```
import pygame
import random
import datetime
import json
import os

pygame.init()

SCREEN_WIDTH = 400
SCREEN_HEIGHT = 600
BIRD_RADIUS = 12
PIPE_WIDTH = 80
PIPE_HEIGHT = 500
PIPE_GAP = 200
GRAVITY = 0.5
FLAP_STRENGTH = -10

WHITE = (255, 255, 255)
BLUE = (0, 0, 255)
GREEN = (0, 255, 0)
BLACK = (0, 0, 0)
RED = (255, 0, 0)

LEADERBOARD_FILE = "leaderboard.json"

screen = pygame.display.set_mode((SCREEN_WIDTH, SCREEN_HEIGHT))
pygame.display.set_caption('My Game')
```

```
clock = pygame.time.Clock()
font = pygame.font.Font(None, 36)



class Button:
    def __init__(self, text, x, y, width, height, color, text_color):
        self.rect = pygame.Rect(x, y, width, height)
        self.color = color
        self.text = text
        self.text_color = text_color
        self.font = pygame.font.Font(None, 36)

    def draw(self, surface):
        pygame.draw.rect(surface, self.color, self.rect)
        text_surf = self.font.render(self.text, True, self.text_color)
        text_rect = text_surf.get_rect(center=self.rect.center)
        surface.blit(text_surf, text_rect)

    def is_hovered(self, pos):
        return self.rect.collidepoint(pos)



class Pipe:
    def __init__(self, x, y, passed=False):
        self.rect = pygame.Rect(x, y, PIPE_WIDTH, PIPE_HEIGHT)
        self.passed = passed

    def draw(self, surface):
```

```
        pygame.draw.rect(surface, GREEN, self.rect)

    def move(self, speed):
        self.rect.x -= speed



def draw_bird(bird_rect):
    pygame.draw.circle(screen, BLUE, bird_rect.center, BIRD_RADIUS)



def draw_score(score):
    score_text = font.render(f'Score: {score}', True, BLACK)
    screen.blit(score_text, (10, 10))



def game_over_screen(score):
    screen.fill(WHITE)
    game_over_text = font.render('Game Over', True, BLACK)
    score_text = font.render(f'Score: {score}', True, BLACK)
    restart_button = Button('Restart', SCREEN_WIDTH // 2 - 60, SCREEN_HEIGHT // 2 +
50, 120, 50, RED, WHITE)
    leaderboard_button = Button('Leaderboard', SCREEN_WIDTH // 2 - 75,
SCREEN_HEIGHT // 2 + 120, 150, 50, RED, WHITE)
    screen.blit(game_over_text, (SCREEN_WIDTH // 2 - game_over_text.get_width() // 2,
SCREEN_HEIGHT // 2 - 50))
    screen.blit(score_text, (SCREEN_WIDTH // 2 - score_text.get_width() // 2,
SCREEN_HEIGHT // 2))
    restart_button.draw(screen)
    leaderboard_button.draw(screen)
    pygame.display.flip()
```

```
    return restart_button, leaderboard_button



def save_score(score):
    if os.path.exists(LEADERBOARD_FILE):
        with open(LEADERBOARD_FILE, "r") as file:
            leaderboard = json.load(file)
    else:
        leaderboard = []

    current_time = datetime.datetime.now().strftime("%d/%m/%Y, %H:%M")
    leaderboard.append({"score": score, "date": current_time})
    leaderboard = sorted(leaderboard, key=lambda x: x["score"], reverse=True)[:10]

    with open(LEADERBOARD_FILE, "w") as file:
        json.dump(leaderboard, file)



def show_leaderboard():
    if os.path.exists(LEADERBOARD_FILE):
        with open(LEADERBOARD_FILE, "r") as file:
            leaderboard = json.load(file)
    else:
        leaderboard = []

    screen.fill(WHITE)
    title_text = font.render('Leaderboard', True, BLACK)
    screen.blit(title_text, (SCREEN_WIDTH // 2 - title_text.get_width() // 2, 50))
```

```
    for i, entry in enumerate(leaderboard):
        score_text = font.render(f'{i + 1}. {entry["score"]} - {entry["date"]}', True, BLACK)
        screen.blit(score_text, (20, 100 + i * 40))

    back_button = Button('Back', SCREEN_WIDTH // 2 - 60, SCREEN_HEIGHT - 100,
120, 50, RED, WHITE)
    back_button.draw(screen)
    pygame.display.flip()
    return back_button



def main():
    bird_rect = pygame.Rect(100, SCREEN_HEIGHT // 2, BIRD_RADIUS * 2,
BIRD_RADIUS * 2)
    bird_velocity = 0
    pipes = []
    pipe_timer = 0
    score = 0
    running = True
    game_over = False
    pipe_speed = 5
    passed_pipes = set()

    while running:
        for event in pygame.event.get():
            if event.type == pygame.QUIT:
                running = False
            if event.type == pygame.KEYDOWN:
```

```
            if event.key == pygame.K_SPACE and not game_over:
                bird_velocity = FLAP_STRENGTH
            if event.key == pygame.K_r and game_over:
                main()
                return

    if not game_over:
        bird_velocity += GRAVITY
        bird_rect.y += int(bird_velocity)

        pipe_timer += 1
        if pipe_timer > 90:
            pipe_timer = 0
            pipe_height = random.randint(100, 400)
            pipes.append(Pipe(SCREEN_WIDTH, pipe_height - PIPE_HEIGHT))
            pipes.append(Pipe(SCREEN_WIDTH, pipe_height + PIPE_GAP))

        for pipe in pipes:
            pipe.move(pipe_speed)

        for pipe in pipes:
            if bird_rect.colliderect(pipe.rect):
                game_over = True
        if bird_rect.y > SCREEN_HEIGHT or bird_rect.y < 0:
            game_over = True

        pipes = [pipe for pipe in pipes if pipe.rect.x > -PIPE_WIDTH]
```

```
        for pipe in pipes:
            if bird_rect.x > pipe.rect.x + PIPE_WIDTH and not pipe.passed:
                if pipe not in passed_pipes:
                    score += 0.5
                    passed_pipes.add(pipe)

        screen.fill(WHITE)
        draw_bird(bird_rect)
        for pipe in pipes:
            pipe.draw(screen)
        draw_score(int(score))
        pygame.display.flip()

        clock.tick(30)

        pipe_speed += 0.005

    if game_over:
        save_score(int(score))
        restart_button, leaderboard_button = game_over_screen(int(score))
        while True:
            for event in pygame.event.get():
                if event.type == pygame.QUIT:
                    pygame.quit()
                    return
                if event.type == pygame.MOUSEBUTTONDOWN:
                    if restart_button.is_hovered(event.pos):
                        main()
```

```
                    return
                if leaderboard_button.is_hovered(event.pos):
                    back_button = show_leaderboard()
                    while True:
                        for event in pygame.event.get():
                            if event.type == pygame.QUIT:
                                pygame.quit()
                                return
                            if event.type == pygame.MOUSEBUTTONDOWN:
                                if back_button.is_hovered(event.pos):
                                    game_over_screen(int(score))
                                    break
                        else:
                            continue
                        break
                    return

            pygame.display.flip()
            clock.tick(30)

    pygame.quit()


if __name__ == "__main__":
    main()
```