🔧 Correção do Erro de Deploy no Render

📋 Problema Identificado

O erro `Cannot find module '/opt/render/project/src/backend/dist/main'` ocorria porque:

- O NestJS compila os arquivos TypeScript **preservando a estrutura de pastas** do `src/`
- O arquivo compilado fica em: `dist/src/main.js`
- O comando `start:prod` estava tentando executar: `node dist/main`
- Isso causava o erro porque o Node buscava `dist/main.js` (que não existe)

✅ Solução Aplicada

Alteramos o comando `start:prod` no `package.json`:

Antes:

```
"start:prod": "node dist/main"
```

Depois:

```
"start:prod": "node dist/src/main"
```

🚀 Próximos Passos no Render

Agora que a correção foi feita e enviada para o GitHub, você precisa:

1. Triggar um novo deploy no Render

O Render vai automaticamente:
1. Detectar as mudanças no repositório
2. Executar o build novamente
3. Usar o novo comando `start:prod` corrigido

2. Verificar os Logs

Após o deploy, verifique os logs para confirmar que:
- ✅ O build foi concluído com sucesso
- ✅ As migrations do Prisma rodaram
- ✅ O servidor iniciou sem erros
- ✅ A mensagem “Application is running on: http://[::1]:XXXX” aparece

3. Configurações do Render (já configuradas anteriormente)

Se você precisar reconfigurar:

```
Root Directory: backend
Build Command: npm install && npm run build && npx prisma generate
Start Command: npx prisma migrate deploy && npm run start:prod
```

📊 Estrutura de Pastas do Build

Após o build, a estrutura é:

```
backend/
├── dist/
│   ├── src/
│   │   ├── main.js          ← Arquivo principal compilado
│   │   ├── app.module.js
│   │   └── ...
│   └── prisma/
├── src/
│   ├── main.ts              ← Arquivo TypeScript original
│   └── ...
└── package.json
```

🎯 Por que isso aconteceu?

O NestJS, por padrão, mantém a estrutura de pastas durante a compilação porque:
- Facilita o debugging (source maps correspondem aos arquivos originais)
- Mantém organização do código
- Permite imports relativos consistentes

💡 Alternativas Consideradas

Poderíamos também ter configurado o `nest-cli.json` para compilar direto em `dist/`:

```
{
  "$schema": "https://json.schemastore.org/nest-cli",
  "collection": "@nestjs/schematics",
  "sourceRoot": "src",
  "compilerOptions": {
    "deleteOutDir": true,
    "outDir": "dist",
    "assets": ["**/*.json", "**/*.prisma"]
  }
}
```

Mas isso exigiria mais mudanças e poderia afetar imports. A solução atual é **mais simples e segura**.

🐛 Como Detectamos o Problema

1. ✅ Clonamos o repositório localmente
2. ✅ Analisamos o `package.json`, `nest-cli.json` e `tsconfig.json`
3. ✅ Executamos `npm run build` localmente
4. ✅ Verificamos que o arquivo estava em `dist/src/main.js` e não em `dist/main.js`
5. ✅ Corrigimos o comando `start:prod`

6. ✅ Testamos localmente para confirmar
7. ✅ Fizemos commit e push das alterações

Status: ✅ Correção aplicada e enviada para o GitHub (commit: 23b13d6)

Próxima ação: Aguardar o deploy automático do Render ou fazer deploy manual