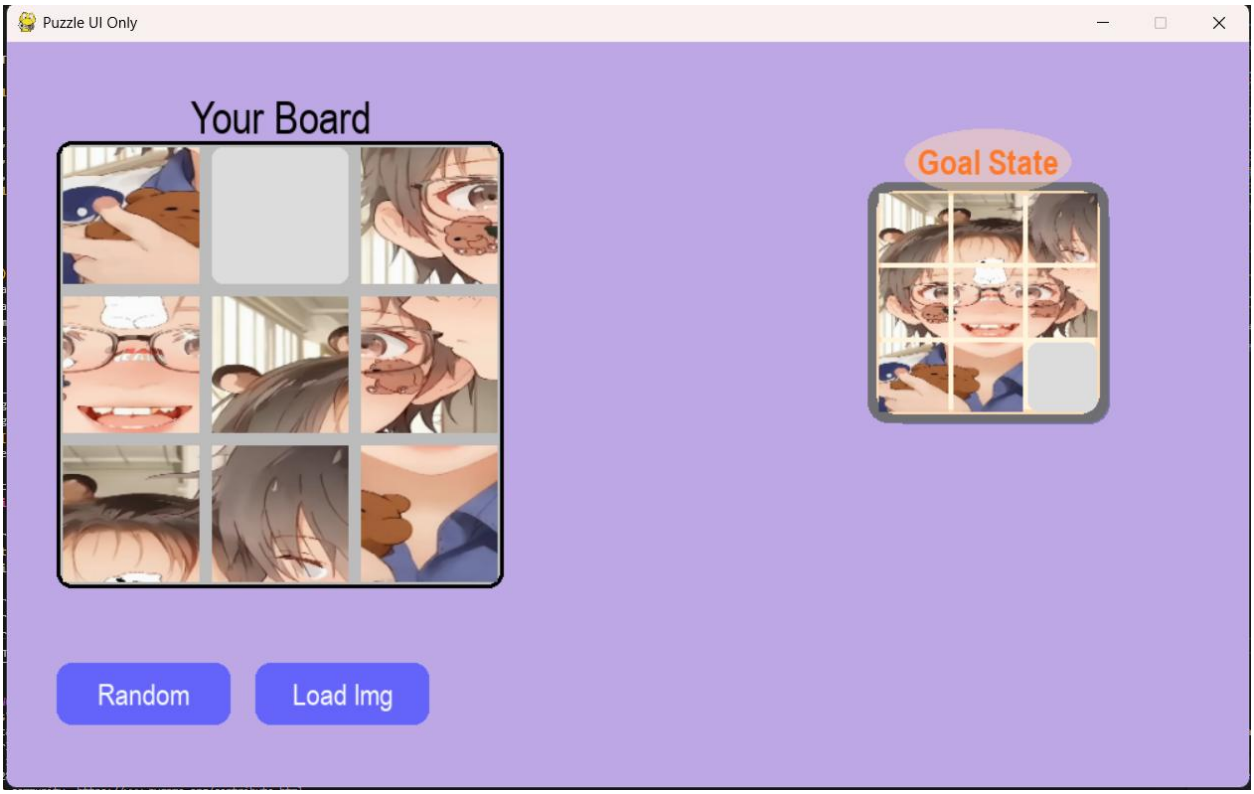
```
import pygame
import random
from tkinter import Tk, filedialog
import os

WIDTH, HEIGHT = 1000, 600
ROWS, COLS = 3, 3
CELL_SIZE = 120
GAP = 5
LEFT_BOARD_X, LEFT_BOARD_Y = 40, 80
LEFT_BOARD_W, LEFT_BOARD_H = CELL_SIZE * COLS, CELL_SIZE * ROWS
GOAL_BOARD_X, GOAL_BOARD_Y = 700, 120  # Di chuyển sang phải, cách xa bảng chính
GOAL_BOARD_W, GOAL_BOARD_H = CELL_SIZE * COLS // 2, CELL_SIZE * ROWS // 2
BG_COLOR = (190, 167, 229)
BOARD_GREY = (188, 188, 188)
BUTTON_COLOR = (100, 100, 250)
TEXT_COLOR = (255, 255, 255)

pygame.init()
screen = pygame.display.set_mode((WIDTH, HEIGHT))
pygame.display.set_caption("Puzzle UI Only")
clock = pygame.time.Clock()
font = pygame.font.SysFont("Arial", 36)
small_font = pygame.font.SysFont("Arial", 24)

def load_and_slice(path, cell_size=CELL_SIZE):
    img = pygame.image.load(path).convert_alpha()
    img = pygame.transform.scale(img, (cell_size*COLS, cell_size*ROWS))
    tiles = []
    w, h = cell_size, cell_size
    for r in range(ROWS):
        for c in range(COLS):
            if r == ROWS-1 and c == COLS-1:
                continue
            rect = pygame.Rect(c*w, r*h, w, h)
            tiles.append(img.subsurface(rect).copy())
    return tiles

def draw_board(screen, board, x, y, label, font, tiles=None, cell_size=CELL_SIZE,
gap=GAP, w=LEFT_BOARD_W, h=LEFT_BOARD_H):
    pygame.draw.rect(screen, BOARD_GREY, (x, y, w, h), border_radius=12)
    pygame.draw.rect(screen, (0,0,0), (x, y, w, h), 3, border_radius=12)
    for i, num in enumerate(board):
        r, c = divmod(i, COLS)
        rect = pygame.Rect(
```

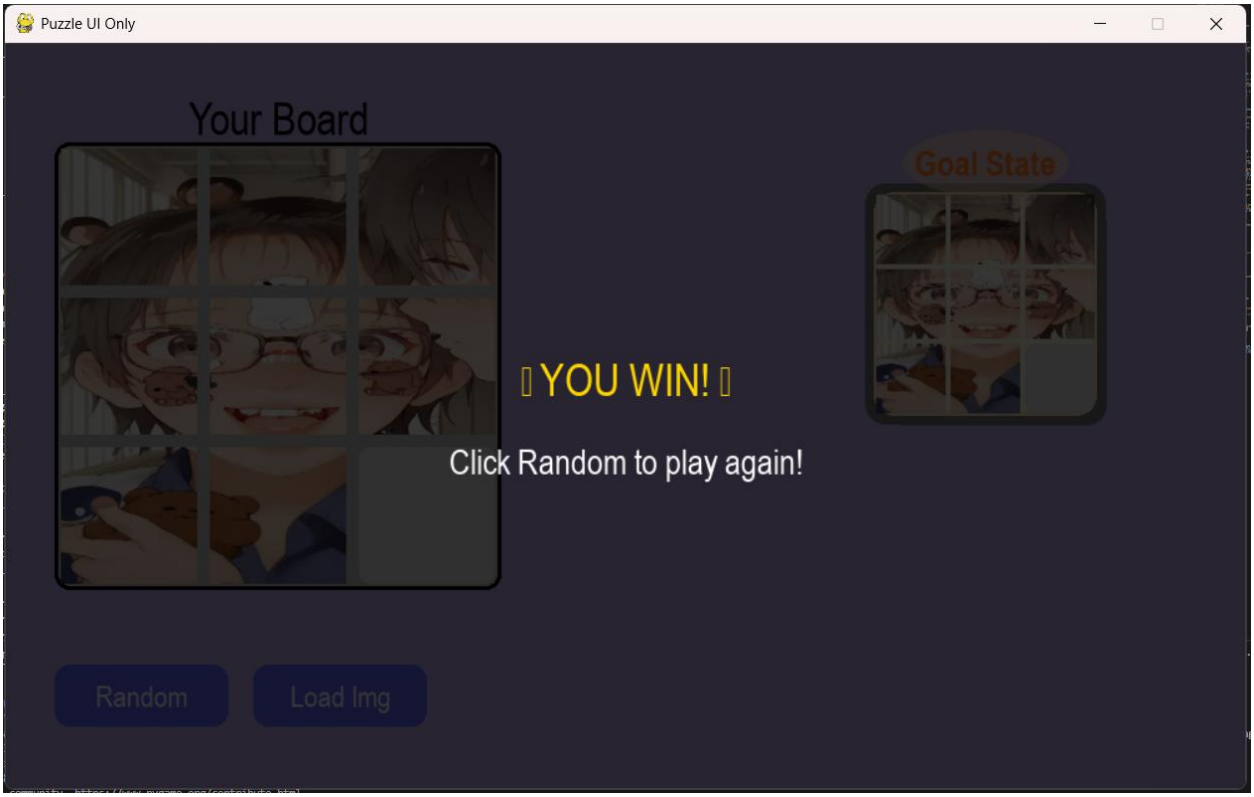
```
            x + c * cell_size + gap,
            y + r * cell_size + gap,
            cell_size - 2*gap,
            cell_size - 2*gap
        )
        if tiles and num:
            img = tiles[num-1]
            img_s = pygame.transform.smoothscale(img, (cell_size-2*gap,
cell_size-2*gap))
            screen.blit(img_s, rect)
        else:
            pygame.draw.rect(screen, (255,255,255) if num else (220,220,220),
rect, border_radius=8)
            if num:
                txt = font.render(str(num), True, (30,30,30))
                txt_rect = txt.get_rect(center=rect.center)
                screen.blit(txt, txt_rect)
    txt = font.render(label, True, (0,0,0))
    screen.blit(txt, (x + w//2 - txt.get_width()//2, y-40))

def draw_goal_board(screen, x, y, font, tiles=None):
    cell_size = CELL_SIZE // 2
    gap = GAP // 2
    w = cell_size * COLS
    h = cell_size * ROWS
    # Shadow effect
    shadow_rect = pygame.Rect(x+8, y+8, w, h)
    pygame.draw.rect(screen, (120, 120, 180, 80), shadow_rect, border_radius=16)
    # Gradient border (simulate)
    for i in range(8):
        color = (180-i*10, 180-i*10, 255-i*20)
        pygame.draw.rect(screen, color, (x-i, y-i, w+2*i, h+2*i),
border_radius=16)
    # Main background
    bg_rect = pygame.Rect(x, y, w, h)
    pygame.draw.rect(screen, (255, 245, 220), bg_rect, border_radius=16)
    # Inner glowing effect
    pygame.draw.rect(screen, (255, 220, 180), bg_rect, 4, border_radius=16)
    goal = [1,2,3,4,5,6,7,8,None]
    for i, num in enumerate(goal):
        r, c = divmod(i, COLS)
        rect = pygame.Rect(
            x + c * cell_size + gap,
            y + r * cell_size + gap,
            cell_size - 2*gap,
```

```
            cell_size - 2*gap
        )
        if tiles and num:
            img = tiles[num-1]
            img_s = pygame.transform.smoothscale(img, (cell_size-2*gap,
cell_size-2*gap))
            screen.blit(img_s, rect)
        else:
            pygame.draw.rect(screen, (255,255,255) if num else (220,220,220),
rect, border_radius=8)
            if num:
                txt = font.render(str(num), True, (255, 120, 40))
                txt_rect = txt.get_rect(center=rect.center)
                screen.blit(txt, txt_rect)
    # Tiêu đề nổi bật
    title_font = pygame.font.SysFont("Arial", 28, bold=True)
    txt = title_font.render("Goal State", True, (255, 120, 40))
    glow = pygame.Surface((txt.get_width()+20, txt.get_height()+20),
pygame.SRCALPHA)
    pygame.draw.ellipse(glow, (255,220,180,120), glow.get_rect())
    screen.blit(glow, (x + w//2 - txt.get_width()//2 - 10, y-50))
    screen.blit(txt, (x + w//2 - txt.get_width()//2, y-40))

def draw_button(screen, rect, text, font):
    pygame.draw.rect(screen, BUTTON_COLOR, rect, border_radius=12)
    txt = font.render(text, True, TEXT_COLOR)
    screen.blit(txt, (rect.centerx - txt.get_width()//2, rect.centery -
txt.get_height()//2))

def draw_victory(screen, font):
    m_default_img = os.path.join("images", "icon.png")
    overlay = pygame.Surface((WIDTH, HEIGHT))
    overlay.set_alpha(200)
    overlay.fill((0,0,0))
    screen.blit(overlay, (0,0))
    text = font.render("🎉 YOU WIN! 🎉", True, (255,215,0))
    screen.blit(text, (WIDTH//2 - text.get_width()//2, HEIGHT//2 - 50))
    sub = pygame.font.SysFont("Arial", 28).render("Click Random to play again!",
True, (255,255,255))
    screen.blit(sub, (WIDTH//2 - sub.get_width()//2, HEIGHT//2 + 20))

def move_tile(board, pos):
    idx = board.index(None)
    row, col = divmod(idx, COLS)
    r2, c2 = divmod(pos, COLS)
```

```
    if abs(row - r2) + abs(col - c2) == 1:
        board[idx], board[pos] = board[pos], board[idx]

def board_random():
    board = list(range(1, 9)) + [None]
    def is_solvable(board):
        arr = [x for x in board if x is not None]
        inv_count = 0
        for i in range(len(arr)):
            for j in range(i+1, len(arr)):
                if arr[i] > arr[j]:
                    inv_count += 1
        return inv_count % 2 == 0
    while True:
        random.shuffle(board)
        if board != list(range(1, 9)) + [None] and is_solvable(board):
            return board

def check_win(board):
    return board == [1,2,3,4,5,6,7,8,None]

btn_random = pygame.Rect(40, 500, 140, 50)
btn_load   = pygame.Rect(200, 500, 140, 50)

board = board_random()
m_default_img = os.path.join("images", "icon.png")
tiles = load_and_slice(m_default_img)
goal_tiles = load_and_slice(m_default_img, cell_size=CELL_SIZE//2)
moves = 0
win = False
running = True
while running:
    for ev in pygame.event.get():
        if ev.type == pygame.QUIT:
            running = False
        elif ev.type == pygame.MOUSEBUTTONDOWN:
            if btn_random.collidepoint(ev.pos):
                board = board_random()
                tiles = load_and_slice(m_default_img)
                moves = 0
                win = False
            elif btn_load.collidepoint(ev.pos):
                Tk().withdraw()
                path = filedialog.askopenfilename(filetypes=[("Image
files","*.png;*.jpg;*.jpeg;*.bmp")])
```

```
                if path:
                    tiles = load_and_slice(path)
                    # goal_tiles giữ nguyên ảnh mặc định
                    board = board_random()
                    moves = 0
                    win = False
            else:
                mx,my = ev.pos
                if LEFT_BOARD_X <= mx < LEFT_BOARD_X+LEFT_BOARD_W and
LEFT_BOARD_Y <= my < LEFT_BOARD_Y+LEFT_BOARD_H:
                    c = (mx-LEFT_BOARD_X)//CELL_SIZE
                    r = (my-LEFT_BOARD_Y)//CELL_SIZE
                    before = board[:]
                    move_tile(board, r*COLS+c)
                    if board != before:
                        moves += 1
                        if check_win(board):
                            win = True
    screen.fill(BG_COLOR)
    draw_board(screen, board, LEFT_BOARD_X, LEFT_BOARD_Y, "Your Board", font,
tiles)
    draw_goal_board(screen, GOAL_BOARD_X, GOAL_BOARD_Y, small_font, goal_tiles)
    draw_button(screen, btn_random, "Random", small_font)
    draw_button(screen, btn_load, "Load Img", small_font)
    if win:
        draw_victory(screen, font)
    pygame.display.flip()
    clock.tick(60)
pygame.quit()
```


Puzzle UI Only
Your Board
Goal State
Random
Load Img


Puzzle UI Only
Your Board
Goal State
YOU WIN!
Click Random to play again!
Random
Load Img