## Project Phases Template

## Project Title:

CleanTech: Transforming Waste Management with Transfer Learning

## Team ID:

LTVIP2025TMID41544

## Team Members:

- ❖ Rihan khan
- ❖ Bethamcherla Balakrishna
- ❖ Matta Renuka
- ❖ Veeravalli Sriya

# Phase-1:Brainstorming & Ideation

## Objective:

- ❖ Identify the problem statement
- ❖ Purpose and impact of the project

## Key points:

## 1.Problem statement:

Waste management is a growing concern in urban and rural areas, where the classification and disposal of waste are often inefficient, leading to environmental pollution, health hazards, and increased operational costs. Traditional waste sorting methods are manual, time-consuming, and error-prone.
There is a need for a smart, automated system that can classify different types of waste (organic, plastic, metal, paper, etc.) accurately to enable efficient recycling, composting, or disposal.

## 2.Proposed Solution:

- Use image classification to automatically detect and classify waste as recyclable or non-recyclable.
- Train a machine learning model using a labeled dataset from Kaggle containing various waste categories.
- Preprocess the dataset with resizing, normalization, and data splitting to improve model accuracy and reduce noise.
- Develop a lightweight, user-friendly HTML web interface where users can:

❖ Upload images of waste

❖ Get instant classification results(Recyclable / Non-Recyclable)

## 3.Target Users:

### 🏙️ Smart Cities & Municipal Corporations

- To implement AI-powered waste segregation in smart bins and public areas.

### 🏫 Educational Institutions

- Schools and colleges can use it to teach students about waste management and sustainability.

### 🏭 Recycling Centers & Waste Management Companies

- To speed up waste sorting on conveyor belts and improve recycling efficiency.

### 👨 👩 👧 👦 General Public / Households

- Individuals can use the web app to learn how to properly classify their daily waste.

### 🧑‍🔧 Sanitation Workers & Waste Collectors

- To reduce their manual effort and improve workplace safety by automating waste classification.

### 🌱 Environmental NGOs & Awareness Campaigns

- To promote eco-conscious behavior and educate communities using interactive tech tools**.**

## 4.Excepted Outcome:

❖ 🧠 Trained ML Model that accurately classifies waste as Recyclable or Non-Recyclable using image data.

❖ 🌐 Fully Functional Webpage where users can upload waste images and get instant classification results.

# Phase-2: Requirement Analysis:

## Objective:

❖ Define technical and functional requirements

## Key Points:

## 1.Technical Requirements:

### 💻 Programming Languages

- Python – for model training, preprocessing, and backend
- HTML– for building the front-end web interface.

### 🧠 Frameworks & Libraries

Data Handling:

- NumPy, Pandas

Image Processing:

- OpenCV

### 🧪 Model Training & Testing Tools

- **Google Colab** – for training and experimenting
- **Kaggle** – for accessing and exploring the dataset

## 🌐 To run the application:

Anaconda command prompt

## 2.Functional Requirements:

### 🖼️ Image Upload Feature

- Users should be able to upload images of waste through the web interface.

### 🤖 Waste Classification

- The system should analyze the uploaded image and classify it as:
  - ♻️ **Recyclable**
  - 🚯 **Non-Recyclable**

### 🧠 ML Model Integration

- A trained machine learning model must process the image and return accurate predictions.

### 💻 User Interface (UI)

- A clean, simple HTML-based webpage to allow:
  - Uploading images
  - Viewing results
  - Displaying status or messages

## 3.Constraints&Challenges:

### 📦 1. Dataset Limitations

- Limited variety of waste types in the Kaggle dataset
- Imbalanced data between recyclable and non-recyclable categories
- Difficult to generalize to real-world scenarios (e.g., mixed waste)

# Phase-3:Project Design:

## Objective:

❖ Create the architecture and user flow

**Key points:**

**1.System Architecture Diagram:**

[User (Web Browser)]

|

| Uploads Image

↓

[Frontend Webpage (HTML)]

|

| Sends image

↓

[Backend Server (Flask/Django)]

|

| Feeds image into

↓

[Trained ML Model]

|

| Prediction result

↓

[Backend Server]

|

| Sends result

↓

[Frontend Webpage]

|

| Displays:

↓

[♻ Recyclable / Trash/Bio degradable]

## 2.User Flow:

🧑‍💻 1. User Opens Webpage

🖼️ 2. User Uploads Waste Image

📤 3. Image Sent to Backend

🧠 4. ML Model Processes Image

📩 5. Prediction Returned

🖥️ 6. Result Displayed on Webpage

## Phase 4: Project Planning(Agile Methodologies):

## Objective:

❖ Break down the tasks using Agile methodologies

**Key Points:**

**1.Sprint Planning &Task Allocation:**

Nasiruddin Sayyad :Report making

N Dilsha Begum : Report making

N Himasree : Project Development(Frontend) ,ppt making

N Dharmesh : Project Development(Backend), ppt making

**2.Milestones & Timeline:**

- Day 1: Finalize project idea, define objectives, and assign team roles
- Day 2: Collect and download dataset from Kaggle (recyclable & non-recyclable waste)
- Day 3: Preprocess data (resize images, label, normalize, split into train/test sets)
- Day 4-5: Train the machine learning model and evaluate accuracy
- Day 6: Develop the frontend using HTML (upload feature)
- Day 7: Integrate the trained ML model with the backend
- Day 8: Test end-to-end functionality, fix bugs, and refine the UI
- Day 9: Evaluate final results, take screenshots, and prepare project output

- Day 10: Create and finalize the PowerPoint presentation & report for demonstration

## ☑ Phase-5: Integration & Development

### Objective:

Code the project and integrate components**.**

### Key Points:

1. **Technology Stack Used:**
   - Frontend: HTML
   - Backend: Python
   - AI Model: pre-trained CNN, Keras/TensorFlow
   - Libraries: NumPy, OpenCV, Pillow, TensorFlow
2. **Development Process:**
   - Dataset downloaded from Kaggle (Recyclable/Non-Recyclable waste).
   - Data preprocessing: resizing, normalization, label encoding.
   - Model trained on Google Colab using transfer learning (VGG16).
   - Flask backend created for inference API.
   - Frontend developed with upload form, prediction result display, and single-page app navigation.
   - AJAX used to connect frontend to /predict_api.
3. **Challenges & Fixes:**
   - Challenge: Class imbalance in dataset
     Fix: Used data augmentation and class weighting during training.

- Challenge: Real-time image preview was not updating
  Fix: Implemented FileReader preview in JS with error fallback.
- Challenge: Inconsistent predictions due to poor lighting in test images
  Fix: Added normalization and brightness-contrast adjustment during preprocessing.

---

## ☑ Phase-6: Functional & Performance Testing

### Objective:

Ensure the project works as expected.

### Key Points:

1. **Test Cases Executed:**
   - Uploaded various test images (plastic, food, mixed waste).
   - Verified correct category output (Biodegradable, Recyclable, Trash).
   - Tested responsiveness on mobile and desktop browsers.
   - Handled invalid file types and missing files gracefully.
2. **Bug Fixes & Improvements:**
   - Fixed image filename sanitization issues using secure_filename.
   - Added custom modals instead of default alert() for better UX.
   - Improved prediction loading and error message display.

3. **Final Validation:**
    - ☑ Meets all functional requirements listed in Phase-2.
    - ☑ End-to-end flow (image → backend → prediction → frontend) works without manual intervention.
4. **Deployment (if applicable):**
    - **Can be hosted on Render / PythonAnywhere / Heroku or local deployment using:**

nginx

CopyEdit

python app.py

http://127.0.0.1:2222

---

## ▩ Final Submission

1. **Project Report:**
   **Completed using the given template (content from cleantech.docx and this summary).**
2. **Demo Video (3–5 Minutes):**
    - **Record a walkthrough:**
        - **Uploading image**
        - **Real-time prediction**
        - **Description of technology used**
        - **Impact and applications**

3. **GitHub / Code Repository Link:**
   **Include:**
   - **app.py**
   - **index.html**
   - **static/uploads (empty or placeholder)**
   - **Model file: vgg16.h5**
   - **README.md explaining setup**