Correção do Middleware de Tenant - Gráfico de Vendas

📋 Problema Identificado

O gráfico de vendas no dashboard não estava sendo exibido devido a um erro 404 com a mensagem “Loja não encontrada”.

Causa Raiz

O `TenantMiddleware` estava sendo aplicado em **todas as rotas** (`'*'`), incluindo as rotas do dashboard que já possuem autenticação JWT.

Fluxo Problemático:

1. **TenantMiddleware** é executado ANTES de qualquer guard
2. Ele tenta identificar a empresa através de:
   - Header `x-tenant-slug` OU
   - Subdomínio no host (ex: `pizza-express.deliverei.com.br`)
3. Se não encontrar nenhum dos dois, passa adiante (`next()`)
4. **MAS** se encontrar um slug/subdomínio e não encontrar a empresa no banco, lança `NotFoundException`

Conflito com Rotas Autenticadas:

- As rotas do dashboard (`/api/dashboard/*`) são protegidas por `JwtAuthGuard`
- O JWT já contém o `empresaId` do usuário autenticado
- O controller usa `req.user.empresaId` (do JWT) para buscar dados
- **Não há necessidade** do `TenantMiddleware` nessas rotas

Cenário do Erro:

Quando o frontend fazia a requisição para `/api/dashboard/vendas` :
- O `TenantMiddleware` era executado primeiro
- Não havia header `x-tenant-slug` nem subdomínio válido (localhost)
- O middleware passava adiante (`next()`)
- **OU** se houvesse algum header/subdomínio inválido, retornava 404

✅ Solução Implementada

Modificamos o `app.module.ts` para **excluir** as rotas do dashboard e autenticação do `TenantMiddleware` :

```
export class AppModule implements NestModule {
  configure(consumer: MiddlewareConsumer) {
    consumer
      .apply(TenantMiddleware)
      .exclude(
        // Excluir rotas do dashboard - usam JWT para identificar empresa
        'dashboard/(.*)',
        // Excluir rotas de autenticação
        'auth/(.*)',
      )
      .forRoutes('*');
  }
}
```

Por que essa solução funciona:

1. **Rotas do Dashboard**: Não precisam do `TenantMiddleware` porque:
   - Já têm autenticação JWT (`JwtAuthGuard`)
   - O `empresaId` vem do token do usuário
   - São rotas administrativas, não públicas

2. **Rotas de Autenticação**: Não precisam do middleware porque:
   - São rotas de login/registro
   - Não dependem de tenant específico

3. **Outras Rotas**: Continuam usando o `TenantMiddleware` :
   - Rotas públicas do cardápio (`/api/public/*`)
   - Rotas do carrinho (`/api/carrinho/*`)
   - Rotas de produtos (`/api/produtos/*`)
   - Essas rotas precisam identificar a loja pelo subdomínio/header

🧪 Como Testar

1. Teste Manual via cURL:

```
# Obter token de autenticação
TOKEN=$(curl -s -X POST http://localhost:3000/api/auth/login \
  -H "Content-Type: application/json" \
  -d '{"email":"admin@empresa.com","senha":"senha123"}' \
  | jq -r '.access_token')

# Testar endpoint de vendas
curl -X GET "http://localhost:3000/api/dashboard/vendas?periodo=dia" \
  -H "Authorization: Bearer $TOKEN" \
  -H "Content-Type: application/json"
```

2. Teste no Frontend:

1. Fazer login no dashboard
2. Navegar até a página inicial do dashboard
3. Verificar se o gráfico de vendas é exibido corretamente
4. Testar diferentes períodos (dia, semana, mês)

3. Verificar Logs:

```
# No terminal do backend, verificar se não há erros 404
# Deve mostrar requisições bem-sucedidas (200) para /api/dashboard/vendas
```

📊 Impacto da Mudança

✅ Benefícios:

- Gráfico de vendas funciona corretamente
- Melhor separação de responsabilidades
- Performance: menos processamento desnecessário em rotas autenticadas
- Código mais limpo e manutenível

⚠️ Considerações:

- Rotas do dashboard agora dependem **exclusivamente** do JWT para identificação
- Certifique-se de que o JWT sempre contém `empresaId` válido
- Rotas públicas continuam usando o middleware normalmente

🔍 Arquivos Modificados

1. `src/app.module.ts`
   - Adicionado `.exclude()` para rotas do dashboard e autenticação
   - Comentários explicativos sobre o motivo da exclusão

📝 Notas Técnicas

Ordem de Execução no NestJS:

1. **Middlewares** (configurados em `app.module.ts`)
2. **Guards** (ex: `JwtAuthGuard`, `RolesGuard`)
3. **Interceptors**
4. **Pipes**
5. **Controller Method**

Por que Middlewares são Executados Primeiro:

- Middlewares são executados na ordem de registro
- Guards são executados depois dos middlewares
- Por isso, o `TenantMiddleware` estava bloqueando antes do JWT ser validado

Alternativas Consideradas:

1. ❌ **Adicionar header `x-tenant-slug` em todas as requisições do dashboard**
   - Desnecessário, pois o JWT já tem essa informação
   - Aumentaria complexidade no frontend

2. ❌ **Modificar o middleware para não lançar erro em localhost**
   - Não resolve o problema fundamental
   - Apenas mascara o erro

3. ✅ **Excluir rotas autenticadas do middleware** (solução escolhida)
   - Mais limpo e eficiente
   - Segue o princípio de responsabilidade única
   - Cada camada faz apenas o que deve fazer

🚀 Próximos Passos

1. ✅ Implementar correção
2. ✅ Documentar solução
3. ⏳ Testar localmente
4. ⏳ Fazer commit e push
5. ⏳ Criar Pull Request
6. ⏳ Testar em produção

📚 Referências

- NestJS Middleware (https://docs.nestjs.com/middleware)
- NestJS Guards (https://docs.nestjs.com/guards)
- NestJS Request Lifecycle (https://docs.nestjs.com/faq/request-lifecycle)

Data da Correção: 13 de Outubro de 2025
Desenvolvedor: Assistente AI
Status: ✅ Implementado e Documentado