



Correção: Pesquisa e Criação de Grupos no Chat

Data: 21 de Novembro de 2025

Status: ☒ Corrigido e Testado

Reportado por: Bruno (OrganiZen)



PROBLEMAS IDENTIFICADOS

Problema 1: Barra de Pesquisa Não Funcionava

Sintoma: Ao digitar "Duarion" na barra de pesquisa, nenhum resultado aparecia.

Causa Raiz:

- A pesquisa apenas filtrava **conversas existentes**
- Não mostrava **usuários disponíveis** para iniciar nova conversa
- Usuário sem conversas ativas não conseguia encontrar colegas

Print de Evidência:

[Barra de pesquisa com "Duarion" digitado]

Resultado: "Sem conversas"

Problema 2: Botão "Criar" Grupo Não Funcionava

Sintoma: Ao selecionar membros e clicar em "Criar", nada acontecia.

Causa Raiz:

- API `/api/chat/conversations` **não tinha método POST**
- Faltava endpoint para criar grupos
- Frontend chamava API inexistente
- Sem feedback de erro para o usuário

Print de Evidência:

Modal "Criar Grupo"

- Nome: "Teste"
- Membros: Ana, Beto, Camila selecionados
- Botão "Criar": clicável mas sem ação

✓ SOLUÇÕES IMPLEMENTADAS

Solução 1: Pesquisa de Usuários Disponíveis

Lógica Implementada:

```
// 1. Filtrar usuários baseado na pesquisa
const availableUsers = searchTerm ? users.filter(user => {
  const searchLower = searchTerm.toLowerCase();
  return user.name?.toLowerCase().includes(searchLower) ||
    user.email.toLowerCase().includes(searchLower);
}) : [];

// 2. Remover usuários que já têm conversa ativa
const usersWithoutConversation = availableUsers.filter(user =>
  !conversations.some(conv =>
    !conv.isGroup && conv.participants.some(p => p.userId === user.id)
  )
);

// 3. Exibir na lista com seção separada
{searchTerm && usersWithoutConversation.length > 0 && (
  <>
    <div className="px-3 py-2">
      <p className="text-xs text-muted-foreground font-semibold">
        {language === 'pt' ? 'NOVOS CONTATOS' : 'NEW CONTACTS'}
      </p>
    </div>
    {usersWithoutConversation.map((user) => (
      <Card onClick={() => handleStartConversationWithUser(user)}>
        {/* Card do usuário */}
      </Card>
    ))}
  </>
)}
```

Funcionalidade Adicionada:

- ✓ Busca por nome ou email
- ✓ Seção “NOVOS CONTATOS” separada
- ✓ Click para iniciar conversa
- ✓ Mensagem clara quando não há resultados

Fluxo de Uso:

1. Usuário digita "Duarion" na pesquisa
2. Sistema filtra conversas existentes
3. Sistema filtra usuários disponíveis
4. Exibe:
 - Conversas existentes (se houver)
 - NOVOS CONTATOS (usuários sem conversa)
5. Usuário clica em "Duarion"
6. Abre janela de chat com Duarion
7. Primeira mensagem cria conversa automaticamente

Solução 2: Criação de Grupos

Endpoint POST Implementado:

```
// POST /api/chat/conversations
export async function POST(request: NextRequest) {
  const { name, isGroup, memberIds } = body;

  // 1. Validar dados
  if (!name || !memberIds || memberIds.length === 0) {
    return NextResponse.json({
      error: 'Group name and members are required'
    }, { status: 400 });
  }

  // 2. Criar grupo
  const group = await prisma.chatGroup.create({
    data: {
      name,
      companyId,
      createdById: currentUserId,
      isActive: true
    }
  });

  // 3. Adicionar membros (incluindo criador)
  const allMemberIds = [currentUserId, ...memberIds];
  await prisma.chatGroupMember.createMany({
    data: uniqueMemberIds.map(userId => ({
      groupId: group.id,
      userId,
      role: userId === currentUserId ? 'admin' : 'member',
      companyId
    }))
  });

  // 4. Retornar conversa formatada
  return NextResponse.json(conversation, { status: 201 });
}
```

Funcionalidade Adicionada:

- ✓ Validação de nome e membros
- ✓ Criador automático como admin
- ✓ Todos membros adicionados
- ✓ Retorno formatado para frontend
- ✓ Toast de sucesso após criação

Fluxo de Uso:

1. Usuário clica em "+ Grupo"
2. Modal abre com formulário
3. Preenche "Nome do Grupo"
4. Seleciona membros (checkboxes)
5. Clica em "Criar"
6. API cria grupo no banco
7. Grupo aparece na lista
8. Toast "Grupo criado com sucesso!"
9. Modal fecha automaticamente

MELHORIAS ADICIONAIS




1. Conversas Temporárias (1:1)

Quando usuário clica em contato novo:

```
const handleStartConversationWithUser = (user: User) => {
  // Criar conversa temporária
  const newConversation: Conversation = {
    id: `temp-${user.id}`,
    name: null,
    isGroup: false,
    // ... participantes
  };

  setSelectedConversation(newConversation);
  setSearchTerm(''); // Limpar busca
};
```

Benefícios:





-  Usuário vê janela de chat imediatamente
-  Primeira mensagem cria conversa real
-  Sem necessidade de pré-criar conversas

2. Validação de Envio

```
// Detectar conversa temporária
if (selectedConversation.id.startsWith('temp-')) {
  // Enviar mensagem direta (cria conversa)
  const response = await fetch('/api/chat/messages', {
    method: 'POST',
    body: JSON.stringify({
      receiverId,
      content: newMessage.trim()
    })
  });

  if (response.ok) {
    router.refresh(); // Atualizar com conversa real
  }
} else {
  // Enviar para grupo existente
}
```

3. Feedback Visual

-  Mensagem **“Nenhum resultado encontrado”** quando pesquisa vazia
 -  Seção **“NOVOS CONTATOS”** destacada
 -  Hover em cards de usuários
 -  Toast de sucesso ao criar grupo
-

TESTES REALIZADOS

✓ Teste 1: Pesquisa de Usuário

1. Abrir página de chat
 2. Digitar "Duarion" na busca
 3. Verificar aparecimento em NOVOS CONTATOS
 4. Clicar no usuário
 5. Verificar abertura do chat
- RESULTADO: ✓ PASSOU

✓ Teste 2: Criação de Grupo

1. Clicar em "+ Grupo"
 2. Digitar "Teste" como nome
 3. Selecionar Ana, Beto, Camila
 4. Clicar em "Criar"
 5. Verificar grupo na lista
 6. Verificar toast de sucesso
- RESULTADO: ✓ PASSOU

✓ Teste 3: Mensagem em Conversa Nova

1. Pesquisar usuário "Juliana"
 2. Clicar para abrir chat
 3. Digitar mensagem "Olá"
 4. Enviar
 5. Verificar criação da conversa
 6. Verificar mensagem enviada
- RESULTADO: ✓ PASSOU

✓ Teste 4: Mensagem em Grupo

1. Criar grupo "Equipe"
 2. Adicionar membros
 3. Abrir grupo
 4. Enviar mensagem
 5. Verificar recebimento
- RESULTADO: ✓ PASSOU

IMPACTO DAS CORREÇÕES

Antes:

- ✗ Pesquisa não mostrava usuários
- ✗ Botão criar grupo não funcionava
- ✗ Impossível iniciar conversas novas
- ✗ Sem feedback ao usuário

Depois:

- ✓ Pesquisa mostra usuários disponíveis
- ✓ Criação de grupos funcional
- ✓ Conversas 1:1 criadas automaticamente
- ✓ Feedback claro em todas ações

Métricas:

- **Linhas adicionadas:** +254
- **Linhas modificadas:** -19
- **Arquivos alterados:** 2
- **Funções novas:** 2
- **Endpoints criados:** 1 (POST)



ARQUIVOS MODIFICADOS

1. `/app/api/chat/conversations/route.ts`

Mudanças:

- ✓ Adicionado método POST
- ✓ Criação de grupos com validação
- ✓ Retorno formatado

Linhas: +108 novas

2. `/components/chat-group-content.tsx`

Mudanças:

- ✓ Filtro de usuários disponíveis
- ✓ Função `handleStartConversationWithUser`
- ✓ Exibição de NOVOS CONTATOS
- ✓ Lógica de conversas temporárias

Linhas: +146 novas



COMO USAR

1. Buscar e Conversar com Usuário

1. Ir para página Chat
2. Digitar nome na busca
3. Clicar no usuário em NOVOS **CONTATOS**
4. Escrever e enviar mensagem

2. Criar Grupo


1. Clicar em "+ Grupo"
2. Digitar nome do grupo
3. Selecionar membros (mínimo 1)
4. Clicar em "Criar"
5. Grupo aparece na lista

3. Enviar Mensagem em Grupo

1. Clicar no grupo na lista
2. Digitar mensagem
3. (Opcional) Anexar arquivo
4. Clicar em enviar

CASOS DE USO

Caso 1: Novo Funcionário

Contexto: Maria acabou de entrar na empresa
Problema: Não tem conversas ainda
Solução:
1. Maria pesquisa "João" (seu supervisor)
2. João aparece em NOVOS CONTATOS
3. Maria clica e envia mensagem
4. Conversa  criada automaticamente

Caso 2: Criar Grupo de Projeto




Contexto: Início de novo projeto
Problema: Precisa grupo para comunicação
Solução:
1. Líder cria grupo "Projeto Alpha"
2. Adiciona membros da equipe
3. Envia primeira mensagem explicando projeto
4. Todos recebem notificação

Caso 3: Procurar Colega




Contexto: Precisa falar com colega específico
Problema: Não lembra o nome completo
Solução:
1. Digita parte do nome "Dua"
2. Sistema filtra "Duarion"
3. Aparece em NOVOS CONTATOS
4. Clica e inicia conversa

NOTAS IMPORTANTES





Permissões:

-  ADMIN, MANAGER, SUPERVISOR podem criar grupos
-  Todos podem iniciar conversas 1:1
-  Criador do grupo vira admin automaticamente

Validações:











-  Nome do grupo obrigatório
-  Mínimo 1 membro além do criador
-  Validação de duplicatas

Comportamento:

-  Pesquisa case-insensitive
-  Busca em nome e email
-  Conversas temporárias não salvas no banco
-  Primeira mensagem cria conversa real

PRÓXIMOS PASSOS

Melhorias Futuras:

1.  Adicionar descrição do grupo
2.  Permitir adicionar foto do grupo
3.  Busca avançada (por departamento, cargo)
4.  Histórico de buscas recentes
5.  Sugestões de contatos
6.  Favoritar conversas
7.  Arquivar conversas
8.  Silenciar notificações por tempo
9.  Transferir admin do grupo
10.  Remover membros do grupo

CHECKLIST DE VERIFICAÇÃO

- [x] Pesquisa funcional
- [x] Usuários disponíveis exibidos
- [x] Criação de grupos funcional
- [x] Conversas 1:1 automáticas
- [x] Feedback visual adequado
- [x] Validações implementadas
- [x] Testes realizados
- [x] Build passou sem erros
- [x] TypeScript sem erros

- [x] Git commit realizado
- [x] Checkpoint salvo
- [x] Documentação completa

Status Final:  **PRODUÇÃO**

Data de Conclusão: 21/11/2025

Desenvolvido por: DeepAgent para Bruno (OrganiZen)

SUPORTE

Se encontrar problemas:

1. Verificar se está autenticado
2. Verificar permissões do usuário
3. Conferir console do navegador
4. Consultar esta documentação
5. Verificar logs do servidor

Logs Úteis:

```
# API de conversas
console.log('Create group:', group)

# Filtro de usuários
console.log('Available users:', availableUsers)

# Conversas temporárias
console.log('Temp conversation:', newConversation)
```