**📑 Análise de Requisitos – Sistema MEDSYNC**

**1. Descrição Geral do Sistema**

O **MEDSYNC** é um sistema para gerenciamento e preenchimento de **laudos hospitalares**, composto por duas plataformas:

* **Web (React + JavaScript + SQL Server)**: voltada para administradores e faturistas do hospital.
* **Aplicativo Mobile (React Native)**: voltado para médicos e administradores.

O sistema permite cadastro de usuários (somente administradores têm essa permissão), controle de acessos, notificações para médicos, preenchimento de laudos digitais e encaminhamento automático para faturistas. Além disso, terá integração com o software hospitalar via **API**.

**2. Requisitos Funcionais (RF)**

1. **RF01 – Login e Autenticação**
   * O sistema deve permitir login com usuário e senha, cadastrados previamente pelo administrador.
2. **RF02 – Cadastro de Usuários**
   * Apenas administradores podem cadastrar novos usuários (médicos, faturistas, outros administradores).
3. **RF03 – Cadastro de Pacientes**
   * O sistema deve permitir cadastrar, editar e consultar informações de pacientes.
4. **RF04 – Criação de Atendimentos**
   * O administrador/faturista pode criar atendimentos vinculados a pacientes.
5. **RF05 – Notificação ao Médico**
   * Após a criação de um atendimento, o sistema deve enviar notificação ao médico no aplicativo.
6. **RF06 – Preenchimento de Laudos Médicos (atualizado):**  
   O médico deve preencher laudos selecionando opções pré-definidas (ex.: diagnósticos, sintomas, observações clínicas), podendo adicionar informações complementares em campo de texto livre, se necessário.
7. **RF07 – Encaminhamento Automático**
   * Após preenchimento do laudo, o sistema deve disponibilizar o documento aos faturistas.
8. **RF08 – Edição e Impressão de Laudos**
   * Os faturistas devem poder revisar, editar e imprimir laudos médicos.
9. **RF09 – Relatórios Gerenciais**
   * O sistema deve gerar relatórios de atendimentos e laudos para administradores.
10. **RF10 – Integração via API**

O sistema deve se integrar ao software hospitalar para troca de dados (ex.: pacientes, atendimentos).  
  
 **RF11 – Cadastro de Opções de Laudos:**  
O sistema deve permitir que **administradores médicos** cadastrem, editem e removam as opções pré-definidas que aparecerão nos laudos (ex.: sintomas, diagnósticos, observações).

 **RF12 – Atualização Dinâmica dos Laudos:**  
O sistema deve permitir que as opções de preenchimento dos laudos sejam atualizadas sem necessidade de modificar o código-fonte (ex.: via banco de dados).

 **RF13 – Chat Interno:**  
O sistema deve disponibilizar uma aba de chat no aplicativo e na versão web, permitindo a comunicação em tempo real entre médicos, administradores e faturistas.

 **RF14 – Histórico de Conversas:**  
O sistema deve armazenar o histórico das conversas realizadas no chat, com identificação de participantes e data/hora das mensagens.

 **RF15 – Dashboard Administrativo:**  
O sistema deve apresentar ao administrador um painel com métricas, como:

* número de pacientes cadastrados;
* número de atendimentos em andamento;
* quantidade de laudos preenchidos;
* estatísticas de uso (ex.: laudos por médico).

 **RF16 – Filtros e Relatórios do Dashboard:**  
O dashboard deve permitir aplicar filtros por período, médico e status do atendimento.

**3. Requisitos Não Funcionais (RNF)**

1. **RNF01 – Usabilidade**: A interface deve ser intuitiva e responsiva (Web e Mobile).
2. **RNF02 – Desempenho**: O sistema deve suportar múltiplos acessos simultâneos sem queda de performance.
3. **RNF03 – Segurança**:
   * Senhas devem ser armazenadas criptografadas.
   * Acesso controlado por níveis de permissão.
   * Implementar HTTPS e autenticação JWT para API.
4. **RNF04 – Portabilidade**: O aplicativo deve estar disponível para Android e iOS.
5. **RNF05 – Disponibilidade**: O sistema deve estar disponível 24/7, com tempo de indisponibilidade mínimo.
6. **RNF06 – Compatibilidade**: O sistema deve ser compatível com navegadores modernos (Chrome, Edge, Firefox).

**4. Regras de Negócio (RN)**

**RN01** – Apenas administradores podem cadastrar usuários no sistema.

**RN02** – Um laudo só pode ser encaminhado ao faturista após ser preenchido e assinado digitalmente pelo médico.

**RN03** – O sistema deve notificar médicos apenas de atendimentos vinculados a eles.

**RN04** – Os faturistas só podem editar campos administrativos do laudo (ex.: código de cobrança, observações), não o conteúdo médico.

**RN05** – Cada paciente deve ter um histórico de laudos acessível apenas por usuários autorizados.  
  
**RN06 – Padronização de Diagnósticos e Sintomas:**  
O médico deve escolher os sintomas e diagnósticos a partir de listas pré-definidas cadastradas pelo administrador.

**RN07 – Flexibilidade no Preenchimento:**  
O médico poderá adicionar observações em campo de texto livre, mas os **diagnósticos principais** devem ser escolhidos da lista padronizada.

**RN08 – Versionamento das Listas de Laudos:**  
Caso uma lista de opções seja alterada pelo administrador, os laudos já preenchidos não devem ser modificados retroativamente, preservando o histórico.

**RN09 – Restrição de Acesso ao Chat:**  
Apenas usuários autenticados podem utilizar o chat, sendo que mensagens são visíveis apenas para os participantes da conversa.

**RN10 – Sigilo de Informações no Chat:**  
Mensagens enviadas no chat não podem conter dados sensíveis de pacientes (dados pessoais, diagnósticos detalhados), devendo ser usadas apenas para comunicação administrativa.

**RN11 – Acesso ao Dashboard:**  
Apenas administradores têm acesso ao dashboard completo. Faturistas podem ter acesso restrito a relatórios financeiros e de laudos.

**5. Requisitos de Sistema (Tecnológicos)**

1. **Front-end Web**: React.js + JavaScript.
2. **Mobile**: React Native.
3. **Banco de Dados**: SQL Server.
4. **API**: Node.js (Express) ou ASP.NET Core (boa integração com SQL Server).
5. **Autenticação**: JWT + bcrypt (para criptografia de senhas).
6. **Infraestrutura**: Hospedagem em servidor cloud (Azure ou AWS recomendados).
7. **Controle de Versão**: GitHub/GitLab.
8. **Notificações Mobile**: Firebase Cloud Messaging (FCM).
9. **Chat:** Pode ser implementado usando **WebSockets** (via **Socket.IO** no Node.js ou SignalR no .NET) para comunicação em tempo real.
10. **Dashboard:** Pode ser construído em **React** usando bibliotecas como **Recharts**, **Chart.js** ou **D3.js** para exibição de gráficos.
11. **Banco de Dados:** Tabelas adicionais para mensagens do chat e logs de comunicação.

### Tabelas – MEDSYNC

**Usuarios**

* id\_usuario (PK)
* nome
* email
* senha\_hash
* perfil
* status
* data\_criacao

**Pacientes**

* id\_paciente (PK)
* nome
* data\_nascimento
* cpf
* telefone
* endereco
* data\_cadastro

**Atendimentos**

* id\_atendimento (PK)
* id\_paciente (FK)
* id\_medico (FK)
* data\_atendimento
* status

**Laudos**

* id\_laudo (PK)
* id\_atendimento (FK)
* id\_medico (FK)
* observacoes
* data\_preenchimento
* status

**Sintomas**

* id\_sintoma (PK)
* descricao

**Diagnosticos**

* id\_diagnostico (PK)
* descricao

**Laudo\_Sintomas**

* id\_laudo (FK)
* id\_sintoma (FK)

**Laudo\_Diagnosticos**

* id\_laudo (FK)
* id\_diagnostico (FK)

**Mensagens\_Chat**

* id\_mensagem (PK)
* id\_remetente (FK)
* id\_destinatario (FK)
* conteudo
* data\_envio

**Logs\_Acoes**

* id\_log (PK)
* id\_usuario (FK)
* acao
* data\_hora