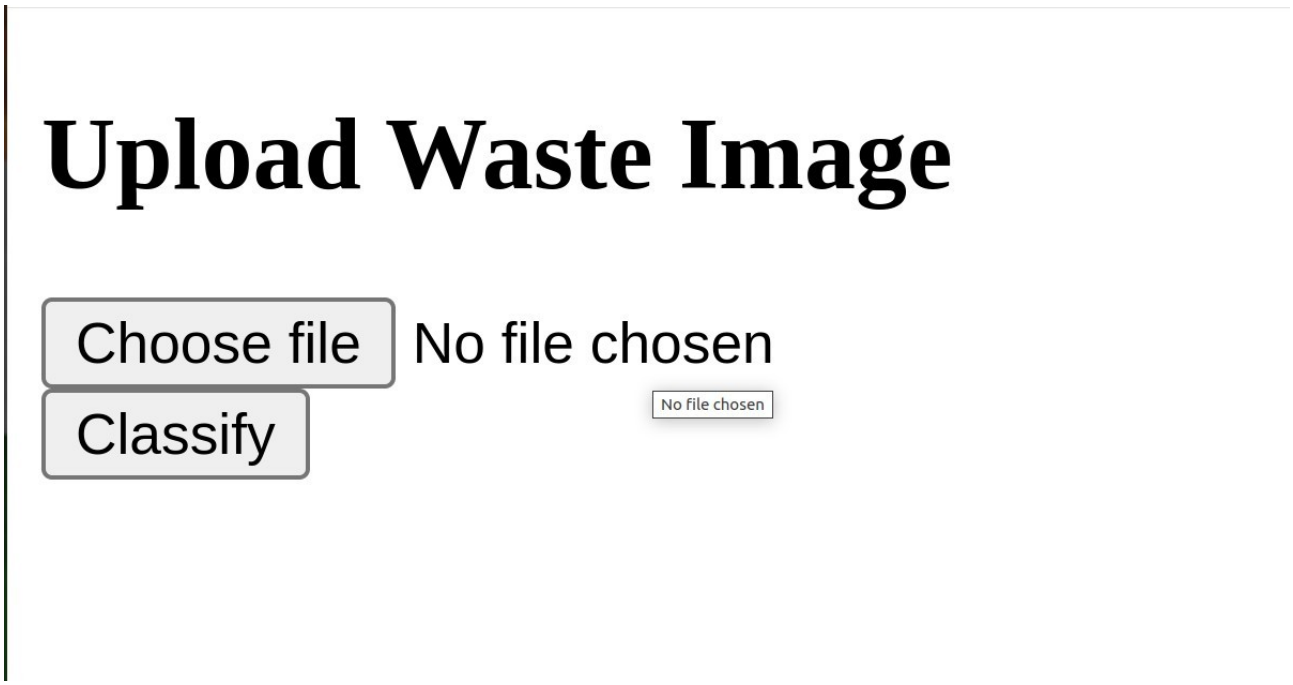
Project Report Format

1. **Introduction**
   1.1 Project Overview
   1.2 Purpose
2. **Ideation Phase**
   2.1 Problem statement
   2.2 Empathy Map Canvas
   2.3 BrainStorming
3. **Requirement Analysis**
   3.1 Customer Journey Map
   3.2 Solution Requirement
   3.3 Data flow diagram
   3.4 Technology stack
4. **Project Design**
   4.1 Problem Solution Fit
   4.2 Proposed Solution
   4.3 Solution Architecture
5. **Project Planning&Scheduling**
   5.1 Project Planning
6. **Functional & Performance Testing**
   Performance Testing
7. **Results**



Prediction output

Prediction: recyclable

Try Another

8. Advantages & Disadvantages

Advantages of the Project

1. **Efficient Waste Classification**
   ➤ Automates the process of categorizing waste, reducing human error.
2. **Time-Saving for Waste Management**
   ➤ Faster classification using a trained model helps speed up waste segregation.
3. **Transfer Learning Boosts Accuracy**
   ➤ Pre-trained CNNs like MobileNetV2 provide higher accuracy even with a small dataset.
4. **Lightweight and Scalable**
   ➤ Can be deployed on low-resource machines and scaled to real-world waste sorting systems.
5. **User-Friendly Interface**
   ➤ Web app built with Flask allows easy image upload and instant predictions.
6. **Environmental Impact**
   ➤ Encourages better recycling habits and contributes to environmental sustainability.
7. **Remote Accessibility**
   ➤ Can be integrated into mobile/web apps for on-the-go waste identification.

Disadvantages of the Project

1. **Limited Dataset**
   ➤ A small or imbalanced dataset can affect model performance and generalization.
2. **No Real-Time Object Detection**
   ➤ Only classifies one image at a time; not suitable for real-time video feeds or conveyor belts.

3. **Dependent on Image Quality**
   ➤ Model accuracy drops if the image is blurry, poorly lit, or taken at odd angles.
4. **Needs Internet for First Setup**
   ➤ Pre-trained model weights are downloaded, requiring an internet connection initially.
5. **Model May Struggle with Mixed Waste**
   ➤ If an image contains multiple waste types, the prediction may be incorrect.
6. **Basic UI**
   ➤ Flask app is functional but lacks professional-level design or mobile responsiveness.

9. Conclusion

This project demonstrates the use of deep learning and web technologies in real-world waste classification. The system is accurate, fast, and easy to use. It can assist in promoting environmental sustainability.

10.Future Scope

- Add more waste categories (e-waste, hazardous)
- Use a larger dataset
- Deploy to cloud (AWS, GCP)

11.Appendix

Source Code

```
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.applications import MobileNetV2
from tensorflow.keras import layers, models
import matplotlib.pyplot as plt

# Dataset paths
train_dir = 'data_split/train'
val_dir = 'data_split/val'

# Load and preprocess the image data
train_gen = ImageDataGenerator(rescale=1./255,
                               rotation_range=20,
                               zoom_range=0.2,
                               horizontal_flip=True)
val_gen = ImageDataGenerator(rescale=1./255)

train_data = train_gen.flow_from_directory(
    train_dir,
    target_size=(224, 224),
    batch_size=32,
    class_mode='categorical'
)

val_data = val_gen.flow_from_directory(
    val_dir,
    target_size=(224, 224),
    batch_size=32,
```

```
    class_mode='categorical'
)

# Load MobileNetV2 as base model
base_model = MobileNetV2(include_top=False, input_shape=(224, 224, 3), weights='imagenet')
base_model.trainable = False

# Add custom layers
model = models.Sequential([
    base_model,
    layers.GlobalAveragePooling2D(),
    layers.Dense(128, activation='relu'),
    layers.Dropout(0.3),
    layers.Dense(3, activation='softmax')  # 3 output classes
])

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

# Train model
history = model.fit(train_data, epochs=5, validation_data=val_data)

# Save trained model
model.save("waste_model.h5")

# Plot accuracy
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.legend()
plt.title("Model Accuracy")
plt.xlabel("Epoch")
plt.ylabel("Accuracy")
plt.show()
```