Project Documentation

□ Full Stack Development with Flask & TensorFlow

- **Project Title: :** CleanTech :Transforming Waste Management With Transfer Learning

---

1. Introduction

- **Project Title:** CleanTech :Transforming Waste Management With Transfer Learning
- **Team Leader:** Palaparthi Prasanthi
- **Team Members:**
  - Maila Naveen
  - Kethavath Jayendra Prasad Naik
  - Mandadi Keerthi

---

2. Project Overview

- **Purpose:**
  The project automates the classification of waste into **biodegradable**, **recyclable**, and **trash** using deep learning and a pre-trained VGG16 model. It aids efficient waste sorting and reduces manual effort.
- **Features:**
  - Real-time image classification
  - Simple and interactive Flask-based web UI
  - Transfer learning for high accuracy and low training time

---

3. Architecture

- **Frontend:**
  HTML pages served by Flask (`index.html`, `result.html`)
- **Backend:**
  Python Flask handles routing, file uploads, and prediction logic using `app.py`.
- **Model:**
  VGG16 pre-trained on ImageNet and fine-tuned for 3 waste categories.

---

4. Setup Instructions

- **Prerequisites:**
  - Python 3.8+
  - Flask
  - TensorFlow / Keras
  - Pillow, NumPy
- **Installation Steps:**

```
bash
CopyEdit
# Clone the project
git clone https://github.com/YOUR_USERNAME/HematoVision-CleanTech.git
cd HematoVision-CleanTech

# Install required packages
pip install -r requirements.txt

# Run the application
python app.py
```

---

5. Folder Structure

```
cpp
CopyEdit
HematoVision-CleanTech/
├── app.py
├── vgg16_model.h5
├── requirements.txt
├── templates/
│   ├── index.html
│   └── result.html
└── static/
    └── (uploaded images)
```

---

6. Running the Application

- **Frontend:** Flask serves the HTML forms
- **Backend:** Run this in the terminal:

```
bash
CopyEdit
python app.py
```

- **Access:**
  Visit `http://127.0.0.1:5000/` in the browser

---

7. API Documentation

- **Endpoint:** `/predict`
- **Method:** `POST`
- **Input:** Image (JPG/PNG)
- **Output:** Rendered HTML with predicted class (e.g., “recyclable”)

---

8. Authentication

- No authentication required in this version (can be added later)

---

9. User Interface

- Upload interface on `index.html`
- Result display with image on `result.html`

---

10. Testing

- Manual testing using test images of all 3 classes
- Verified in multiple browsers

---

11. Screenshots or Demo

    - UI Home Page
    - Prediction Result Page

---

12. Known Issues

- Large image size may delay predictions
- Limited to 3 classes in current model

---

13. Future Enhancements

- Add user authentication and image history
- Expand to more waste types
- Deploy online or turn into a mobile app