

# **Project Report Format**

## **1. INTRODUCTION**

### **1.1 Project Overview**

This project combines a VGG16 model with a Flask web app, allowing users to upload waste images and get instant classification results through a user-friendly UI.

### **1.2 Purpose**

The goal is to automate waste segregation, aiding smart city waste management and reducing environmental harm through proper classification.

## **2. IDEATION PHASE**

### **2.1 Problem Statement**

Manual waste segregation is time-consuming, error-prone, and inefficient. There's a need for an automated, accurate, and real-time classification solution.

### **2.2 Empathy Map Canvas**

Initially, no predefined workflows existed. The system was built from scratch, identifying pain points in manual classification to derive a structured approach.

### **2.3 Brainstorming**

Ideas centered around using image recognition models and web apps to make waste classification accessible, efficient, and adaptable for everyday use.

## **3. REQUIREMENT ANALYSIS**

### **3.1 Customer Journey map**

Users upload an image via the web app → Image is analyzed by the model → Category prediction is shown → Informs user on disposal method.

### **3.2 Solution Requirement**

Requires an accurate model (VGG16), Flask backend, HTML/CSS frontend, a reliable dataset, and integration mechanisms between UI and ML model.

### **3.3 Data Flow Diagram**

User → UI (HTML) → Flask Server → VGG16 Model → Prediction Result → Display to User (UI).

### **3.4 Technology Stack**

- **Front end:** HTML and CSS used to build user interface pages like index.html and result.html.
- **Back end:** Python with Flask handles model serving, routing, and HTTP requests for user inputs and predictions.
- **Model:** Pre-trained VGG16 CNN model used for image classification, fine-tuned on waste images from Kaggle.
- **Storage:** Model saved as vgg16.h5; image data handled during runtime with temporary storage through Flask.
- **Tools:** Anaconda, Jupyter Notebook, TensorFlow, Flask, NumPy, Pandas, Matplotlib, Seaborn, OpenCV.

## **4. PROJECT DESIGN**

### **4.1 Problem Solution Fit**

Problem of inefficient waste segregation is solved by automating classification using AI, offering speed and consistency.

#### 4.2 Proposed Solution

Build a transfer learning-based model (VGG16) deployed via Flask to predict waste categories in real-time through a web interface.

#### 4.3 Solution Architecture

UI (HTML) ↔ Flask Backend ↔ VGG16 Model ↔ Result Display; a simple client-server architecture integrated with deep learning.

### 5. PROJECT PLANNING & SCHEDULING

#### 5.1 Project Planning

Steps include data collection, preprocessing, model building, training, testing, and app deployment, completed in defined milestones.

### 6. FUNCTIONAL AND PERFORMANCE TESTING

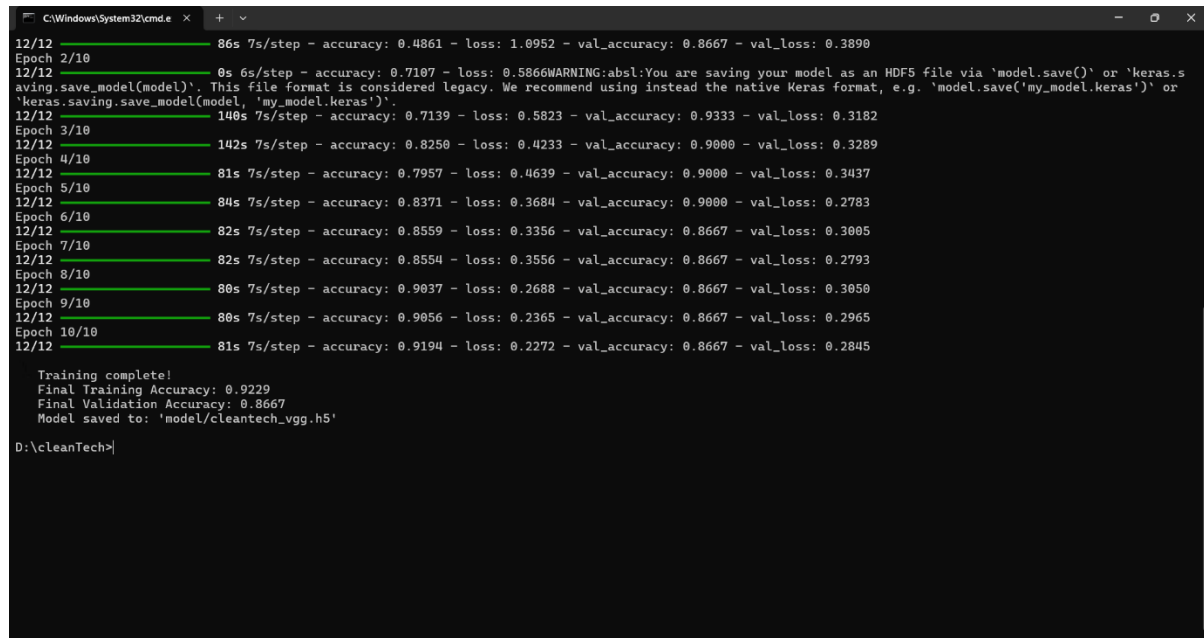
#### 6.1 Performance Testing

Model performance evaluated using accuracy, loss curves, and test data; prediction speed and model response also tested.

### 7. RESULTS

The model successfully classified waste images with high accuracy, proving transfer learning's effectiveness on small datasets.

#### 7.1 Output Screenshots



```
C:\Windows\System32\cmd.exe
12/12 ----- 86s 7s/step - accuracy: 0.4861 - loss: 1.0952 - val_accuracy: 0.8667 - val_loss: 0.3890
Epoch 2/10
12/12 ----- 0s 6s/step - accuracy: 0.7107 - loss: 0.5866WARNING:absl:You are saving your model as an HDF5 file via 'model.save()' or 'keras.s
aving.save_model(model)'. This file format is considered legacy. We recommend using instead the native Keras format, e.g. 'model.save('my_model.keras')' or
'keras.saving.save_model(model, 'my_model.keras')'.
12/12 ----- 140s 7s/step - accuracy: 0.7139 - loss: 0.5823 - val_accuracy: 0.9333 - val_loss: 0.3182
Epoch 3/10
12/12 ----- 142s 7s/step - accuracy: 0.8250 - loss: 0.4233 - val_accuracy: 0.9000 - val_loss: 0.3289
Epoch 4/10
12/12 ----- 81s 7s/step - accuracy: 0.7957 - loss: 0.4639 - val_accuracy: 0.9000 - val_loss: 0.3437
Epoch 5/10
12/12 ----- 84s 7s/step - accuracy: 0.8371 - loss: 0.3684 - val_accuracy: 0.9000 - val_loss: 0.2783
Epoch 6/10
12/12 ----- 82s 7s/step - accuracy: 0.8559 - loss: 0.3356 - val_accuracy: 0.8667 - val_loss: 0.3005
Epoch 7/10
12/12 ----- 82s 7s/step - accuracy: 0.8554 - loss: 0.3556 - val_accuracy: 0.8667 - val_loss: 0.2793
Epoch 8/10
12/12 ----- 80s 7s/step - accuracy: 0.9037 - loss: 0.2688 - val_accuracy: 0.8667 - val_loss: 0.3050
Epoch 9/10
12/12 ----- 80s 7s/step - accuracy: 0.9056 - loss: 0.2365 - val_accuracy: 0.8667 - val_loss: 0.2965
Epoch 10/10
12/12 ----- 81s 7s/step - accuracy: 0.9194 - loss: 0.2272 - val_accuracy: 0.8667 - val_loss: 0.2845

Training complete!
Final Training Accuracy: 0.9229
Final Validation Accuracy: 0.8667
Model saved to: 'model/cleantech_vgg.h5'

D:\cleanTech>
```

### 8. ADVANTAGES & DISADVANTAGES

#### Advantages:

- Fast and automated waste classification
- High prediction accuracy
- Easy web-based access

**Disadvantages:**

- Limited to 3 waste categories
- Performance may vary on poor-quality or ambiguous images

**9. CONCLUSION**

Clean Tech provides an efficient AI-based solution for automatic waste classification using transfer learning. It enhances accuracy and simplifies waste segregation.

**10. FUTURE SCOPE**

Future improvements include adding more waste categories and integrating the system with smart bins and mobile applications for broader usability.

**11. APPENDIX**

Source Code(if any)

Dataset Link

GitHub & Project Demo Link

<https://github.com/RachamaduguNarasimhaRao/ccc/blob/main/Project/templates/index.html>

<https://github.com/RachamaduguNarasimhaRao/ccc/tree/main/Project/dataset/train/Biodegradable%20Images>

<https://github.com/RachamaduguNarasimhaRao/ccc>

<https://drive.google.com/file/d/1MfLTWax2BsGROR7QzAbiLZLDgL7YS1Ub/view?usp=sharing>