def training\_detail(request, training\_id):

    with connection.cursor() as cursor:

            sql\_training = """

                SELECT training\_id, title, training\_date, start\_time, end\_time, venue, participants\_number, status

                FROM trainings

                WHERE training\_id = %s

            """

            cursor.execute(sql\_training, [training\_id])

            row = cursor.fetchone()

            if row:

                training = {

                    'training\_id': row[0],

                    'title': row[1],

                    'training\_date': row[2],

                    'start\_time': row[3],

                    'end\_time': row[4],

                    'venue': row[5],

                    'participants\_number': row[6],

                    'status': row[7]

                }

            # 获取该 training 的所有参与者

            id\_list = []

            participants = []

            if row[6] > 0:

                sql\_participants = """

                    SELECT artists.artist\_id, artists.name, artists.age, artists.gender, training\_participants.body\_status, artists.phone

                    FROM training\_participants INNER JOIN artists

                    ON training\_participants.artist\_id = artists.artist\_id

                    WHERE training\_id = %s

                """

                cursor.execute(sql\_participants, [training\_id])

                row = cursor.fetchall()

                if row:

                    for p in row:

                        participants.append({

                            'artist\_id': p[0],

                            'artist\_name': p[1],

                            'artist\_age': p[2],

                            'artist\_gender': p[3],

                            'artist\_body\_status': p[4],

                            'artist\_phone': p[5]

                        })

                        id\_list.append(p[0])

            # 查询artists表

            if len(id\_list) > 0:

                sql\_artists = """

                    SELECT artist\_id, name, age, gender, phone FROM artists

                    WHERE artist\_id NOT IN %s

                """

                cursor.execute(sql\_artists, [id\_list])

            else:

                sql\_artists = """

                    SELECT artist\_id, name, age, gender, phone FROM artists

                """

                cursor.execute(sql\_artists)

            row = cursor.fetchall()

            # 将查询结果转换为字典列表

            artists = []

            for artist in row:

                artists.append({

                    'artist\_id': artist[0],

                    'artist\_name': artist[1],

                    'age': artist[2],

                    'gender': artist[3],

                    'phone': artist[4]

                })

    # 检查是否处于编辑模式

    edit\_mode = request.GET.get('edit') == 'true'

    # 处理编辑表单的提交

    if request.method == 'POST':

        # 更新 training 信息

        title = request.POST.get('title')

        training\_date = request.POST.get('date')

        start\_time = request.POST.get('start\_time')

        end\_time = request.POST.get('end\_time')

        venue = request.POST.get('venue')

        status = request.POST.get('status')

        with connection.cursor() as cursor:

            participants\_ids = json.loads(request.POST.get('participants\_ids', '[]'))

            participants\_body\_status =json.loads(request.POST.get('participants\_body\_status', '[]'))

            # 更新表演信息

            sql = """

                UPDATE trainings

                SET title = %s, training\_date = %s, start\_time = %s, end\_time = %s, venue = %s, participants\_number = %s, status = %s

                WHERE training\_id = %s

            """

            cursor.execute(sql, [title, training\_date, start\_time, end\_time, venue, len(participants\_ids), status, training\_id])

            # 更新参与者信息

            sql = """

                SELECT artist\_id

                FROM training\_participants

                WHERE training\_id = %s

            """

            cursor.execute(sql, [training\_id])

            row = cursor.fetchall()

            exist\_ids = [str(x[0]) for x in row]

            if len(exist\_ids) > len(participants\_ids):

                new\_ids = list(set(exist\_ids) - set(participants\_ids))

                for new\_id in new\_ids:

                    sql\_delete = """

                        DELETE FROM training\_participants

                        WHERE training\_id = %s

                        AND artist\_id = %s

                    """

                    cursor.execute(sql\_delete, [training\_id, new\_id])

            elif len(exist\_ids) < len(participants\_ids):

                new\_ids = list(set(participants\_ids) - set(exist\_ids))

                for new\_id in new\_ids:

                    sql\_insert = """

                        INSERT INTO training\_participants

                        (training\_id, artist\_id, body\_status)

                        VALUES (%s, %s, %s)

                    """

                    cursor.execute(sql\_insert, [training\_id, new\_id, 'Normal'])

            index = 0

            for status in participants\_body\_status:

                sql\_update = """

                    UPDATE training\_participants

                    SET body\_status = %s

                    WHERE training\_id = %s AND artist\_id = %s

                """

                cursor.execute(sql\_update, [status, training\_id, participants\_ids[index]])

                index += 1

        # 重定向到非编辑模式的详情页

        return HttpResponseRedirect(reverse('training\_detail', args=[training\_id]))

    # 渲染模板

    context = {

        'training': training,

        'participants': participants,

        'artists': artists,

        'edit\_mode': edit\_mode,

    }

    return render(request, 'dance/training\_detail.html', context)

def performance\_detail(request, performance\_id):

    try:

        # 获取表演详情及其相关参与者

        performance = Performance.objects.select\_related().get(performance\_id=performance\_id)

        # 获取该表演的所有参与者

        participants = Artist.objects.filter(performances=performance).select\_related().values()

        # 获取未参与该表演的艺术家

        existing\_artist\_ids = participants.values\_list('artist\_id', flat=True)

        available\_artists = Artist.objects.exclude(artist\_id\_\_in=existing\_artist\_ids).values()

        # 检查是否处于编辑模式

        edit\_mode = request.GET.get('edit') == 'true'

        # 处理编辑表单的提交

        if request.method == 'POST':

            try:

                with transaction.atomic():  # 使用事务确保数据一致性

                    # 更新表演信息

                    performance.title = request.POST.get('title')

                    performance.performance\_date = parse\_date(request.POST.get('date'))

                    performance.start\_time = parse\_time(request.POST.get('start\_time'))

                    performance.end\_time = parse\_time(request.POST.get('end\_time'))

                    performance.venue = request.POST.get('venue')

                    performance.status = request.POST.get('status')

                    # 获取新的参与者列表

                    new\_participant\_ids = json.loads(request.POST.get('participants\_ids', '[]'))

                    performance.participants\_number = len(new\_participant\_ids)

                    # 验证并保存表演信息

                    performance.save()

                    # 更新参与者信息

                    current\_participants = list(performance.participants.values\_list('artist\_id', flat=True))

                    new\_participant\_ids = list(map(int, new\_participant\_ids))

                    # 要删除的参与者

                    to\_remove = []

                    for id in current\_participants:

                        if id not in new\_participant\_ids:

                            to\_remove.append(id)

                    if to\_remove:

                        PerformanceParticipant.objects.filter(performance=performance, artist\_id\_\_in=to\_remove).delete()

                    # 要添加的参与者

                    to\_add = []

                    for id in new\_participant\_ids:

                        if id not in current\_participants:

                            to\_add.append(id)

                    if to\_add:

                        new\_participants = []

                        for id in to\_add:

                            new\_artist = Artist.objects.get(artist\_id=id)

                            new\_participants.append(PerformanceParticipant(performance=performance, artist=new\_artist, role='artist'))

                        PerformanceParticipant.objects.bulk\_create(new\_participants)

                return HttpResponseRedirect(reverse('performance\_detail', args=[performance\_id]))

            except Exception as e:

                print(e)

                return HttpResponseRedirect(reverse('performance\_detail', args=[performance\_id]))

        context = {

            'performance': performance,

            'participants': participants,

            'available\_artists': available\_artists,

            'edit\_mode': edit\_mode,

        }

        return render(request, 'dance/performance\_detail.html', context)

    except Exception as e:

        print(e)

        return HttpResponseRedirect(reverse('performance\_detail', args=[performance\_id]))