“FLOOR PLAN GENERATION BY USING PROMPTS GIVEN BY USER”

```
import matplotlib.pyplot as plt
import matplotlib.patches as patches
from ipywidgets import widgets
from IPython.display import display, clear_output

# Function to parse the user prompt into a list of room specifications
def parse_prompt(prompt):
    room_specs = []
    for part in prompt.split(','):
        count, room_type, *spec = part.strip().split()
        if 'x' in spec[0]:  # Size is specified
            size = tuple(map(int, spec[0].split('x')))
        else:  # No size specified, assume default 3x3
            size = (3, 3)
        for _ in range(int(count)):  # Add room specifications
            room_specs.append((room_type, size))
    return room_specs

# Function to draw a room on the floor plan with doors and windows
def generate_room(x, y, w, h, room_type, ax, width, height, alignment, prev_x, prev_y):
    # Draw the room
    ax.add_patch(patches.Rectangle((x, y), w, h, fill=None, edgecolor='blue', linewidth=1))
    ax.text(x + w/2, y + h/2, room_type, ha='center', va='center', fontsize=8)

    # Door and window dimensions
    door_width, door_height = 0.2 * w, 0.1 * h
    window_width, window_height = 0.2 * w, 0.1 * h

    # Place the door towards the living room
    if alignment == 'left' or (alignment == 'right' and prev_y != y):
        door_x, door_y = x + w - door_width, y + (h - door_height) / 2
    elif alignment == 'right' or (alignment == 'left' and prev_y != y):
        door_x, door_y = x, y + (h - door_height) / 2
    elif alignment == 'top' or (alignment == 'bottom' and prev_x != x):
        door_x, door_y = x + (w - door_width) / 2, y + h - door_height
    else:  # 'bottom' or horizontal arrangement
        door_x, door_y = x + (w - door_width) / 2, y

    # Place the window facing outside
    if x == 0:  # Window on the left wall
        window_x, window_y = x, y + (h - window_height) / 2
    elif x + w == width:  # Window on the right wall
        window_x, window_y = x + w - window_width, y + (h - window_height) / 2
    elif y == 0:  # Window on the bottom wall
        window_x, window_y = x + (w - window_width) / 2, y
    else:  # Window on the top wall
        window_x, window_y = x + (w - window_width) / 2, y + h - window_height

    # Draw door and window
    ax.add_patch(patches.Rectangle((door_x, door_y), door_width, door_height, fill=True,
color='brown'))
    ax.add_patch(patches.Rectangle((window_x, window_y), window_width, window_height, fill=True,
color='lightblue'))
    return x + w, y + h

# Function to create a single floor plan with specified alignment of rooms
def create_floor_plan(rooms, width, height, alignment):
    fig, ax = plt.subplots(figsize=(8, 8))
    ax.set_xlim(0, width)
    ax.set_ylim(0, height)
    ax.set_aspect('equal', 'box')
    ax.axis('off')
    # Draw the outline for the floor plan
    ax.add_patch(patches.Rectangle((0, 0), width, height, fill=None, edgecolor='black', linewidth=2))
    x, y, prev_x, prev_y = 0, 0, 0, 0
    for room_type, (room_w, room_h) in rooms:
        if x + room_w <= width and y + room_h <= height:
            prev_x, prev_y = generate_room(x, y, room_w, room_h, room_type, ax, width, height,
alignment, prev_x, prev_y)
            if alignment in ['left', 'right']:
                y += room_h
                if y >= height:
                    y = 0
                    x += room_w
            else:
                x += room_w
                if x >= width:
                    x = 0
                    y += room_h
    return fig

# Function to generate multiple floor plans with different alignments
def generate_aligned_floor_plans(prompt):
    room_specs = parse_prompt(prompt)
    alignments = ['left', 'right', 'top', 'bottom']
    for alignment in alignments:
        fig = create_floor_plan(room_specs, width=12, height=12, alignment=alignment)
        plt.show()

# Interactive widgets for user input
prompt_input = widgets.Text(
    value='',
    placeholder='Enter room specs, e.g., "2 bedrooms 3x3, 1 bathroom 2x2"',
    description='Prompt:',
    disabled=False
)

generate_button = widgets.Button(description="Generate Floor Plans")

# Function to handle button click event
def on_generate_button_clicked(b):
    with output:
        clear_output(wait=True)
        generate_aligned_floor_plans(prompt_input.value)

# Output widget to display the floor plans
```

```
output = widgets.Output()

# Display the widgets
display(prompt_input, generate_button, output)

# Bind the button click to the function
generate_button.on_click(on_generate_button_clicked)
```