```
import random as random
from kivy.app import App
from kivy.clock import Clock
from kivy.lang import Builder
from kivy.properties import NumericProperty, ReferenceListProperty, ObjectProperty
from kivy.uix.boxlayout import BoxLayout
from kivy.graphics import Color, Ellipse, Rectangle
from kivy.uix.floatlayout import FloatLayout
from kivy.uix.label import Label
from kivy.uix.widget import Widget
from kivy.vector import Vector

# Kvlang string to create widgets and binding for GUI
kv = """

<GamePiece1>:
  size_hint: .025, .03
  canvas:
    Color:
      rgb: 1, 0, 0
    Ellipse:
      pos: self.pos
      size: self.size
<GamePiece2>:
  size_hint: .025, .03
  canvas:
    Color:
      rgb: 1, 1, 0
    Ellipse:
      pos: self.pos
      size: self.size
<GamePiece3>:
  size_hint: .025, .03
  canvas:
    Color:
      rgb: 1, 1, 1
    Ellipse:
      pos: self.pos
      size: self.size
<GamePiece4>:
  size_hint: .025, .03
  canvas:
    Color:
      rgb: 0, .6, 0
    Ellipse:
      pos: self.pos
      size: self.size
<GamePiece5>:
  size_hint: .025, .03
```

```
    canvas:
        Color:
            rgb: 0, .4, .8
        Ellipse:
            pos: self.pos
            size: self.size
<GamePiece6>:
    size_hint: .025, .03
    canvas:
        Color:
            rgb: .4, 0, .8
        Ellipse:
            pos: self.pos
            size: self.size

<MainWidget>:
    piece_red: scarlet
    piece_yellow: mustard
    piece_white: white
    piece_green: green
    piece_blue: blue
    piece_purple: purple

    FloatLayout:
        id: board
        canvas:
            Rectangle:
                size: root.size
                source: "clue_board2.png"

    GamePiece1:
        id: scarlet
        pos: self.new_pos

    GamePiece2:
        id: mustard
        pos: self.new_pos

    GamePiece3:
        id: white
        pos: self.new_pos

    GamePiece4:
        id: green
        pos: self.new_pos

    GamePiece5:
        id: blue
        pos: self.new_pos
```

```
    GamePiece6:
        id: purple
        pos: self.new_pos



"""

# load Kvlang string to build layout
Builder.load_string(kv)


# declare classes for player widgets
class GamePiece1(Widget):
    pos_x = NumericProperty(0)
    pos_y = NumericProperty(0)
    new_pos = ReferenceListProperty(pos_x, pos_y)

    def move_piece(self, x, y):
        self.pos_x = x
        self.pos_y = y
        self.pos = Vector(*self.new_pos)


class GamePiece2(Widget):
    pos_x = NumericProperty(0)
    pos_y = NumericProperty(0)
    new_pos = ReferenceListProperty(pos_x, pos_y)

    def move_piece(self, x, y):
        self.pos_x = x
        self.pos_y = y
        self.pos = Vector(*self.new_pos)


class GamePiece3(Widget):
    pos_x = NumericProperty(0)
    pos_y = NumericProperty(0)
    new_pos = ReferenceListProperty(pos_x, pos_y)

    def move_piece(self, x, y):
        self.pos_x = x
        self.pos_y = y
        self.pos = Vector(*self.new_pos)


class GamePiece4(Widget):
    pos_x = NumericProperty(0)
```

```
    pos_y = NumericProperty(0)
    new_pos = ReferenceListProperty(pos_x, pos_y)

    def move_piece(self, x, y):
        self.pos_x = x
        self.pos_y = y
        self.pos = Vector(*self.new_pos)


class GamePiece5(Widget):
    pos_x = NumericProperty(0)
    pos_y = NumericProperty(0)
    new_pos = ReferenceListProperty(pos_x, pos_y)

    def move_piece(self, x, y):
        self.pos_x = x
        self.pos_y = y
        self.pos = Vector(*self.new_pos)


class GamePiece6(Widget):
    pos_x = NumericProperty(0)
    pos_y = NumericProperty(0)
    new_pos = ReferenceListProperty(pos_x, pos_y)

    def move_piece(self, x, y):
        self.pos_x = x
        self.pos_y = y
        self.pos = Vector(*self.new_pos)


# define main widget class and functions
class MainWidget(FloatLayout):
    rounds = 2
    # delay = 2
    players = ['Miss Scarlet', "Colonel Mustard", "Mrs. White", "Mr. Green", "Mrs. Peacock", "Professor Plum"]
    coord_list = [[1, 17], [8, 24], [25, 15], [25, 10], [19, 1], [6, 1]]
    player = ''
    piece_red = ObjectProperty(None)
    piece_yellow = ObjectProperty(None)
    piece_white = ObjectProperty(None)
    piece_green = ObjectProperty(None)
    piece_blue = ObjectProperty(None)
    piece_purple = ObjectProperty(None)

    def __init__(self, **kwargs):
        super(MainWidget, self).__init__(**kwargs)
        Clock.schedule_once(self.init_player_positions, 0)
```

```
    def on_touch_down(self, touch):
        self.coord_list = self.generate_new_coordinates()
        Clock.schedule_once(self.init_player_positions, 0)

    def generate_new_coordinates(self):
        # Written by Nayana - updated by Harshitha and Brian
        new_coord_list = []
        for _ in range(6):
            x = random.randint(1, 24)
            y = random.randint(1, 25)
            new_coord_list.append([x, y])
        return new_coord_list

    def init_player_positions(self, dt):
        for i in range(0, len(self.players)):
            self.player = self.players[i]
            if self.player == "Miss Scarlet":
                x, y = self.get_coords('Miss Scarlet')
                self.piece_red.move_piece(self.width*x/26, self.height*y/27)
            if self.player == "Colonel Mustard":
                x, y = self.get_coords('Colonel Mustard')
                self.piece_yellow.move_piece(self.width*x/26, self.height*y/27)
            if self.player == "Mrs. White":
                x, y = self.get_coords('Mrs. White')
                self.piece_white.move_piece(self.width*x/26, self.height*y/27)
            if self.player == "Mr. Green":
                x, y = self.get_coords('Mr. Green')
                self.piece_green.move_piece(self.width*x/26, self.height*y/27)
            if self.player == "Mrs. Peacock":
                x, y = self.get_coords('Mrs. Peacock')
                self.piece_blue.move_piece(self.width*x/26, self.height*y/27)
            if self.player == "Professor Plum":
                x, y = self.get_coords('Professor Plum')
                self.piece_purple.move_piece(self.width*x/26, self.height*y/27)

    def get_coords(self, name):
        for i in range(0, len(self.players)):
            if self.players[i] == name:
                x = self.coord_list[i][1]
                y = 26 - self.coord_list[i][0]
        return x, y


class DEEPClueApp(App):

    def build(self):
        game = MainWidget()
        return game
```

```
if __name__ == '__main__':
    DEEPClueApp().run()
```