |
|         shee Poriaeettins         |  22.50  |
|           ere wrod BBD            |  30.99  |
|       EMC haatroo Risotto         |  74.99  |
|           eeu RECHEURRE           |   0.82  |
|             sees fora             |  195.77 |
|                                   |   0.00  |
|          Totot ronslaen Blac      |  16.38  |
| visit Doe Website wee wovorityco  |   .20   |
+-----------------------------------+---------+
```

Output of the bill

EVIDENCE OF COMPLETION

COMPARISONS OF OCR Models

Python OCR Model	Tesseract OCR Engine
Offers high accuracy through custom training and fine-tuning.	Provides good accuracy but might require extensive training for specific use cases.
Highly customizable, allowing for fine-tuning and adaptation to unique requirements.	Offers customization options but may require deeper technical knowledge for optimization.
Seamlessly integrates with Python-based applications and workflows.	Requires extra effort to integrate into Python and lacks native Python support.
Benefits from the active Python open-source community, ensuring ongoing development and support.	Has its community but may not be as extensive as Python's.
Strong multi-language support, including less common scripts.	Strong multi-language support, including less common scripts.
Simplifies model training with Python tools and resources.	Offers model training, but it requires expertise and specific data preparation.
Proficient in recognizing diverse document types, from invoices to handwritten text.	Proficient in recognizing diverse document types, from invoices to handwritten text.

SYSTEM REQUIREMNTS

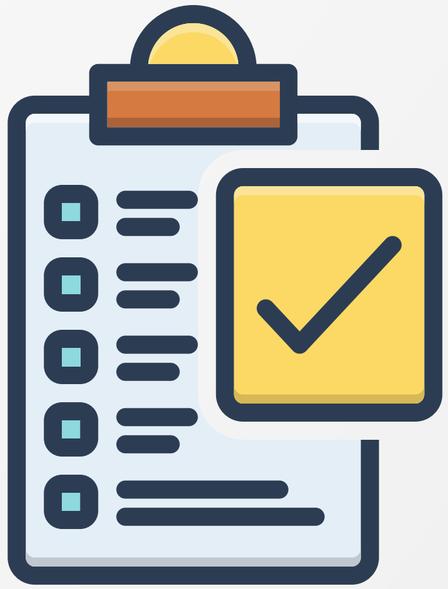
FUNCTIONAL REQUIREMENTS

- User Registration and Vehicle Details Input.
- Users should be able to periodically update their vehicle's mileage.
- Users should be able to upload images of service bills and invoices.
- The system should predict the optimal replacement time for each spare part based on vehicle mileage and historical service records.
- Users should be able to input and track maintenance activities, spare part replacements, and relevant data through a digital monitoring calendar.
- Alert system should notify users about upcoming service requirements.
- System should be able to accurately extract of relevant maintenance information from bill photos.

SYSTEM REQUIREMNTS

NON-FUNCTIONAL REQUIREMENTS

- Performance
- Security
- Reliability
- Scalability
- Compatibility
- Maintainability
- Usability



SYSTEM REQUIREMENTS

- OCR-Based Bill Capture
- Machine learning algorithms
- Database integration
- Mobile compatibility
- Performance
- Integration with predictive algorithms for generating service reminders. Reliability
- System backups and data recovery
- Data exchange protocols and APIs for component integration.

COMPLETION AND FUTURE WORKS

COMPLETION OF COMPONENTS

FUTURE IMPLEMENTATIONS

- DATA GATHERING AND PREPROCESSING
- TRAINING THE MACHINE LEARNING MODEL
- OPTICAL CHARACTOR RECOGNITION
- PREDICTIVE ALGORITHM
- IMPLEMENTATION OF USER INTERFACES
- DIGITAL MONITORING CALENDAR
- HOSTING – IN PROGRESS

THANK YOU