1. Introduction

Project Title: CleanTech

Team Members:

- Panguluri Nagalohitha
- Paladugu Mukunda Priya
- Nagisetty Kalyani
- Manda Sri Chakrapani
- Katragadda Tejaswini

---

2. Project Overview

Purpose:
The goal of CleanTech is to classify waste into **biodegradable**, **recyclable**, or **trash** categories using a **pre-trained deep learning model (VGG16)**. The application helps automate smart waste management using computer vision.

-

Features:

-
    -

    Upload image through a web interface

    -
    -

    Predict waste type using a trained model

    -
    -

    Display result and classification confidence

    -
    -

    Flask-based backend with VGG16 integration

    -

---

3. Architecture

-

Frontend:
HTML, CSS, and JavaScript-based UI for uploading images.

-
-

Backend:
Python Flask handles image processing and routing.

-
-

Model:
VGG16 (Keras + TensorFlow) fine-tuned for image classification.

-

---

4. Setup Instructions

-

Prerequisites:

-
    -

Python 3.8+

    -
    -

Flask

    -
    -

TensorFlow

    -
    -

Keras

    -
    -

OpenCV / Matplotlib

    -

-

Installation:

-
-

bash

-
-

CopyEdit

-
-

git clone <repo-url>cd cleantech

```
pip install -r requirements.txt
python app.py
```

---

5. Folder Structure

```
cpp
CopyEdit
cleantech/
 |
 ├──── static/
 ├──── templates/
 |     └──── index.html
 ├──── app.py
 ├──── vgg16_model.h5
 └──── requirements.txt
```

---

6. Running the Application

Start the Flask server:

```
bash
```

```
CopyEdit
```

```
python app.py
```

Visit:
`http://localhost:5000`

---

7. API Documentation

POST `/predict`

Payload: Uploaded image

Response: JSON with class label (biodegradable / recyclable / trash)

---

8. Authentication

No login required.

Optionally, Flask-Login can be integrated for admin logging in future.

---

9. User Interface

Simple upload interface

-
-

Shows uploaded image

-
-

Displays predicted class and confidence

-

---

10. Testing

-

Manual testing with unseen images

-
-

Sample test set accuracy: ~91%

-
-

Future plans: Add automated unit tests with pytest

-

---

12. Known Issues

-

Image file size >5MB might slow down predictions

-
-

UI not optimized for mobile

-
-

No retry mechanism for failed uploads

-

---

13. Future Enhancements

-

Dockerize the app for deployment

-
-

Add login/auth system

-
-

Host on AWS/GCP

-
-

Add prediction logs and analytics dashboard

-