```python
from os import closerange
from ursina import *
import json
from ursina.prefabs.first_person_controller import FirstPersonController

# Classes
def init_world(x, z):
    sky = Sky(texture="assets/top.png")

    floor = Button(parent= scene,
                   color = color.white33,
                   position = (0,0,0),
                   origin_y = 0,
                   model = "cube",
                   texture = "white_block",
                   scale = (x, 1, z)
                   )

    player = FirstPersonController()


def resize(texture: Texture, new_x):
        aspect_ratio = texture.width / texture.height
        new_y = new_x / aspect_ratio

        return (new_x, new_y)

class Frame(Button):
    def __init__(self, img, position, scale_x, description):
        texture = load_texture(img)
        scale = resize(texture, scale_x)
        self.description = description
        super().__init__(model = "cube",
                        parent = scene,
                        texture = texture,
                        position = position,
                        color = color.white,
                        scale = scale,
                        origin_y = -0.5)

    def input(self, key):

        if self.hovered:
            if key == "left mouse down":
                #pass to a viewer-like vision
                print(self.description)
                pass

```

```

def load_frames(json_file):
    entities = []
    with open(json_file, "r") as f:
        frames = json.load(f)

    north, south = frames["North"], frames["South"]
    west, east = frames["West"], frames["East"]

    for frame in north:
        #positive z, x remain
        #negative x rotation
        fr = Frame(img="assets/" + frame["file"],
                   position = (frame["pos"]["x"], frame["pos"]["y"], frame["pos"]["z"],),
                   scale_x = frame["scale"],
                   description = frame["description"])
        fr.rotation_y += frame["Yrotate"]
        fr.rotation_x -= frame["Xrotate"]
        entities.append(fr)

        antifr = Frame(img="assets/" + frame["file"],
                   position = (frame["pos"]["x"], - frame["pos"]["y"] - 1, frame["pos"]["z"],),
                   scale_x = frame["scale"],
                   description = frame["description"])
        antifr.rotation_y += frame["Yrotate"]
        antifr.rotation_x -= frame["Xrotate"] - 185
        entities.append(fr)

    for frame in south:
        #negative z, remain x
        #positive x rotation
        fr = Frame(img="assets/" + frame["file"],
                   position = (frame["pos"]["x"], frame["pos"]["y"], - frame["pos"]["z"],),
                   scale_x = frame["scale"],
                   description = frame["description"])
        fr.rotation_y += frame["Yrotate"]
        fr.rotation_x += frame["Xrotate"]
        entities.append(fr)

    for frame in west:
        #remain z, negative x
        #positive x rotation + 90Âº
        fr = Frame(img="assets/" + frame["file"],
                   position = (-frame["pos"]["x"], frame["pos"]["y"], frame["pos"]["z"],),
                   scale_x = frame["scale"],
                   description = frame["description"])
        fr.rotation_y += frame["Yrotate"] + 90
        fr.rotation_x += frame["Xrotate"]
```

```python
            entities.append(fr)

    for frame in east:
        #remain z, negative x
        #positive x rotation - 90Âº
        fr = Frame(img="assets/" + frame["file"],
                      position = (frame["pos"]["x"], frame["pos"]["y"], -frame["pos"]["z"],),
                      scale_x = frame["scale"],
                      description = frame["description"])
        fr.rotation_y += frame["Yrotate"] + 90
        fr.rotation_x -= frame["Xrotate"]
        entities.append(fr)



    return entities

class Spectator(FirstPersonController):
    def __init__(self):
        super().__init__(position = (0, 1, 0))
        self.flying = False

    def update(self):
        if self.flying:
            self.gravity = 0
            self.y += (held_keys["v"] - held_keys["b"]) * time.dt * 10
        else:
            self.gravity = 1

        return super().update()

    def input(self, key):
        if key == "v":
            self.start_flying()
        elif key == "c":
            self.stop_flying()

        return super().input(key)

    def start_flying(self):
        self.flying = True

    def stop_flying(self):
        self.flying = False

app = Ursina()

init_world(100, 100)
```

```python
frames = load_frames("/Users/rserrano/grepos/ProyectoFinal/src/obras.json")
player = Spectator()

app.run()
```