```
void Application::fire_guard(float fire_guard_radius, int fire_guard_subdiv, int num_brick_rows,
    float brick_width, float brick_height, float brick_depth,
    float fire_guard_x_position, float fire_guard_y_position, float fire_guard_z_position,
    float fire_guard_x_rotation, float fire_guard_y_rotation, float fire_guard_z_rotation) {

    // draw seat---------------------------------
    vec3 centroid(fire_guard_x_position, fire_guard_y_position, fire_guard_z_position);

    float radian_incr = 2 * pi<float>() / fire_guard_subdiv;

    for (int i = 0; i < num_brick_rows; i++) {
        for (int j = 0; j < fire_guard_subdiv; j++) {
            if (i % 2 == 0) {       // if is an even row
                vec3 curr_point = vec3(centroid.x + fire_guard_radius,
                    centroid.y + (brick_height * 0.5) + (brick_height * i),
                    centroid.z);

                bm_fire_guard_object_ptr = new basic_model;
                bm_fire_guard_object_ptr->mesh = rectangular_prism(brick_width, brick_height, brick_depth,
                    curr_point.x, curr_point.y, curr_point.z,
                    0, degrees(radian_incr * 0.5) * j, 0, 0, degrees(radian_incr) * j, 0, 1);
                bm_fire_guard_object_ptr->color = vec3(0.3, 0.3, 0.3);    // assign color to object
                bm_fire_guard_object_ptr->shader = just_shader;    // assign shader to object
                m_fire_guard_objects.push_back(bm_fire_guard_object_ptr);      // append current basic_model object
            }
            else {          // if is odd row
                vec3 curr_point = vec3(centroid.x + fire_guard_radius,
                    centroid.y + (brick_height * 0.5) + (brick_height * i),
                    centroid.z);

                bm_fire_guard_object_ptr = new basic_model;
                bm_fire_guard_object_ptr->mesh = rectangular_prism(brick_width, brick_height, brick_depth,
                    curr_point.x, curr_point.y, curr_point.z,
                    0, degrees(radian_incr * 0.5) * j, 0, 0, degrees(radian_incr) * j + degrees(radian_incr * 0.5), 0, 1);
                bm_fire_guard_object_ptr->color = vec3(0.3, 0.3, 0.3);    // assign color to object
                bm_fire_guard_object_ptr->shader = just_shader;    // assign shader to object
                m_fire_guard_objects.push_back(bm_fire_guard_object_ptr);      // append current basic_model object
            }
        }
    }


}
```