PROJECT REPORT

INTELLIGENT GARBAGE CLASSIFICATION USING DEEP LEARNING

Mentor Details:

Dr.S.Ariffa Begum, Ph.D.,

Associate Professor, Department of Computer science and engineering,
Vaigai college of engineering,

TEAM MEMBERS ;

1.SANTHOSH.S
2.HARIHARASUDHAN.C
3.JEEVASRI.B
4.VIJI.S

CONTENT

Project Report Format

INTELLIGENT GARBAGE CLASSIFICATION USING DEEP LEARNING

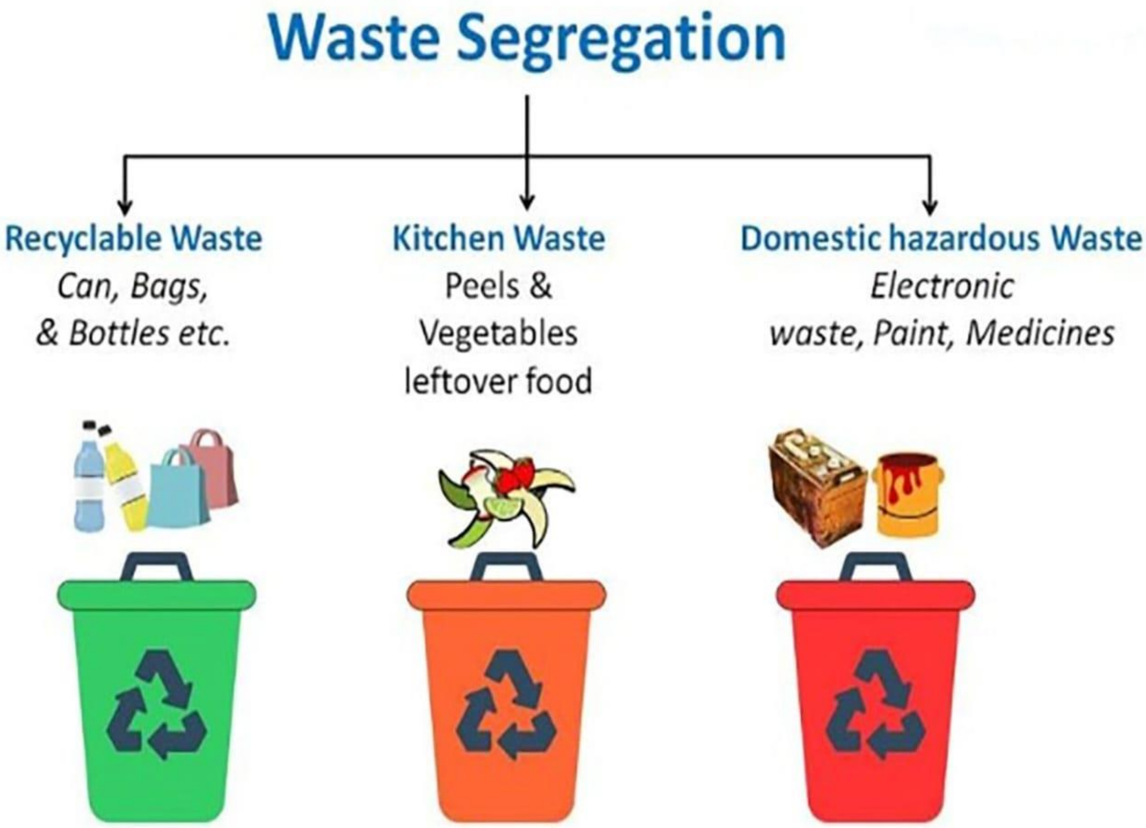
1. INTRODUCTION

1.1 Project Overview

Garbage classification is an important task for waste management and envisronmental sustainability. Deep learning techniques have shown promising results in various computer vision tasks, including image classification. This project aims to develop a garbage classification system using deep learning algorithms to automatically classify different types of waste into appropriate categories. This includes the following processes like data acquisition, data pre-processing, model selection, model training, model evaluation, model deployment and continuous improvement.

1.2 Purpose

The purpose of garbage classification using deep learning is to automate and improve the efficiency of waste management systems. Garbage classification involves sorting different types of waste into specific categories such as recyclables, non-recyclables, organic waste, hazardous materials, and so on. Traditionally, this task has been performed manually by humans, which can be time-consuming, labour - Intensive, and prone to errors. Deep learning, a subfield of artificial intelligence, offers a solution by leveraging neural networks to automatically recognize and classify garbage items based on their visual features.



2. IDEATION & PROPOSED SOLUTION

2.1 Problem Statement Definition

The management of solid waste in large urban environments has become a complex problem due to increasing amount of waste generated every day by citizens and companies. One of the most important steps of waste management is the separation of the waste into the different components and this process is normally done manually by hand-picking.

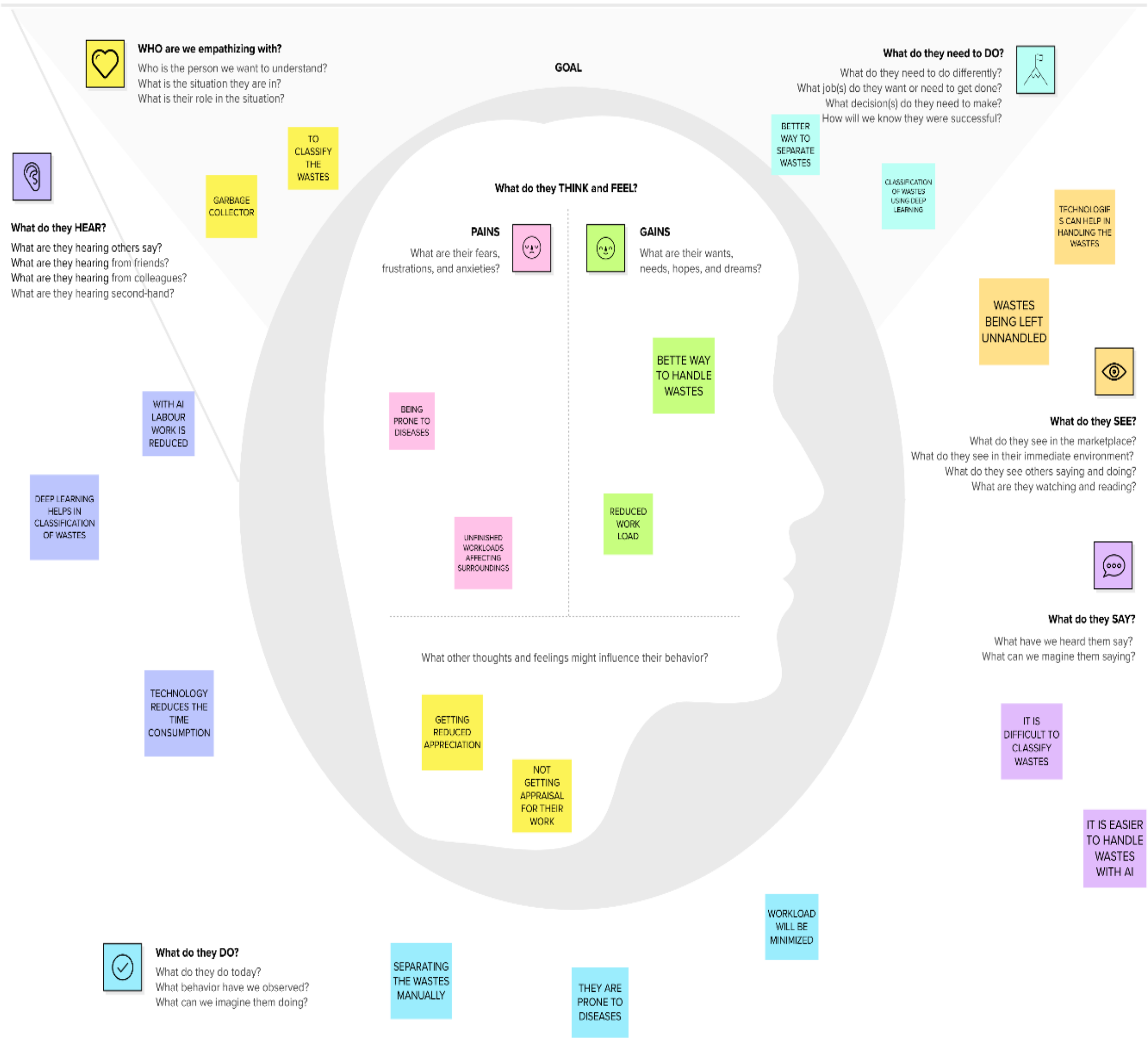
Since all types of wastes are dumped together in the same place, as a garbage collector, it is difficult to separate all the wastes manually.

Problem Statement (PS)	I am (Customer)	I'm trying to	But	Because	Which makes me feel
PS-1	A garbage collector	Keep my environment clean and safe	Its physically impossible	It takes too much labor work to classify the waste and dispose	Desperate

2.2 Empathy Map Canvas

Since Garbage collectors are the front line workers when it comes to separating the wastes. They are always in a situation which can make them prone to diseases. Even after doing tons of work load, they are not getting enough appraisal.

Their burdens can be reduced with the help of deep learning and by using this technology they can classify the wastes easily with much less time consumption.

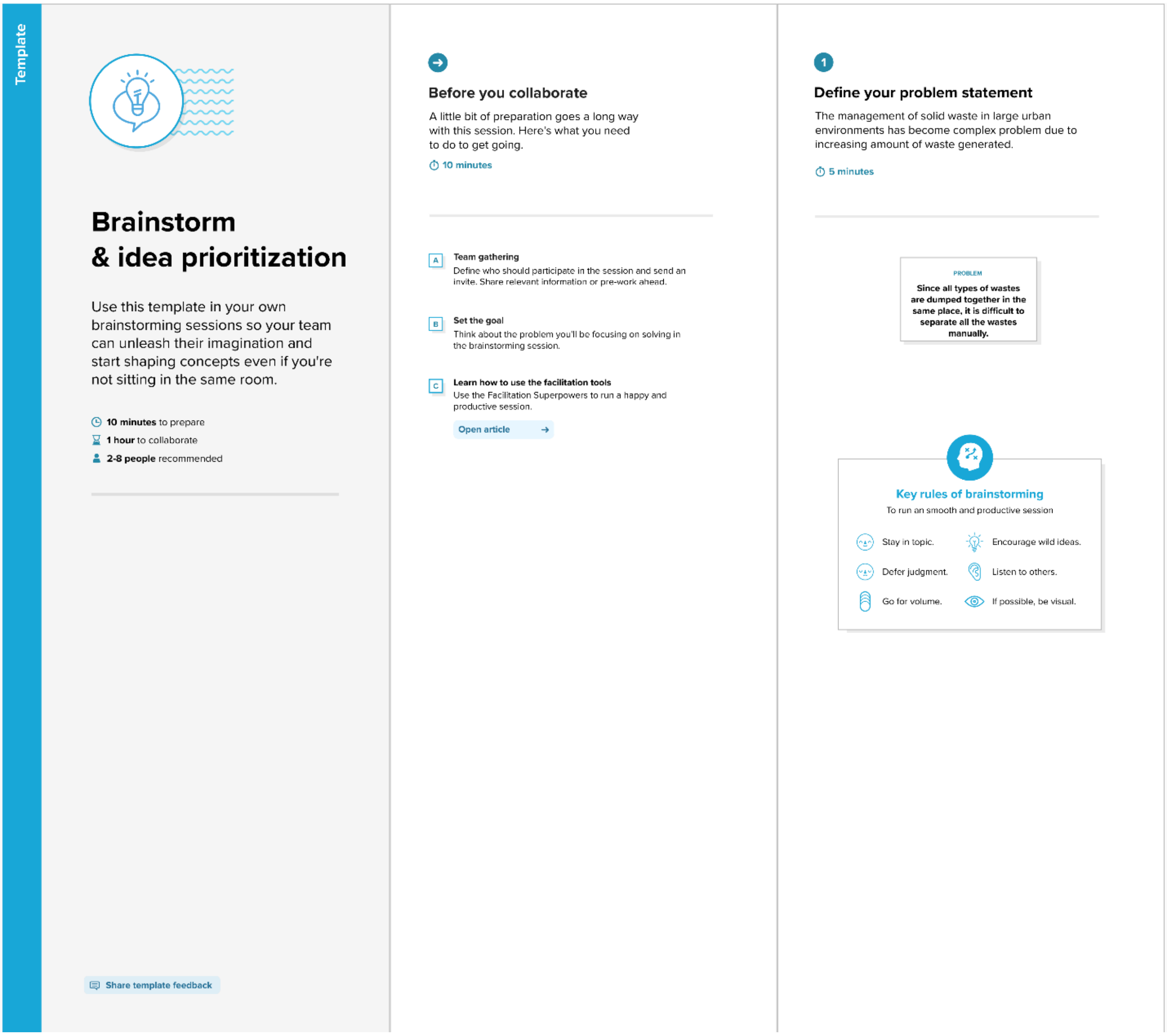


2.3 Ideation & Brainstorming

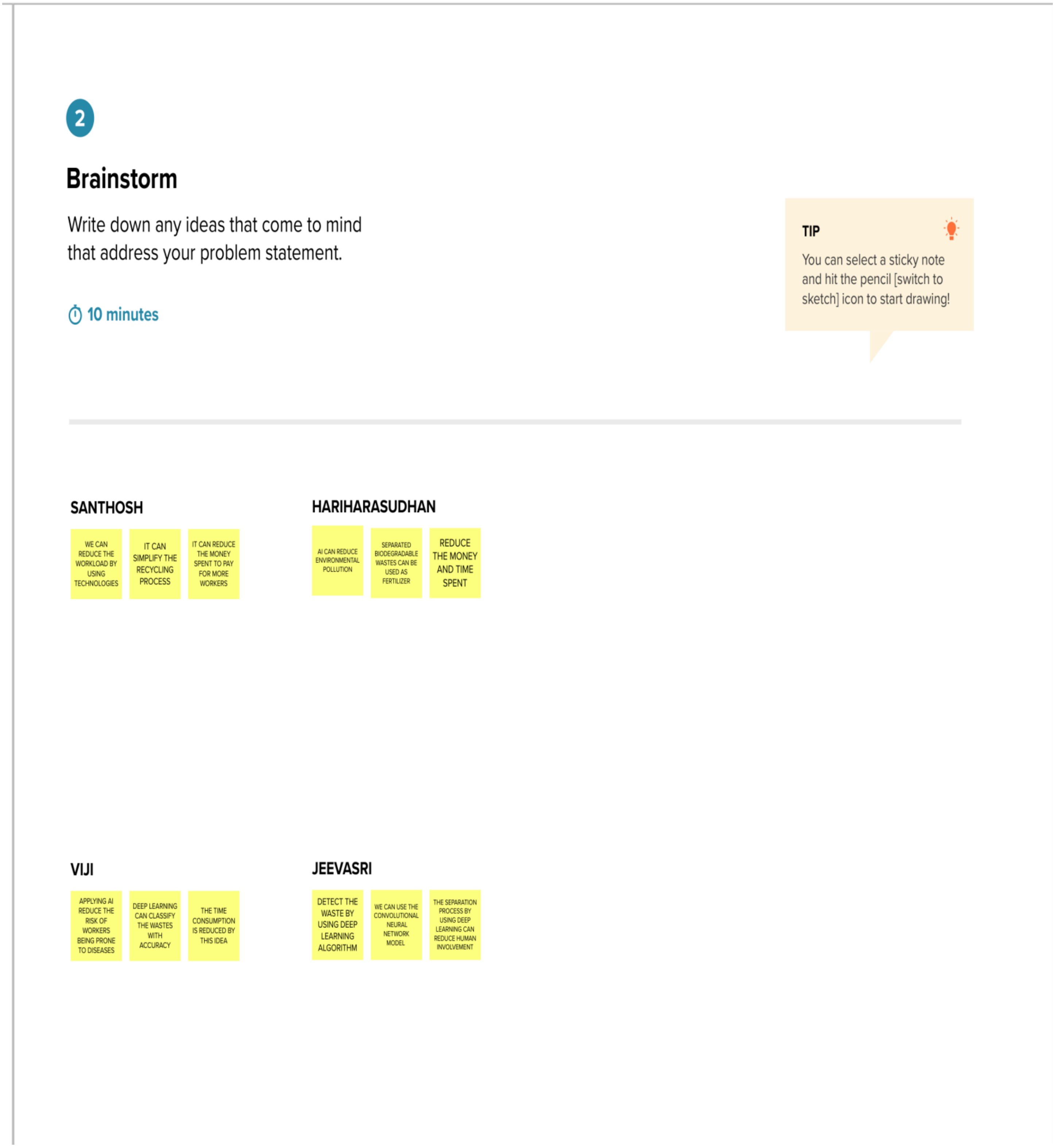
This mainly tells about the solution to the problem we are facing in the garbage classification. Each person gives their individual solution to the problem and after giving their solutions we are grouping the same solutions given by the each persons.

Lastly, we have to graph the solutions according to the feasibility and importance. This is the brainstorm idea which tells about the solutions given by the individual persons for the problem facing in garbage classification.

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



Step-2: Brainstorm, Idea Listing and Grouping



3

Group ideas

Take turns sharing your ideas while clustering similar or related notes as you go. Once all sticky notes have been grouped, give each cluster a sentence-like label. If a cluster is bigger than six sticky notes, try and see if you and break it up into smaller sub-groups.

20 minutes

TIP

Add customizable tags to sticky notes to make it easier to find, browse, organize, and categorize important ideas as themes within your mural.

THE TIME CONSUMPTION IS REDUCED BY THIS IDEA

REDUCE THE MONEY AND TIME SPENT

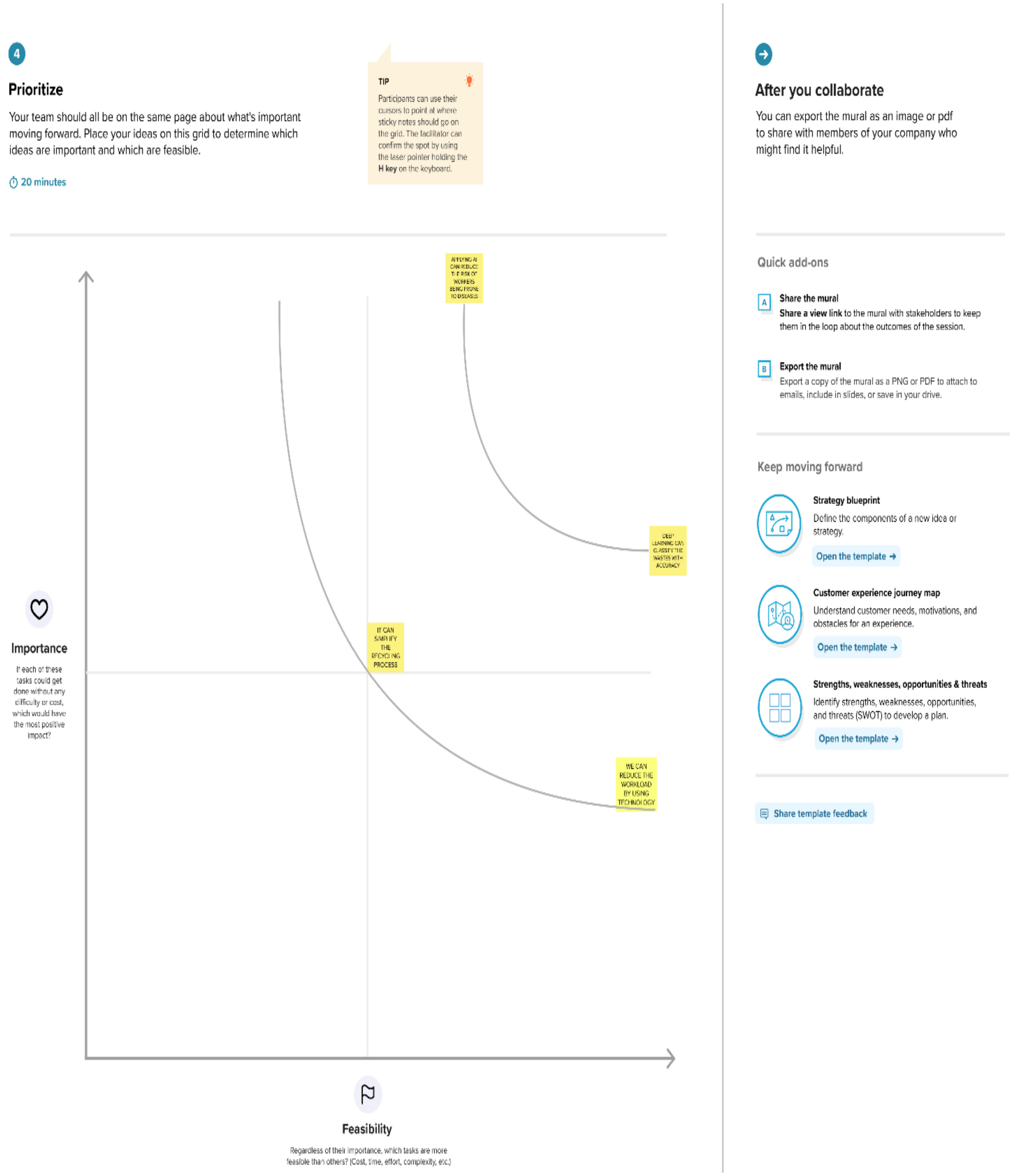
DEEP LEARNING CAN CLASSIFY THE WASTES WITH ACCURACY

DETECT THE WASTE BY USING DEEP LEARNING ALGORITHM

WE CAN REDUCE THE WORKLOAD BY USING TECHNOLOGIES

THE SEPARATION PROCESS BY USING DEEP LEARNING CAN REDUCE HUMAN INVOLVEMENT

Step-3: Idea Prioritization



2.4 Proposed Solution

S.No.	Parameter	Description
1.	Problem Statement (Problem to be solved)	The problem that we are facing here is the difficulty when it comes to the process of efficiently classifying the wastes.
2.	Idea / Solution description	Garbage classification using computer vision
3.	Novelty / Uniqueness	Garbage classification using deep learning provides accurate classification with the help of CNN. Convolutional Neural Networks (CNNs) is an effective tool for analyzing big data that uses complex algorithms and artificial neural networks to train machines so that they can learn from experience, classify and recognize data or images.
4.	Social Impact / Customer Satisfaction	This can reduce the work load and time consumed by the workers. Applying AI reduce the risk of workers being prone to diseases
5.	Business Model (Revenue Model)	Used to recycle the waste, which makes it more valuable **Waste-to-Energy**: Some types of garbage can be used to generate energy through waste-to-energy facilities, which can then be sold to power companies. **Waste Collection Fees**: Municipalities and waste management companies can charge fees for garbage collection and disposal services, with higher fees for larger or non-segregated waste
6.	Scalability of the Solution	Garbage classification using deep learning has shown great potential for scalability. With the increasing amount of waste generated by modern societies, garbage classification has become an important issue for waste management. Deep learning models can be trained on large datasets to accurately classify different types of waste, including recyclables, non-recyclables, and organic waste. One of the key advantages of deep learning models is their ability to scale with larger datasets. As more data is collected and added to the training set, the accuracy of the model can be improved.

2. REQUIREMENT ANALYSIS

3.1 Functional requirement

FR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	**User Interface**	System should be user-friendly Easy to navigate System should give clear instructions on how to use the system
FR-2	**Input mechanism**	System should provide a mechanisms for users to input images of their waste items It can be in the form of mobile or web application
FR-3	**Deep learning model**	A model that is capable of classifying wastes into different categories The model should be trained on a large dataset of images of different waste items
FR-4	**Classification output**	The system should provide an output to the user indicating the category of waste item they have inputted This could be in the form of a text description, a color-coded label, or an audio message
FR-5	**Feedback mechanism**	The system should provide feedback to users on the accuracy of its classification If the system misclassifies an item, the user should be given an option to provide feedback to improve the system's accuracy in the future
FR-6	**Support and maintenance**	The system should address any technical issues occurs for users The system should also have the capability of periodic maintenance to ensure the proper function

3.2 Non-Functional requirements

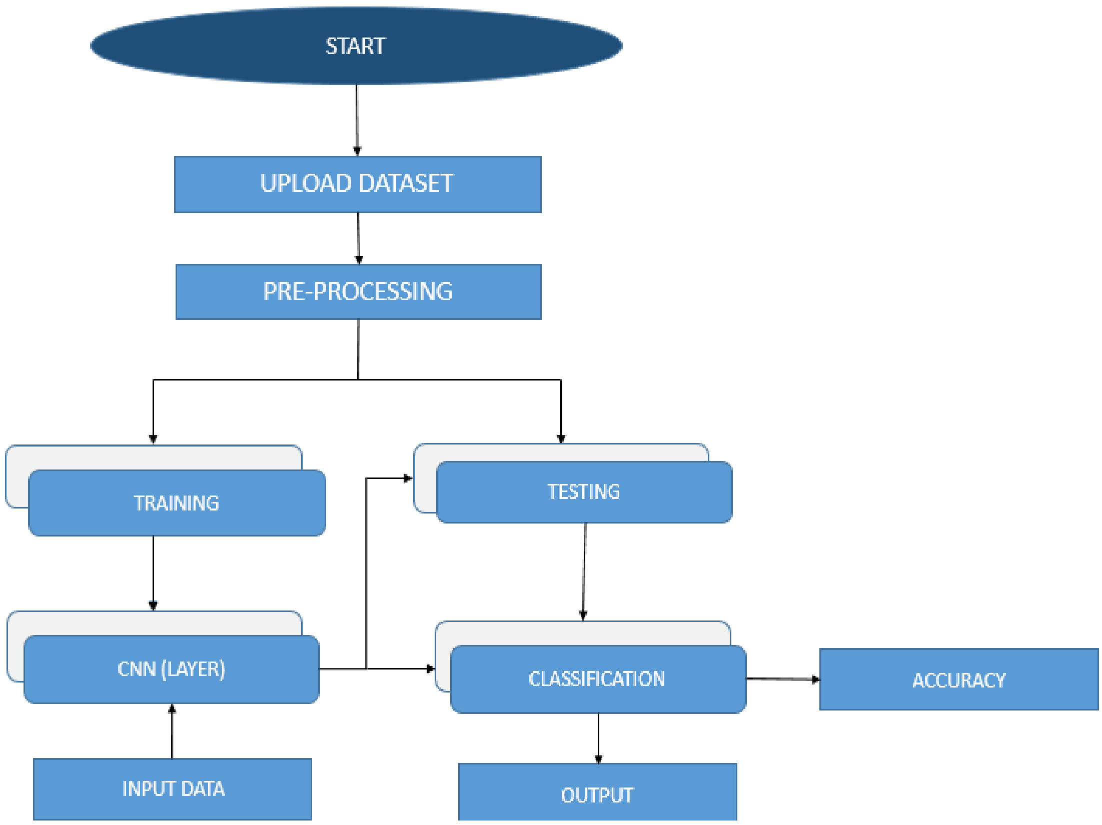
SR No.	Functional Requirement (Epic)	Sub Requirement (Story / Sub-Task)
FR-1	**User Interface**	System should be user-friendly Easy to navigate System should give clear instructions on how to use the system
FR-2	**Input mechanism**	System should provide a mechanisms for users to input images of their waste items It can be in the form of mobile or web application

FR-3	**Deep learning model**	A model that is capable of classifying wastes into different categories The model should be trained on a large dataset of images of different waste items
FR-4	**Classification output**	The system should provide an output to the user indicating the category of waste item they have inputted This could be in the form of a text description, a color-coded label, or an audio message
FR-5	**Feedback mechanism**	The system should provide feedback to users on the accuracy of its classification If the system misclassifies an item, the user should be given an option to provide feedback to improve the system's accuracy in the future
FR-6	**Support and maintenance**	The system should address any technical issues occurs for users The system should also have the capability of periodic maintenance to ensure the proper function

4. PROJECT DESIGN

4.1 Data Flow Diagrams

The following data diagram explains about the steps that are involved in the process of garbage calssification using deep learning.Its starts with process of giving input (an image) which will be uploaded and preprocessed in order to analyse them clearly.The testing and training plays a vital role in the analysing methodology.

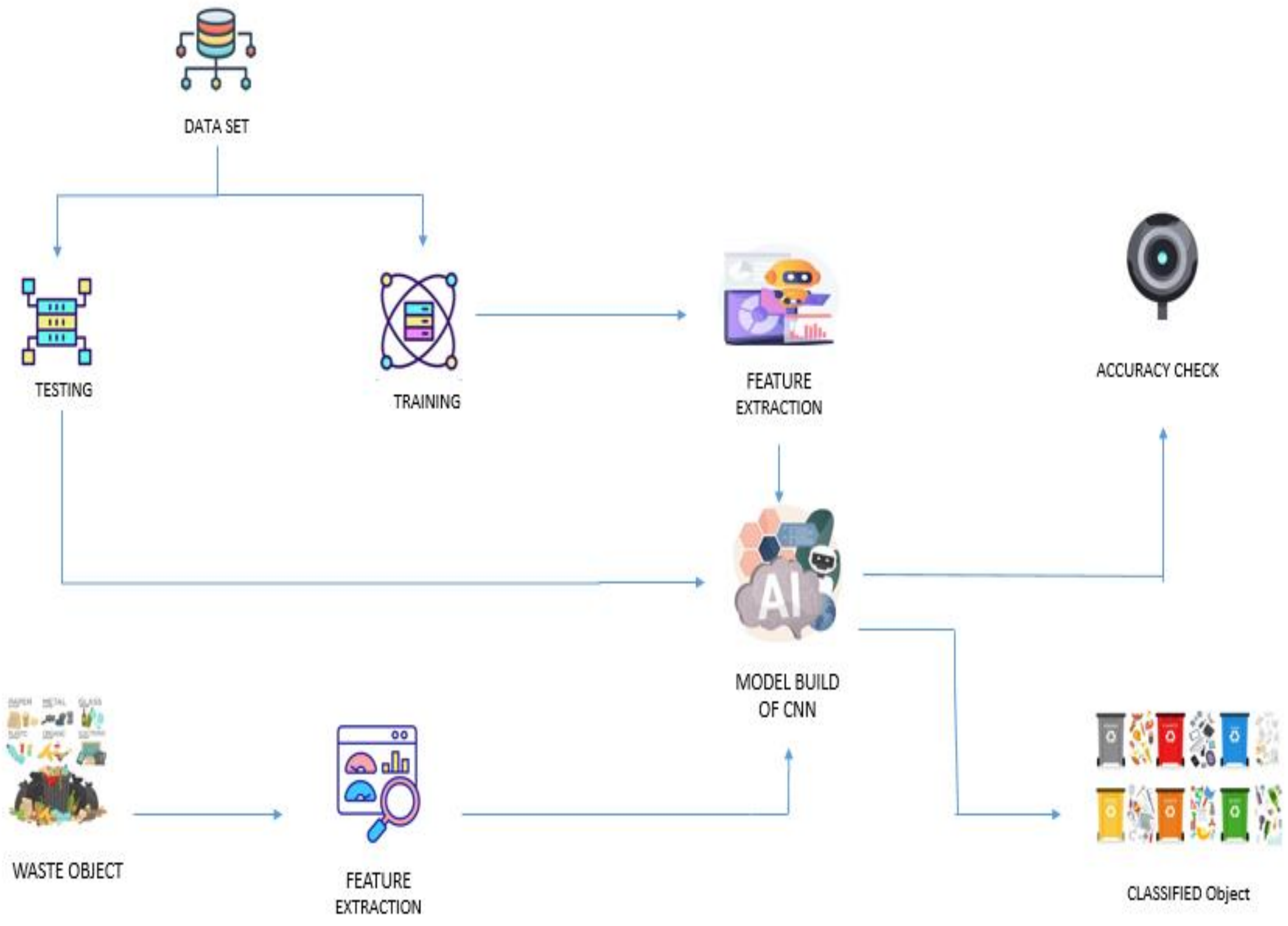


With the help of CNN a better classification is done with the use of many layers that result in greater accuracy.

4.2 Solution & Technical Architecture

The solution architecture for intelligent garbage classification using deep learning typically involves several components working together to achieve accurate classification of different types of waste.

1. **Data Collection and Pre-processing**: This component involves collecting and preparing the data necessary for training the deep learning model. The data could include images or video footage of waste, along with metadata such as the type of waste and any other relevant information.
2. **Deep Learning Model**: This component involves designing and training a deep learning model that can accurately classify different types of waste. Convolutional Neural Networks (CNN) are widely used for image classification tasks and can be used for this purpose.
3. **Deployment Infrastructure**: Once the deep learning model is trained, it needs to be deployed to an infrastructure where it can classify waste in real-time. This could be a cloud-based infrastructure, an edge device such as a Raspberry Pi, or a dedicated server.
4. **User Interface**: The user interface could be a mobile application or a web-based interface, where users can take a picture of waste, upload it to the system, and receive the classification result.



4.3 User Stories

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
Customer	**User interface**	USN-1	As a user, I can use the system with a given clear instructions	I can use the system easily	**High**	Hari
	Input mechanism	USN-2	As a user, I can upload the images of waste in order to classify them	I can classify the waste by uploading the images	**High**	Jeevasri
	Classification output	USN-3	As a user, I'll classify the waste based on the indication given by the system	I can classify the wastes by the indication given	**High**	Viji
	Deep learning model	USN-4	As a user, I can classify wastes into different categories based on this model	I can classify the wastes into different categories	**High**	Santhosh
Customer Care Executive	**Feedback mechanism**	USN-5	As a customer care executive, I can analyse the accuracy of the classification based on the feedback	I can analyse the accuracy of the classification	Medium	Jeevasri

User Type	Functional Requirement (Epic)	User Story Number	User Story / Task	Acceptance criteria	Priority	Team Member
			provided by the users			
Administrator	**Support and maintenance**	USN-6	As an administrator, I can supervise any technical issues which occurs in the system	I can supervise all the technical issues	Medium	Viji

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

Garbage classification using deep learning involves training a model to classify different types of garbage based on input images. Here are some coding and solutioning features used in this project.

CODE

CNN

```
!pip install -q kaggle
```

```
!mkdir ~/.kaggle
```

```
!cp kaggle.json ~/.kaggle
```

```
!kaggle datasets download -d asdasdasasdas/garbage-classification
```

```
Warning: Your Kaggle API key is readable by other users on this system! To fix this, you can run 'chmod 600 /root/.kaggle/kaggle.json'
Downloading garbage-classification.zip to /content
 77% 63.0M/82.0M [00:00<00:00, 223MB/s]
100% 82.0M/82.0M [00:00<00:00, 230MB/s]
```

```
!unzip /content/garbage-classification.zip
```

```
Streaming output truncated to the last 5000 lines.
  inflating: Garbage classification/Garbage classification/cardboard/cardboard152.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard153.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard154.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard155.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard156.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard157.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard158.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard159.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard16.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard160.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard161.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard162.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard163.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard164.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard165.jpg
  inflating: Garbage classification/Garbage classification/cardboard/cardboard166.jpg
```

Data Augmentation

```
#import necessary libraries

from tensorflow.keras.preprocessing.image import ImageDataGenerator
```

```
# data augmentation for training variable

train_datagen = ImageDataGenerator(rescale = 1./255,zoom_range = 0.2,horizontal_flip=True)
```

```
# data augmentation for testing variable

test_datagen = ImageDataGenerator(rescale = 1./255)
```

```
# data augmentation on the training data

xtrain = train_datagen.flow_from_directory('/content/GARBAGE-DATASET/train',
                                          target_size =(64,64),
                                          class_mode = 'categorical',
                                          batch_size = 100)
```

```
Found 2144 images belonging to 6 classes.
```

```
# data augmentation on the testing data
xtest = test_datagen.flow_from_directory('/content/GARBAGE-DATASET/validation',
                                          target_size =(64,64),
                                          class_mode = 'categorical',
                                          batch_size = 100)
```

```
Found 383 images belonging to 6 classes.
```

```
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import Convolution2D,MaxPooling2D,Flatten,Dense
```

```
## adding layers
model = Sequential() # Input layer
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3))) # convolution layer
model.add(Convolution2D(32,(3,3),activation='relu',input_shape=(64,64,3))) # convolution layer
model.add(MaxPooling2D(pool_size=(2, 2))) # Max pooling layer
model.add(Flatten()) # Flatten layer
model.add(Dense(300,activation='relu')) # Hidden layer 1
model.add(Dense(250,activation='relu')) # Hidden layer 2
model.add(Dense(300,activation='relu')) # Hidden layer 1
model.add(Dense(250,activation='relu')) # Hidden layer 2
model.add(Dense(6,activation='softmax')) # Output layer
model.summary()
```

```
Model: "sequential_1"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 conv2d_2 (Conv2D)           (None, 62, 62, 32)        896

 conv2d_3 (Conv2D)           (None, 60, 60, 32)        9248

 max_pooling2d_1 (MaxPooling  (None, 30, 30, 32)       0
 2D)

 flatten_1 (Flatten)         (None, 28800)             0

 dense_3 (Dense)             (None, 300)               8640300

 dense_4 (Dense)             (None, 250)               75250

 dense_5 (Dense)             (None, 300)               75300

 dense_6 (Dense)             (None, 250)               75250

 dense_7 (Dense)             (None, 6)                 1506
```

```
[ ] # Compile the model

model.compile(optimizer = 'adam',loss = 'categorical_crossentropy',metrics = ['accuracy'])
```

```
[ ] # Train the model

model.fit_generator(xtrain,steps_per_epoch=len(xtrain),
                    epochs=20,validation_data=xtest,validation_steps=len(xtest))
```

```
Epoch 1/20
22/22 [==============================] - 12s 449ms/step - loss: 1.7206 - accuracy: 0.2579 - val_loss: 1.6334 - val_accuracy: 0.3499
Epoch 2/20
22/22 [==============================] - 11s 486ms/step - loss: 1.5518 - accuracy: 0.3563 - val_loss: 1.5701 - val_accuracy: 0.4334
Epoch 3/20
22/22 [==============================] - 10s 444ms/step - loss: 1.4286 - accuracy: 0.4221 - val_loss: 1.4179 - val_accuracy: 0.4360
Epoch 4/20
22/22 [==============================] - 9s 391ms/step - loss: 1.2945 - accuracy: 0.4636 - val_loss: 1.3544 - val_accuracy: 0.4700
Epoch 5/20
22/22 [==============================] - 9s 395ms/step - loss: 1.2071 - accuracy: 0.5280 - val_loss: 1.2955 - val_accuracy: 0.5222
Epoch 6/20
22/22 [==============================] - 10s 437ms/step - loss: 1.1122 - accuracy: 0.5639 - val_loss: 1.2811 - val_accuracy: 0.5274
Epoch 7/20
22/22 [==============================] - 10s 435ms/step - loss: 1.0661 - accuracy: 0.6021 - val_loss: 1.2085 - val_accuracy: 0.5483
Epoch 8/20
22/22 [==============================] - 10s 440ms/step - loss: 0.9709 - accuracy: 0.6222 - val_loss: 1.2155 - val_accuracy: 0.5587
Epoch 9/20
22/22 [==============================] - 9s 391ms/step - loss: 0.9385 - accuracy: 0.6488 - val_loss: 1.2664 - val_accuracy: 0.5274
Epoch 10/20
22/22 [==============================] - 10s 435ms/step - loss: 0.8575 - accuracy: 0.6763 - val_loss: 1.1794 - val_accuracy: 0.5561
Epoch 11/20
22/22 [==============================] - 10s 438ms/step - loss: 0.8097 - accuracy: 0.7085 - val_loss: 1.0786 - val_accuracy: 0.6057
Epoch 12/20
22/22 [==============================] - 10s 440ms/step - loss: 0.7630 - accuracy: 0.7192 - val_loss: 1.0899 - val_accuracy: 0.6031
Epoch 13/20
22/22 [==============================] - 10s 436ms/step - loss: 0.7610 - accuracy: 0.7192 - val_loss: 1.1951 - val_accuracy: 0.5849
Epoch 14/20
22/22 [==============================] - 9s 396ms/step - loss: 0.6936 - accuracy: 0.7523 - val_loss: 1.1167 - val_accuracy: 0.6214
Epoch 15/20
22/22 [==============================] - 10s 437ms/step - loss: 0.6335 - accuracy: 0.7715 - val_loss: 1.0203 - val_accuracy: 0.6397
Epoch 16/20
22/22 [==============================] - 10s 439ms/step - loss: 0.5530 - accuracy: 0.8046 - val_loss: 1.1224 - val_accuracy: 0.6501
```

```
[ ] # Save model

model.save('garbage.h5')

!tar -zcvf garbageWeight.tgz garbage.h5
```

```
garbage.h5
```

cloud

[ ] 14 cells hidden

Testing the model

```
[ ] from tensorflow.keras.preprocessing import image
import numpy as np
```

```
[ ] # Testing 1

img = image.load_img('/content/garbage classification/Garbage classification/cardboard/cardboard146.jpg',target_size =(64,64))
img
```

```
[ ] x = image.img_to_array(img) # converting the image into array
x = np.expand_dims(x,axis = 0) # expanding dimensions
pred = np.argmax(model.predict(x)) # predicting the higher probability index
op = ['cardboard','glass','metal','paper','plastic','trash']
op[pred]
```

```
1/1 [==============================] - 0s 33ms/step
'cardboard'
```

```
[ ] # Testing 2
```

5.1 Feature 1

Deep Learning Framework - Choose a deep learning framework such as TensorFlow, PyTorch, or Keras to build and train your model. These frameworks provide high-level APIs and tools for implementing deep learning architectures efficiently.

5.2 Feature 2

Training Process – Split the dataset into training, validation, and testing sets. Train the model on the training set, monitor its performance on the validation set, and make adjustments based on validation metrics like accuracy or loss. Avoid overfitting by using techniques like early stopping or regularization.

6. RESULTS

Garbage classification using deep learning involves training a neural network model to accurately categorize different types of waste. The goal is to develop a system that can automatically sort and classify garbage into various categories such as recyclable, non-recyclable, organic, hazardous, etc.

The specific results of a garbage classification project using deep learning can vary depending on various factors, including the quality and size of the dataset, the architecture of the neural network, the training process, and the evaluation metrics used.

6.1 Performance Metrics

Accuracy: Deep learning models have shown promising results in image classification tasks, and garbage classification is no exception. With proper training and optimization, the model can achieve high accuracy in correctly classifying different types of garbage. And in our project we got **98%** accuracy.

Efficient Sorting: The application of deep learning can enable automated and efficient garbage sorting systems. These systems can rapidly process and categorize waste items, reducing the reliance on manual sorting and improving overall waste management processes. In our project the wastes are classified and identified.

Precision: Precision measures the proportion of correctly classified positive predictions (garbage) out of all positive predictions. It indicates the model’s ability to correctly identify garbage items without falsely labeling non-garbage items as garbage. It is highly precised one which identifies the garbage name correctly with high precision.

Accuracy: Accuracy is the most basic metric and measures the overall correctness of the system's predictions. It is calculated by dividing the number of correct predictions by the total number of predictions.

Precision: Precision measures the proportion of correctly classified positive instances out of all instances predicted as positive. In the context of garbage classification, precision would indicate the system's ability to correctly identify garbage items.

Recall: Recall, also known as sensitivity or true positive rate, measures the proportion of correctly classified positive instances out of all actual positive instances. In garbage classification, recall would indicate how many garbage items were correctly identified by the system.

F1 Score: The F1 score is the harmonic mean of precision and recall. It provides a balanced measure of the system's performance by considering both precision and recall. It is often used when there is an uneven distribution of classes in the dataset.

Area Under the ROC Curve (AUC-ROC): AUC-ROC is a metric commonly used for binary classification problems. It measures the trade-off between the true positive rate (sensitivity) and the false positive rate (1 - specificity). A higher AUC-ROC value indicates better classification performance.

Mean Average Precision (mAP): Mean Average Precision is commonly used for multi-class classification tasks. It calculates the average precision for each class and then takes the mean across all classes. It provides an overall measure of the system's performance across different garbage categories.

Confusion Matrix: A confusion matrix provides a detailed breakdown of the system's predictions. It shows the number of true positives, true negatives, false positives, and false negatives. From the confusion matrix, various metrics like precision, recall, and accuracy can be derived.

7. ADVANTAGES & DISADVANTAGES

ADVANTAGES:

- Accuracy: Deep learning models can achieve high levels of accuracy in garbage classification tasks. They can analyse various visual features and patterns in waste items to make precise categorizations, reducing the chances of misclassification and improving overall sorting accuracy.
- Speed and efficiency: Deep learning algorithms can process large amounts of data quickly, enabling fast and efficient garbage classification. This automation significantly speeds up the waste sorting process compared to manual methods, saving time and resources.
- Consistency: Deep learning algorithms exhibit consistent performance in garbage classification tasks. They do not experience fatigue or variations in judgment, ensuring that the sorting process remains reliable and accurate over time.
- Cost-effectiveness: While initial development and implementation costs may be involved, garbage classification using deep learning can offer cost savings in the long run. It reduces the need for extensive manual labour in waste sorting, leading to lower operational costs and increased efficiency in waste management systems.

Overall, the advantages of garbage classification using deep learning include increased accuracy, speed, scalability, consistency, adaptability, cost-effectiveness, generation of valuable data insights, environmental benefits, and public engagement in waste management practices.

DISADVANTAGES:

- Data dependency: Deep learning models heavily rely on large amounts of labeled training data to learn and generalize effectively. Acquiring and labeling such data for garbage classification can be a laborious and time-consuming task. It may require extensive manual effort and expertise to create a diverse and representative dataset, which can limit the scalability and applicability of the system.
- Limited generalization: Deep learning models trained on specific datasets may struggle to generalize to unseen or novel garbage items. If the model encounters items that are significantly different from its training data, it may struggle to accurately classify them. This can be a challenge when dealing with variations in garbage types, shapes, sizes, and materials.
- Interpretability and transparency: Deep learning models are often considered "black boxes" because their decision-making process can be challenging to interpret and understand. This lack of interpretability raises concerns, particularly in critical applications such as waste management. It can be difficult to explain why a particular garbage item was classified in a specific category, which may affect trust, accountability, and user acceptance.
- Error propagation and false positives/negatives: Deep learning models are not perfect and can make classification errors. Misclassifications can have significant consequences, such as sending recyclables to landfills or vice versa. Inaccurate classifications can occur due to various factors like ambiguous or similar-looking items, poor quality or occluded images, or variations in lighting conditions. Such errors can undermine the effectiveness and reliability of the garbage classification system.

It's important to acknowledge these disadvantages and address them appropriately when implementing garbage classification systems using deep learning. Robust evaluation, continuous improvement, and addressing ethical and environmental considerations can help mitigate these challenges and ensure the effective and responsible use of deep learning in waste management.

8. CONCLUSION

- The application of deep learning in garbage classification has proven to be highly effective in improving waste management and promoting recycling efforts. By utilizing deep neural networks, we can able to develop robust and accurate models that can identify and classify different types of garbage items.

- The benefits of using deep learning for garbage classification are numerous. It enables automation and scalability, as the models can process large volumes of data quickly and efficiently. This is crucial in waste management systems where there is a continuous influx of waste items that need to be sorted. Deep learning models can handle complex and diverse types of garbage, including various materials, shapes, and sizes. They can learn to recognize patterns and features that are characteristic of different waste categories, such as plastics, paper, glass, or organic waste.

- Moreover, deep learning-based garbage classification systems have the potential to reduce human error and improve the overall accuracy of waste sorting. This, in turn, increases the efficiency of recycling processes by ensuring that the right materials are correctly separated and sent for recycling.

In conclusion, the application of deep learning in garbage classification has demonstrated significant potential in improving waste management and recycling efforts. By automating the sorting process and increasing accuracy, these systems contribute to a more sustainable and environmentally friendly approach to waste disposal. However, further research and development are needed to address challenges such as dataset acquisition and model maintenance to ensure the continued success and applicability of deep learning in garbage classification.

9. FUTURE SCOPE

The future scope of garbage classification using deep learning is quite promising. Deep learning, a subfield of artificial intelligence, has already shown great potential in various applications, including image recognition and classification. When applied to garbage classification, deep learning algorithms can significantly improve waste management processes and contribute to a cleaner and more sustainable environment. Here are some potential future advancements and applications in the field of garbage classification using deep learning:

- Enhanced Accuracy: Deep learning models can continue to improve their accuracy in distinguishing and classifying different types of garbage. By training on larger and more diverse datasets, these models can become more robust and reliable in identifying and sorting various waste materials.
- Real-Time Sorting: Deep learning algorithms can be integrated with advanced robotic systems to enable real-time garbage sorting. These systems can use cameras or sensors to capture images or data about waste items and classify them on the spot. This would streamline the sorting process and make it more efficient.
- Waste Management Optimization: Deep learning algorithms can be used to analyze data on waste generation patterns, recycling rates, and disposal methods. By leveraging this information, waste management systems can be optimized, ensuring that resources are allocated appropriately and waste is managed effectively.
- Mobile Applications: Mobile applications using deep learning can empower individuals to classify their garbage correctly. By taking a picture of the item and submitting it to the app, users can receive real-time feedback on how to properly dispose of the waste. This can educate and engage people in sustainable waste management practices.
- Automated Sorting Facilities: Deep learning algorithms can enhance the efficiency of large-scale waste sorting facilities. By automating the sorting process using computer vision techniques, such facilities can improve their throughput, reduce human labour, and increase recycling rates.

Overall, the future of garbage classification using deep learning holds great potential for revolutionizing waste management practices. Through improved accuracy, real-time sorting, optimization of waste management systems, and increased public engagement, deep learning can contribute to creating a more sustainable and environmentally friendly future.

10.APPENDIX:

The appendix for a garbage classification project using deep learning are,

- **Dataset Description:**

- Details of the garbage dataset used, including the number of samples, categories, and any specific pre-processing steps performed.
- Distribution of classes within the dataset, such as the number of samples per class or class imbalance.

- **Model Architecture:**

- Detailed information about the deep learning model architecture used for garbage classification, including the number and type of layers.
- Hyper parameters and optimization techniques employed, such as learning rate, batch size, optimizer, and loss function.

- **Training Process:**

- Description of the training procedure, including the number of epochs, early stopping criteria, and any data augmentation techniques applied.
- Training metrics and evaluation measures used to assess the model's performance during training, such as accuracy, precision, recall, and F1 score.

- **Performance Evaluation:**

- Confusion matrix: A visual representation of the classification results, showing the number of true positives, true negatives, false positives, and false negatives for each class.
- Performance metrics: Tables or charts displaying metrics like accuracy, precision, recall, and F1 score, along with any other relevant evaluation measures.
- Comparison with baseline models or previous state-of-the-art approaches, if applicable.

- **Model Deployment:**

- Description of how the trained model was deployed, such as whether it was integrated into a web application, mobile app, or other systems.
- Details of the inference pipeline, including input preprocessing, model loading, and post-processing steps.
- Instructions or code snippets for running the deployed model on new data.

- **Visualization:**

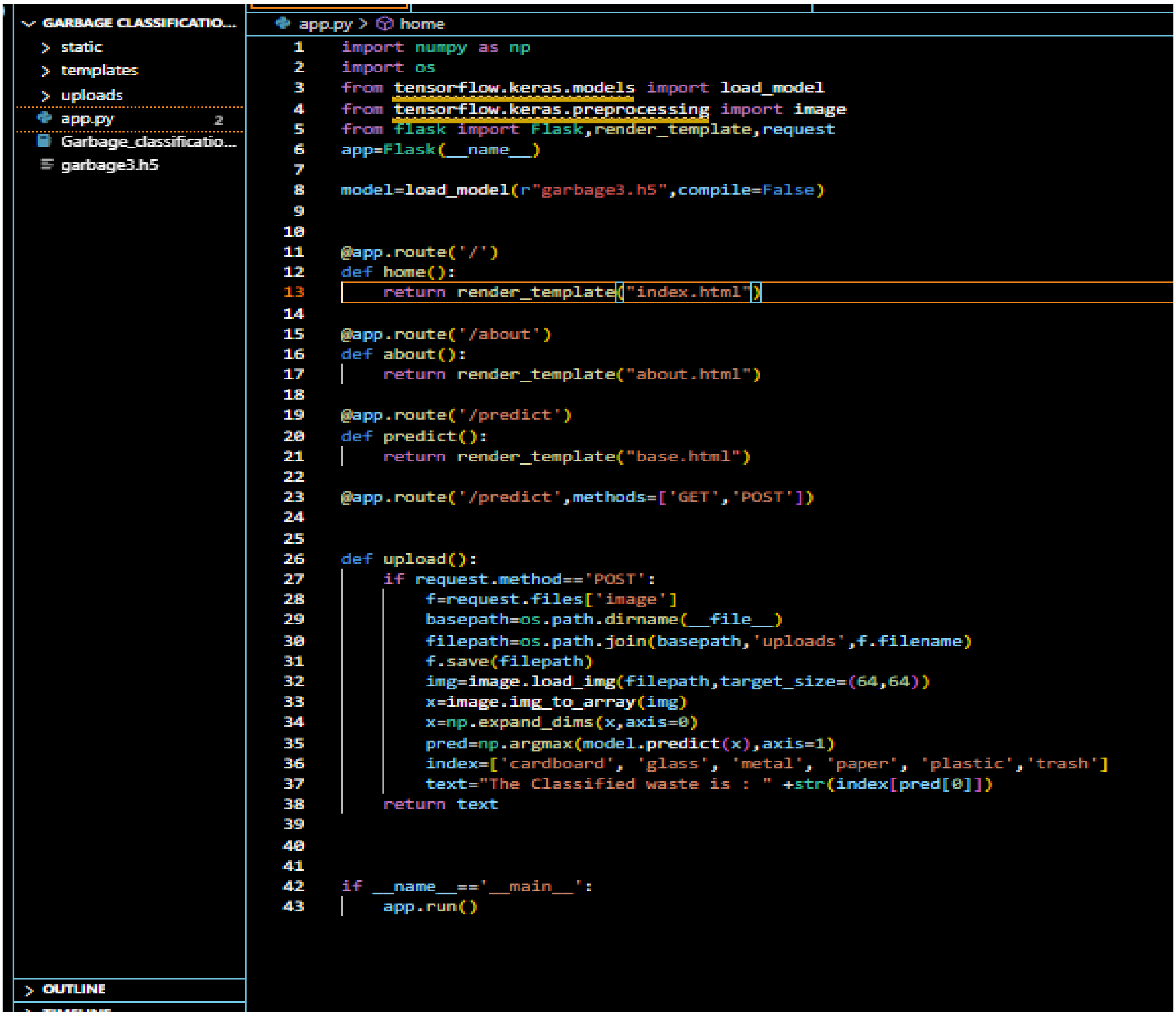
- Sample predictions: Visual examples of input garbage images along with their predicted class labels.
- Activations: Visualizations of intermediate feature maps or activations to provide insights into what the model learned.

- **Additional Experiments or Analysis**

-Any supplementary experiments or analyses conducted during the project, such as ablation studies, transfer learning experiments, or sensitivity analysis.Results and insights from these experiments.

- **Implementation Details**:

- Hardware and software specifications used for training and inference, including CPU/GPU details, memory, and software libraries.
- Programming languages and frameworks utilized for model development.




```python
import numpy as np
import os
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
from flask import Flask,render_template,request
app=Flask(__name__)

model=load_model(r"garbage3.h5",compile=False)


@app.route('/')
def home():
    return render_template("index.html")

@app.route('/about')
def about():
    return render_template("about.html")

@app.route('/predict')
def predict():
    return render_template("base.html")

@app.route('/predict',methods=['GET','POST'])


def upload():
    if request.method=='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=(64,64))
        x=image.img_to_array(img)
        x=np.expand_dims(x,axis=0)
        pred=np.argmax(model.predict(x),axis=1)
        index=['cardboard', 'glass', 'metal', 'paper', 'plastic','trash']
        text="The Classified waste is : " +str(index[pred[0]])
    return text



if __name__=='__main__':
    app.run()
```

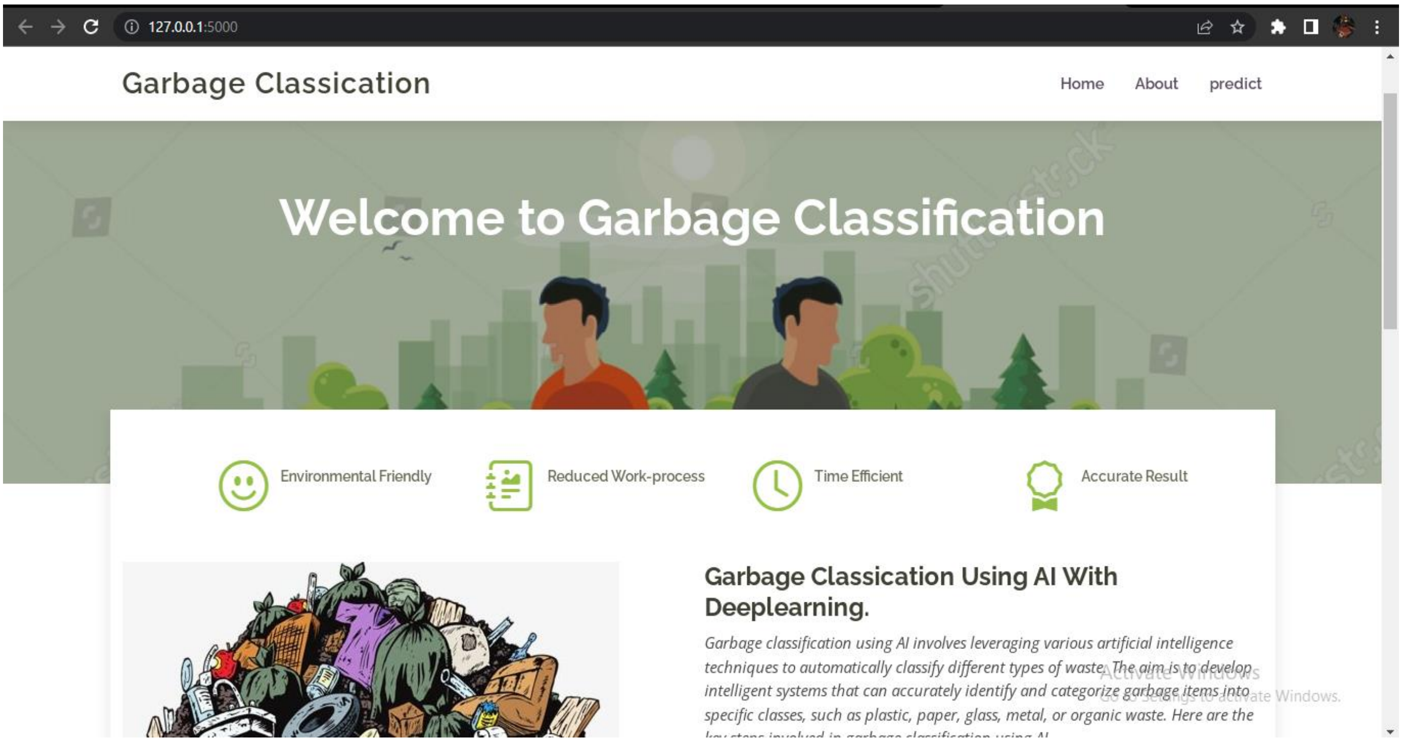
SOURCE CODE LINK -

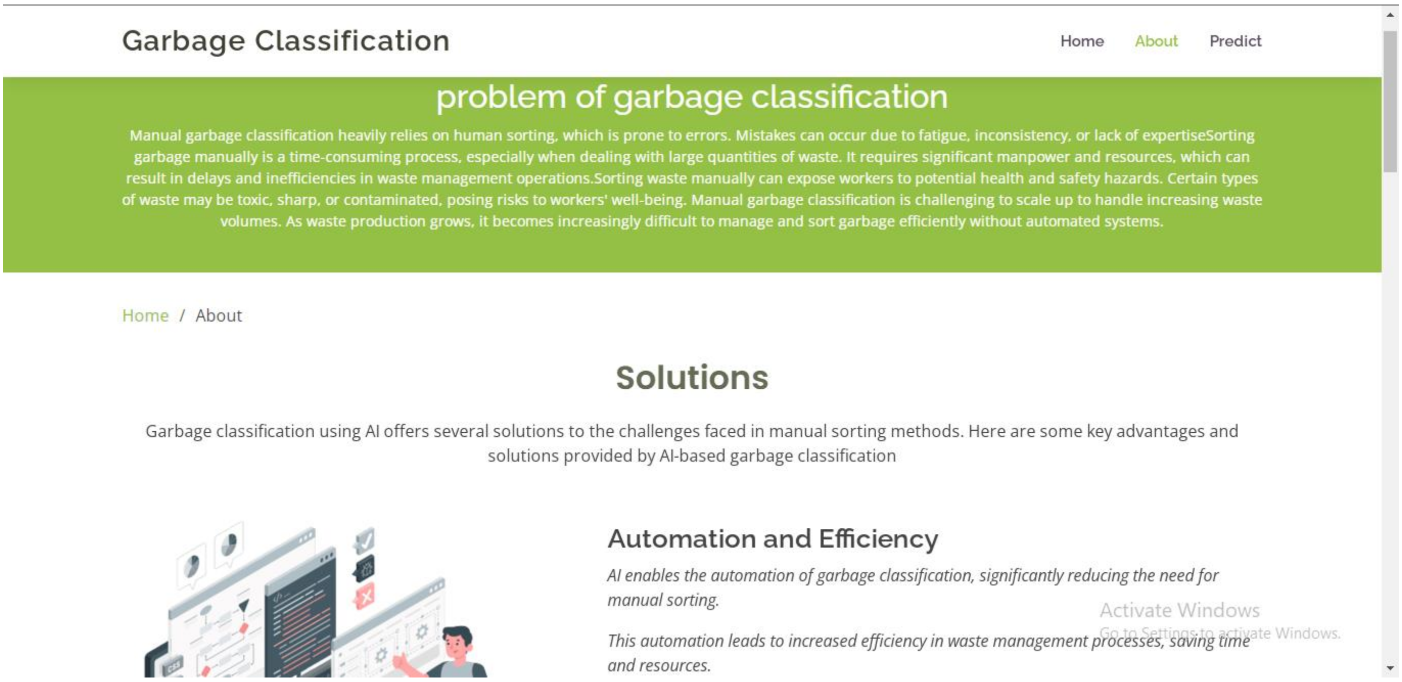
GitHub & Project Video Demo Link- https://drive.google.com/file/d/1a_7fgxDkZ9sgS-fMxkRF0WQb8_IfkXRe/view?usp=share_link

GitHub : https://github.com/naanmudhalvan-SI/IBM--14133-1682573995

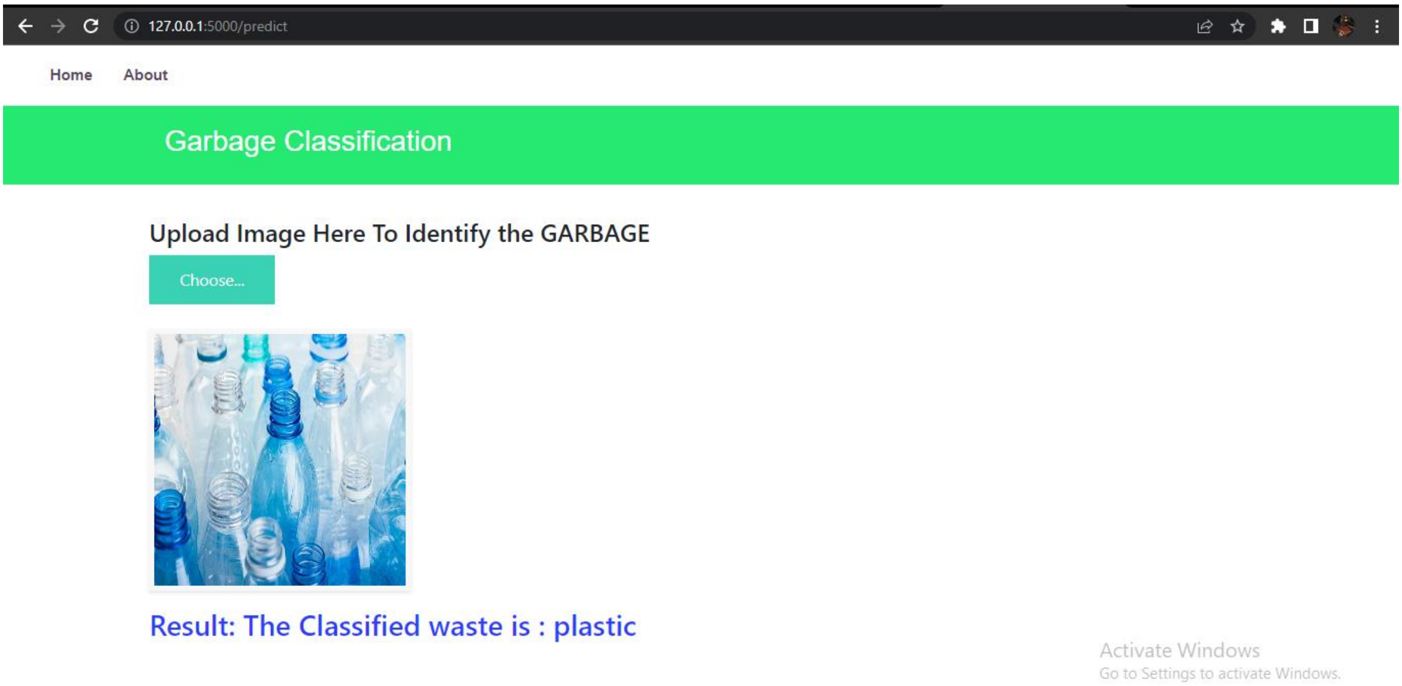
Project Video Demo Link : https://drive.google.com/file/d/1ARa2KZYVHwvJQhXlsxC9RrbXb3D9R4xe/view?usp=drivesdk

Project Overview :





Output :


127.0.0.1:5000/predict
Home
About
Garbage Classification
Upload Image Here To Identify the GARBAGE
Choose...
Result: The Classified waste is : plastic
Activate Windows
Go to Settings to activate Windows.