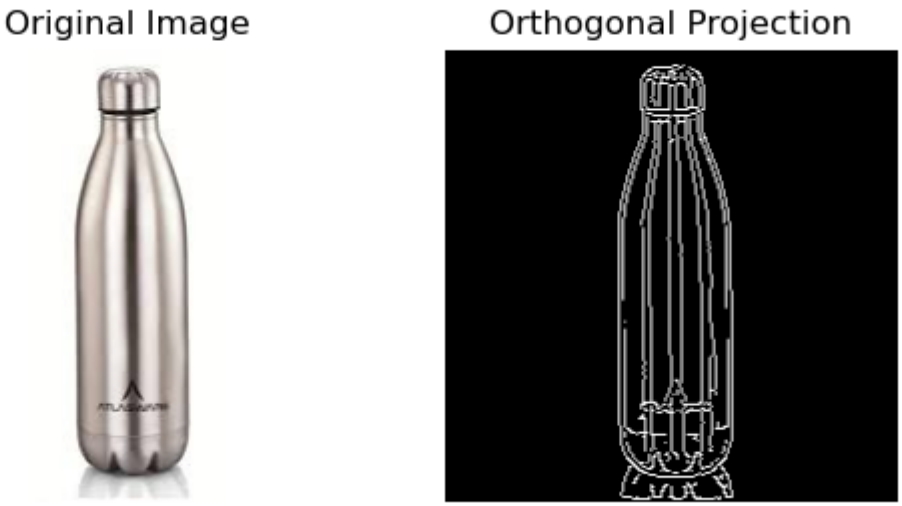
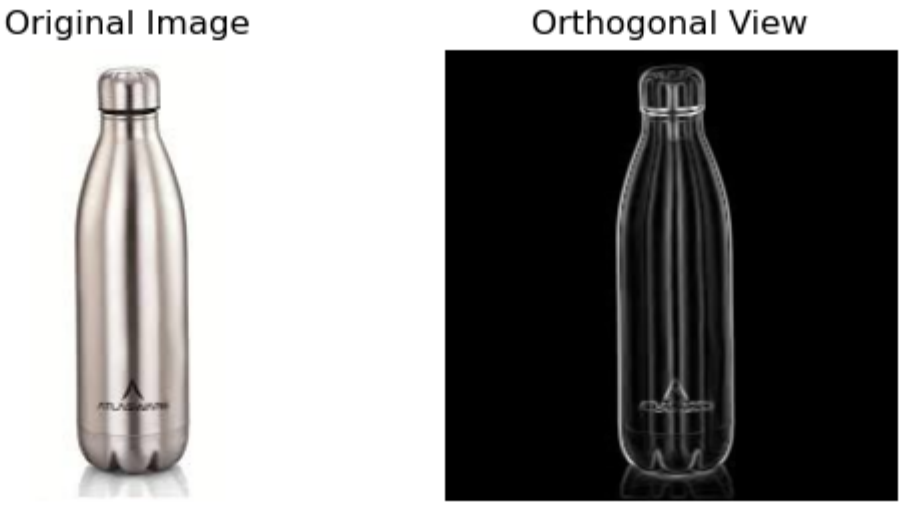
In [7]:

```
import cv2
import numpy as np
import matplotlib.pyplot as plt

# Function to get the image from the user
def get_image():
    image_path = input("Enter the path to the image: ")
    image = cv2.imread(image_path)
    return image

# Function to visualize the orthogonal projection of the image
def visualize_orthogonal_projection(image):
    # Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Apply the Canny edge detection algorithm
    edges = cv2.Canny(gray_image, 50, 150)

    # Display the image and its orthogonal projection
    plt.subplot(1, 2, 1)
    plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
    plt.title("Original Image")
    plt.axis("off")

    plt.subplot(1, 2, 2)
    plt.imshow(edges, cmap="gray")
    plt.title("Orthogonal Projection")
    plt.axis("off")

    plt.show()

# Function to visualize the orthogonal view of the image
def visualize_orthogonal_view(image):
    # Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Apply the Sobel operator to detect edges
    sobel_x = cv2.Sobel(gray_image, cv2.CV_64F, 1, 0, ksize=3)
    sobel_y = cv2.Sobel(gray_image, cv2.CV_64F, 0, 1, ksize=3)

    # Compute the magnitude of the gradient
    gradient_magnitude = np.sqrt(sobel_x**2 + sobel_y**2)

    # Display the image and its orthogonal view
    plt.subplot(1, 2, 1)
    plt.imshow(cv2.cvtColor(image, cv2.COLOR_BGR2RGB))
    plt.title("Original Image")
    plt.axis("off")

    plt.subplot(1, 2, 2)
    plt.imshow(gradient_magnitude, cmap="gray")
    plt.title("Orthogonal View")
    plt.axis("off")

    plt.show()

# Main function
def main():
    # Get the image from the user
```

```
    image = get_image()

    # Visualize the orthogonal projection of the image
    visualize_orthogonal_projection(image)

    # Visualize the orthogonal view of the image
    visualize_orthogonal_view(image)

# Run the main function
if __name__ == "__main__":
    main()
```

```
Enter the path to the image: C:\Users\asus\Desktop\download.jpg
```

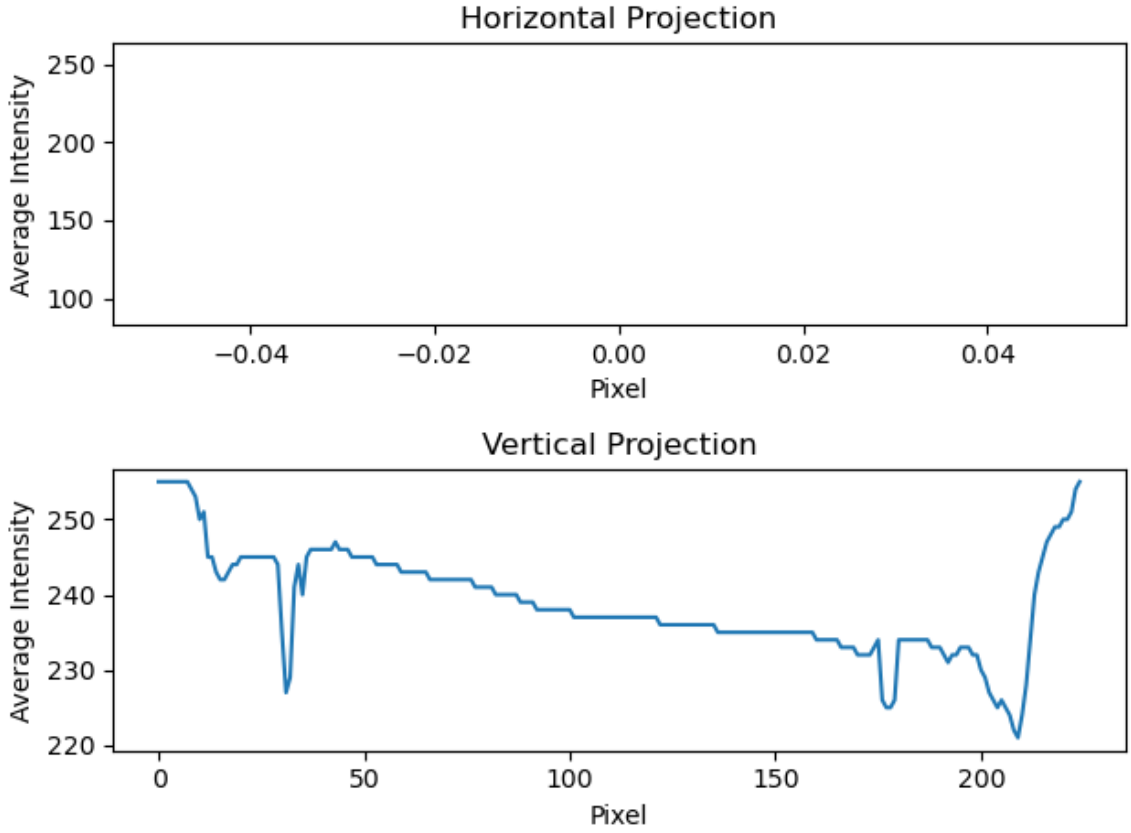


In [8]:

```
import cv2
import matplotlib.pyplot as plt

# Function to visualize the projections of an image
def visualize_projections(image_path):
    # Read the image
    image = cv2.imread(image_path)

    # Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Perform the projections
    horizontal_projection = cv2.reduce(gray_image, 0, cv2.REDUCE_AVG)
    vertical_projection = cv2.reduce(gray_image, 1, cv2.REDUCE_AVG)

    # Plot the projections
    plt.subplot(2, 1, 1)
    plt.plot(horizontal_projection)
    plt.title('Horizontal Projection')
    plt.xlabel('Pixel')
    plt.ylabel('Average Intensity')

    plt.subplot(2, 1, 2)
    plt.plot(vertical_projection)
    plt.title('Vertical Projection')
    plt.xlabel('Pixel')
    plt.ylabel('Average Intensity')

    # Display the image and projections
    plt.tight_layout()
    plt.show()

# Get the image path from the user
image_path = input('Enter the path to the image: ')

# Visualize the projections
visualize_projections(image_path)
```

```
Enter the path to the image: C:\Users\asus\Desktop\download.jpg
```


Horizontal Projection
Average Intensity
250
200
150
100
−0.04
−0.02
0.00
0.02
0.04
Pixel
Vertical Projection
Average Intensity
250
240
230
220
0
50
100
150
200
Pixel

In [6]:

```
import cv2
import matplotlib.pyplot as plt

# Function to visualize the projections of an image
def visualize_projections(image_path):
    # Read the image
    image = cv2.imread(image_path)

    # Convert the image to grayscale
    gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

    # Perform different projections
    projections = []
    projections.append(cv2.resize(image, (300, 300)))  # Original image
    projections.append(cv2.resize(cv2.flip(image, 0), (300, 300)))  # Vertical flip
    projections.append(cv2.resize(cv2.flip(image, 1), (300, 300)))  # Horizontal flip
    projections.append(cv2.resize(cv2.transpose(image), (300, 300)))  # Transpose

    # Display the projections
    fig, axs = plt.subplots(2, 2)
    axs[0, 0].imshow(cv2.cvtColor(projections[0], cv2.COLOR_BGR2RGB))
    axs[0, 0].set_title("Original Image")
    axs[0, 0].axis("off")
    axs[0, 1].imshow(cv2.cvtColor(projections[1], cv2.COLOR_BGR2RGB))
    axs[0, 1].set_title("Vertical Flip")
    axs[0, 1].axis("off")
    axs[1, 0].imshow(cv2.cvtColor(projections[2], cv2.COLOR_BGR2RGB))
    axs[1, 0].set_title("Horizontal Flip")
    axs[1, 0].axis("off")
    axs[1, 1].imshow(cv2.cvtColor(projections[3], cv2.COLOR_BGR2RGB))
    axs[1, 1].set_title("Transpose")
    axs[1, 1].axis("off")

    # Adjust the layout and display the plot
    plt.tight_layout()
    plt.show()

# Get the image path from the user
image_path = input("Enter the path to the image: ")

# Visualize the projections of the image
visualize_projections(image_path)
```

```
Enter the path to the image: C:\Users\asus\Desktop\download.jpg
```

Original Image

Vertical Flip

Horizontal Flip

Transpose