CODE FOR TRANSFORMING WASTE MANAGEMENT WITH TRANSFER LEARNING

```
from flask import Flask, render_template, request, url_for
from tensorflow.keras.models import load_model
from tensorflow.keras.preprocessing import image
import numpy as np
import os

app = Flask(_name_)

model = load_model('model/transfer_model.h5')

class_labels = ['Plastic', 'Organic', 'Metal', 'E-Waste', 'Glass']

UPLOAD_FOLDER = 'static/uploads'
os.makedirs(UPLOAD_FOLDER, exist_ok=True)
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER

detailed_suggestions = {
    'Plastic': {
        'bottle': '♻️ This plastic bottle is recyclable. Reuse for watering plants or DIY organizers.',
        'bag': '🛍️ Plastic bags can be reused or recycled. Avoid thin single-use plastics.',
        'container': '🪣 Clean and reuse plastic containers or recycle at drop-off points.',
        'default': '♻️ Recycle plastics at collection points. Avoid single-use plastic.'
    },
    'Glass': {
        'bottle': '🍾 Reuse glass bottles as décor or recycle at glass facilities.',
        'jar': '🫙 Glass jars are reusable for storage. Recycle if broken.',
        'default': '♻️ Recycle clean glass in glass bins. Avoid mixing with regular trash.'
    },
```

```
    'Metal': {
        'can': ' 🥫 Aluminum cans can be flattened and sold to scrap dealers.',
        'foil': ' 🧻 Clean aluminum foil can be recycled. Avoid food-contaminated foil.',
        'utensil': '🍽 Metal utensils can be reused or recycled as scrap.',
        'default': ' 🧲 Metals are 100% recyclable. Deposit in local scrap centers.'
    },
    'E-Waste': {
        'mobile': ' 📱 Old phones can be exchanged or dropped at e-waste centers.',
        'laptop': ' 💻 Recycle laptops via certified e-waste recyclers.',
        'cable': ' 🔌 Electronic cables should go to e-waste bins.',
        'default': '⚠ Never mix e-waste with regular trash. Use certified recyclers.'
    },
    'Organic': {
        'food': ' 🥗 Food waste is compostable. Use it to enrich soil.',
        'vegetable': ' 🥦 Vegetable peels go in compost bins.',
        'fruit': ' 🍌 Fruit peels are great for composting.',
        'default': ' 🌿 Compost organic waste like food scraps to reduce landfill impact.'
    },
    'Default': {
        'default': '🗑 Make sure to segregate waste properly and follow local disposal guidelines.'
    }
}

keyword_map = {
    'bottle': 'Glass',
    'plastic': 'Plastic',
    'metal': 'Metal',
    'can': 'Metal',
    'foil': 'Metal',
    'mobile': 'E-Waste',
```

```
    'phone': 'E-Waste',
    'jar': 'Glass',
    'food': 'Organic',
    'fruit': 'Organic',
    'vegetable': 'Organic'
}

@app.route('/')
def home():
    return render_template('index.html', prediction=None)

@app.route('/predict', methods=['POST'])
def predict():
    file = request.files['file']
    if file.filename == '':
        return render_template('index.html', prediction="No file selected")

    filepath = os.path.join(app.config['UPLOAD_FOLDER'], file.filename)
    file.save(filepath)

    # Preprocess the image
    img = image.load_img(filepath, target_size=(224, 224))
    img_array = image.img_to_array(img)
    img_array = np.expand_dims(img_array, axis=0)
    img_array /= 255.0

    prediction = model.predict(img_array)
    predicted_class = np.argmax(prediction)
    confidence = np.max(prediction)
    label = class_labels[predicted_class] if predicted_class < len(class_labels) else "Unknown"
```

```
    # Fallback using filename hints
    filename_keywords = file.filename.lower().split(".")[0].split("_")
    for word in filename_keywords:
        if word in keyword_map:
            label = keyword_map[word]
            break

    # Suggestion matching
    matched_keyword = "default"
    for word in filename_keywords:
        if word in detailed_suggestions.get(label, {}):
            matched_keyword = word
            break

    suggestion = detailed_suggestions.get(label, {}).get(
        matched_keyword,
        detailed_suggestions.get(label, {}).get("default", "No suggestion available.")
    )

    return render_template(
        "index.html",
        prediction=label,
        image_path=url_for('static', filename='uploads/' + file.filename),
        suggestion=suggestion
    )

if _name_ == '_main_':
    app.run(debug=True)
```