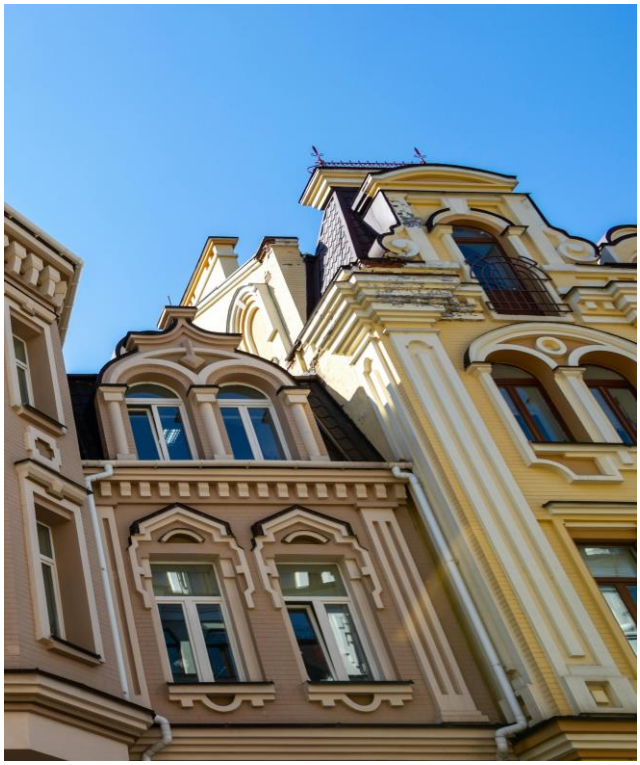
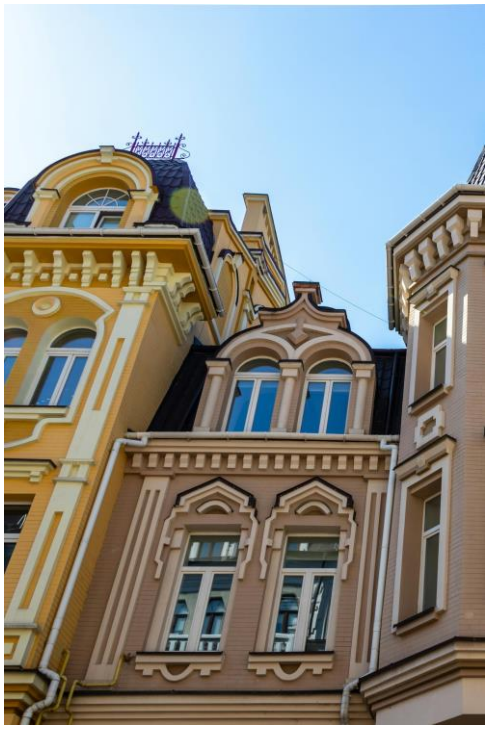
Nguyễn Sỹ Hà - 2151013018

Img1 :

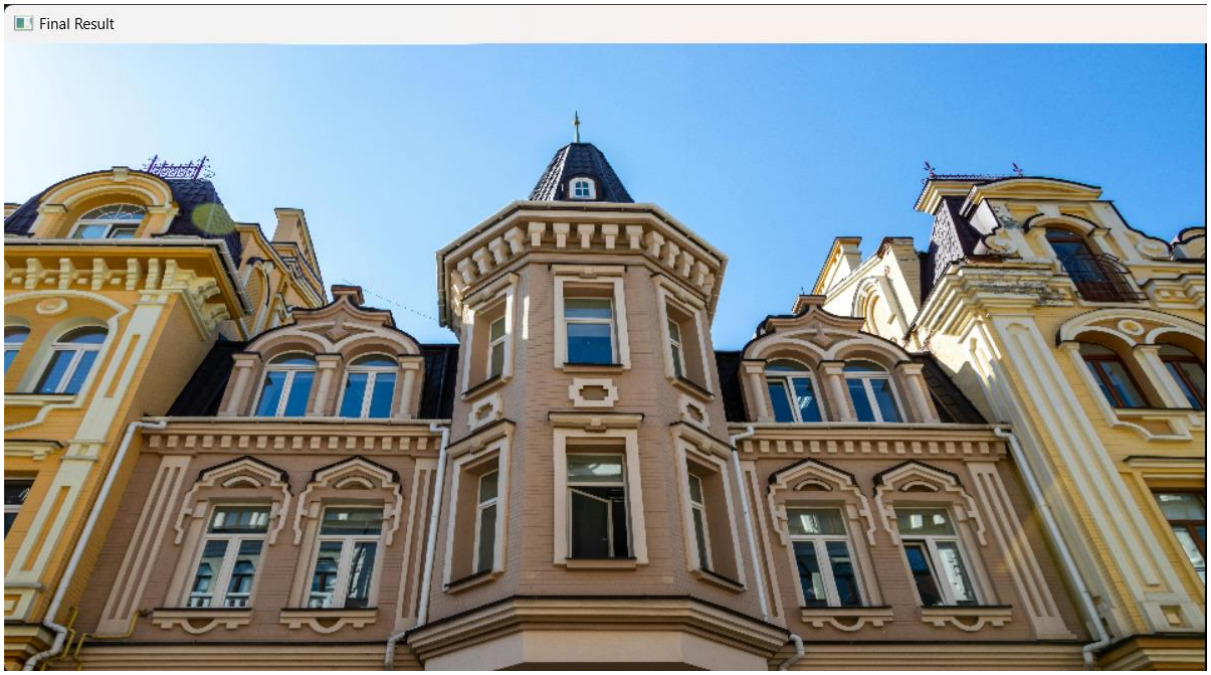


Img 2

Img3



Result



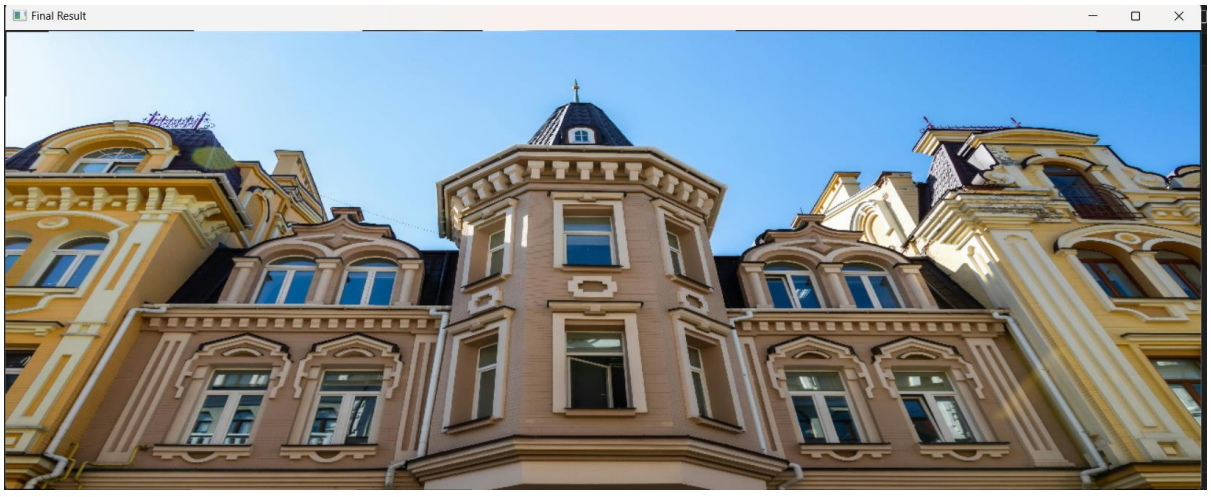
SRC les01

```
import cv2
import numpy as np
def stitch_images(image1, image2, image3):
    # Detect keypoints and compute descriptors for each image
```

```
    sift = cv2.SIFT_create()
    kp1, des1 = sift.detectAndCompute(image1, None)
    kp2, des2 = sift.detectAndCompute(image2, None)
    kp3, des3 = sift.detectAndCompute(image3, None)

    # Match keypoints between adjacent images
    bf = cv2.BFMatcher()
    matches1_2 = bf.knnMatch(des1, des2, k=2)
    matches2_3 = bf.knnMatch(des2, des3, k=2)

    # Apply ratio test to select good matches
    good_matches1_2 = []
    for m, n in matches1_2:
        if m.distance < 0.75 * n.distance:
            good_matches1_2.append(m)

    good_matches2_3 = []
    for m, n in matches2_3:
        if m.distance < 0.75 * n.distance:
            good_matches2_3.append(m)

    # Find homography matrices
    src_pts1 = np.float32([kp1[m.queryIdx].pt for m in
good_matches1_2]).reshape(-1, 1, 2)
    dst_pts1 = np.float32([kp2[m.trainIdx].pt for m in
good_matches1_2]).reshape(-1, 1, 2)
    H1, _ = cv2.findHomography(src_pts1, dst_pts1, cv2.RANSAC, 5.0)

    src_pts2 = np.float32([kp2[m.queryIdx].pt for m in
good_matches2_3]).reshape(-1, 1, 2)
    dst_pts2 = np.float32([kp3[m.trainIdx].pt for m in
good_matches2_3]).reshape(-1, 1, 2)
    H2, _ = cv2.findHomography(src_pts2, dst_pts2, cv2.RANSAC, 5.0)

    # Warp images using homography matrices
    result1 = cv2.warpPerspective(image1, H1, (image2.shape[1] +
image1.shape[1], image2.shape[0]))
    result1[0:image2.shape[0], 0:image2.shape[1]] = image2

    result2 = cv2.warpPerspective(result1, H2, (image3.shape[1] +
result1.shape[1], image3.shape[0]))
    result2[0:image3.shape[0], 0:image3.shape[1]] = image3

    return result2

# Load the three input images
image1 = cv2.imread("image1.jpg")
```

```
image2 = cv2.imread("image2.jpg")
image3 = cv2.imread("image3.jpg")

# Stitch the images together
stitched_image = stitch_images(image1, image2, image3)

# Save the stitched image
cv2.imwrite("stitched_image.jpg", stitched_image)

# Print a success message
print("Images stitched successfully and saved as stitched_image.jpg")

# Display the stitched image

# Hiển thị ảnh với kích thước mới
cv2.namedWindow('Final Result', cv2.WINDOW_NORMAL)  # Cho phép điều chỉnh kích
thước cửa sổ
cv2.resizeWindow('Final Result', 1300,500)  # Đặt kích thước cửa sổ
cv2.imshow('Final Result', stitched_image)


cv2.waitKey(0)
cv2.destroyAllWindows()
```

Les02



Src les02:

```
import cv2

# Load images
image1 = cv2.imread('image1.jpg')
image2 = cv2.imread('image2.jpg')
```

```python
image3 = cv2.imread('image3.jpg')

# Create a Stitcher object
stitcher = cv2.Stitcher_create()

# Stitch the images
status, stitched_image = stitcher.stitch((image1, image2, image3))

# Check if stitching was successful
if status == cv2.Stitcher_OK:
    # Display the stitched image
    cv2.namedWindow('Final Result', cv2.WINDOW_NORMAL)  # Cho phép điều chỉnh kích thước cửa sổ
    cv2.resizeWindow('Final Result', 1300,500)  # Đặt kích thước cửa sổ
    cv2.imshow('Final Result', stitched_image)
    cv2.waitKey(0)
    cv2.destroyAllWindows()

else:
    print("Stitching failed!")
```