Name- Devansh Sharma  
Roll. no - 2215500058

University - GLA University

Mail- devansh.sharma\_cs.aiml22@gla.ac.in

Task - 1 -  Remove Background.  
  
pip install rembg

pip install onnxruntime

**Rembeg -** A Python tool to remove image backgrounds automatically.Uses deep learning models (like U²-Net) for background removal.After installation, you can use it via **CLI** or **Python code**.

**Onnxruntime -** A runtime for executing ONNX (Open Neural Network Exchange) models.rembg depends on onnxruntime for running its neural network models efficiently.

from rembg import remove

from PIL import Image

input\_path = "person.jpeg"

output\_path = "person\_no\_bg.png"

input\_image = Image.open(input\_path)

output\_image = remove(input\_image)

output\_image.save(output\_path)

In this step we are removing background from the image and identifying the person.

  


Task – 2 – Choose a scenery and add this image to scenery

import cv2

import numpy as np

background = cv2.imread("scenery.jpg")

person = cv2.imread("person\_no\_bg.png", cv2.IMREAD\_UNCHANGED)

bh, bw = background.shape[:2]

scale\_percent = int(input("Enter the % of background height the person should cover: "))

target\_height = int(bh \* (scale\_percent / 100.0))

scale\_factor = target\_height / person.shape[0]

person = cv2.resize(person, (int(person.shape[1] \* scale\_factor), target\_height))

ph, pw = person.shape[:2]

x\_offset = (bw - pw) // 2

y\_offset = bh - ph - 10

if person.shape[2] == 4:

alpha = person[:, :, 3] / 255.0

alpha\_inv = 1 - alpha

person\_rgb = person[:, :, :3]

else:

alpha = np.ones((ph, pw))

alpha\_inv = 1 - alpha

person\_rgb = person

roi = background[y\_offset:y\_offset+ph, x\_offset:x\_offset+pw]

for c in range(3):

roi[:, :, c] = alpha \* person\_rgb[:, :, c] + alpha\_inv \* roi[:, :, c]

background[y\_offset:y\_offset+ph, x\_offset:x\_offset+pw] = roi

cv2.imwrite("final\_output.png", background)

print("Saved final\_output.png").



Task - 3 -  Add shadow to the final output.

!pip install --quiet mediapipe opencv-python-headless pillow numpy

mediapipe- A Google framework for building computer vision pipelines (face detection, hand tracking, pose estimation, etc.).

opencv - python-headless- OpenCV library for image processing and computer vision, but without GUI dependencies (useful in servers and Colab).

pillow (PIL) - A Python imaging library for opening, editing, and saving image files.

numpy - A core library for numerical computations in Python (used for handling arrays and matrices).

import cv2

import mediapipe as mp

import numpy as np

from PIL import Image, ImageFilter, ImageEnhance

import os

def add\_shadow\_to\_person(

image\_path,

output\_path="image\_with\_shadow.png",

angle\_deg=30

shadow\_distance=40,

shadow\_opacity=100,

blur\_radius=15

img\_bgr = cv2.imread(image\_path)

if img\_bgr is None:

raise FileNotFoundError(f"Could not load image: {image\_path}")

h, w = img\_bgr.shape[:2]

mp\_selfie = mp.solutions.selfie\_segmentation.SelfieSegmentation(model\_selection=1)

img\_rgb = cv2.cvtColor(img\_bgr, cv2.COLOR\_BGR2RGB)

result = mp\_selfie.process(img\_rgb)

person\_mask = (result.segmentation\_mask > 0.5).astype(np.uint8) \* 255

img\_rgba = cv2.cvtColor(img\_bgr, cv2.COLOR\_BGR2BGRA)

img\_rgba[:, :, 3] = person\_mask # alpha channel = person mask

person\_pil = Image.fromarray(cv2.cvtColor(img\_rgba, cv2.COLOR\_BGRA2RGBA))

alpha = person\_pil.split()[3]

shadow = Image.new("RGBA", person\_pil.size, (0, 0, 0, 0))

shadow.paste((0, 0, 0, shadow\_opacity), mask=alpha)

shadow = shadow.filter(ImageFilter.GaussianBlur(blur\_radius))

dx = int(np.cos(np.radians(angle\_deg)) \* shadow\_distance)

dy = int(np.sin(np.radians(angle\_deg)) \* shadow\_distance)

shadow = shadow.transform(

person\_pil.size, Image.AFFINE, (1, 0, dx, 0, 1, dy), resample=Image.BICUBIC

)

base\_img = Image.open(image\_path).convert("RGBA")

final\_img = Image.alpha\_composite(base\_img, shadow)

final\_img = Image.alpha\_composite(final\_img, person\_pil)

final\_img.save(output\_path)

print(f"Saved: {output\_path}")

return output\_path

This is the code for shadow generation now we need to call a function.

add\_shadow\_to\_person(

input\_image,

output\_image,

angle\_deg=200, # 200° = left-down direction

shadow\_distance=35, # distance of shadow

shadow\_opacity=150, # darkness

blur\_radius=1 # softness

)



**SHADOW**