Plano de Implementação Detalhado

Aqui está um plano para implementar as melhorias discutidas, dividido em fases. Marcarei o progresso à medida que avançamos.

Fase 1: Backend e Banco de Dados - A Base da Vinculação Inteligente

- [x] 1.1: (DB Schema) Adicionar campo link_manual_necessario (BOOLEAN, DEFAULT FALSE) à tabela EXECUCOES.
- [x] 1.2: (DB Schema) Garantir que ordem_execucao (INTEGER, NULLABLE) exista nas tabelas SESSOES, EXECUCOES e UNIMED_SESSOES_CAPTURADAS.
- [ ] 1.3: (Scraping/Extração) Revisar e ajustar o script todas_as_fases_adaptado.py e o processo de extração de PDF (Claude) para capturar e armazenar ordem_execucao da forma mais confiável possível, populando o campo correspondente em UNIMED_SESSOES_CAPTURADAS e SESSOES. Se a ordem não estiver disponível, o campo deve ficar NULL.
- [x] 1.4: (SQL Function) Refatorar a função inserir_execucao_unimed:

Implementar a lógica de busca por sessao_id em múltiplos níveis de confiança (Exato > Tolerância+Ordem > Tolerância s/ Ordem com Unicidade).

Atualizar codigo_ficha e codigo_ficha_temp corretamente ao vincular.

Definir link_manual_necessario = true se a vinculação falhar ou for ambígua.

Aprimorar os logs em unimed_log_processamento para detalhar o resultado da tentativa de vinculação.

- [x] 1.5: (SQL Function) Refatorar a função batch vincular_sessoes_execucoes:

Dividir em passos de confiança crescente (Exato > Tolerância+Ordem > Tolerância s/ Ordem com Unicidade).

Só atualizar se encontrar correspondência única em cada passo.

Integrar atualização de codigo_ficha e codigo_ficha_temp.

- [x] 1.6: (SQL Function) Refatorar a função batch vincular_sessoes_mesmo_dia:

Vincular automaticamente apenas se as contagens baterem E a ordem_execucao permitir um mapeamento 1-para-1 inequívoco.

Em todos os outros casos de múltiplas sessões/execuções, não vincular, mas sim sinalizar os registros (setar link_manual_necessario = true nas EXECUCOES envolvidas).

- [x] 1.7: (Backend API) Criar/Ajustar Endpoints:
- [x] GET /api/sessoes: Adicionados filtros básicos (busca, datas).
- [x] GET /api/execucoes: Adicionados filtros (busca, datas, guia, paciente(parcial), status_vinculacao, link_manual_necessario).
- [x] POST /api/vinculacoes/manual: Endpoint criado e funcional.
- [x] POST /api/vinculacoes/batch: Endpoint criado e funcional.

Fase 2: Frontend - Interface de Vinculação Manual e Feedback

- [x] 2.1: (API Integration) Conectar a página vinculacoes/page.tsx aos endpoints da API (substituir mocks por chamadas fetch).
- [x] 2.2: (Filtering & Sorting) Aprimorar filtros: adicionado busca textual, intervalo de datas, filtro por status de vinculação e link_manual_necessario. Ordenação básica implementada.
- [x] 2.3: (Linking Logic)
- [x] O botão "Vincular Itens" chama o endpoint POST /api/vinculacoes/manual.
- [x] O botão "Vincular Automaticamente" chama o endpoint POST /api/vinculacoes/batch.
- [x] 2.4: (UI Feedback & Highlighting)

- [x] Indicar visualmente execuções que precisam de vínculo manual (borda/fundo laranja).
- [x] Mostrar status de vinculação de forma clara (badges).
- [x] Ao selecionar uma sessão ou execução, destacar potenciais correspondências (mesma guia) na outra coluna.
- [x] Exibir mensagens de sucesso ou erro após as operações de vinculação (toast).
- [ ] 2.5: (Usability) Avaliar se o Drag-and-Drop é a melhor interação ou se um clique duplo ou botão "Selecionar para Vincular" seria mais eficiente, especialmente com listas longas. Exibir detalhes dos itens selecionados de forma mais clara. (Mantido clique + botão por enquanto).

Fase 3: Testes e Refinamentos

- [ ] 3.1: (Testing) Testar exaustivamente todos os cenários de vinculação: 1-para-1, sem correspondência, múltiplas correspondências (ambíguas), casos com e sem ordem, dados com pequenas variações de data.
- [ ] 3.2: (UAT) Realizar Testes de Aceitação do Usuário focando na clareza da interface de vinculação manual e na eficácia das sugestões/sinalizações.
- [ ] 3.3: (Refinement) Ajustar lógicas, SQL e interface com base nos resultados dos testes e feedback.

Integração com Agendamentos

Fase 4: Backend e DB - Preparando Vinculação com Agendamentos

- [x] 4.1: (DB Schema) Adicionar coluna `agendamento_id UUID NULL REFERENCES agendamentos(id)` à tabela `sessoes`.
- [x] 4.2: (DB Schema) Adicionar coluna `agendamento_id UUID NULL REFERENCES agendamentos(id)` à tabela `execucoes`.
- [x] 4.3: (DB Schema) Adicionar coluna `paciente_id UUID NULL REFERENCES pacientes(id)` à tabela `fichas`.
- [x] 4.4: (DB Schema) Criar índices nas novas colunas `agendamento_id` em `sessoes` e `execucoes`, e em `paciente_id` na tabela `fichas`.
- [x] 4.5: (Backend/Upload) Ajustar `backend/routes/upload.py` para buscar e preencher `fichas.paciente_id` ao processar o PDF.
- [x] 4.6: (Backend/Scraping) Ajustar `Unimed_Scraping/todas_as_fases_adaptado.py` e/ou função `inserir_execucao_unimed` para obter e disponibilizar `paciente_id` durante o processamento da execução (talvez adicionando `paciente_id` à `execucoes`).
- [x] 4.7: (SQL Function) Criar função batch `vincular_agendamentos()` para:
- [x] Buscar `sessoes` sem `agendamento_id` e tentar vincular a `agendamentos` (paciente_id, data, proc_id?).
- [x] Buscar `execucoes` sem `agendamento_id` e tentar vincular a `agendamentos`.
- [x] Vincular apenas se encontrar correspondência única.
- [x] Propagar vínculo entre `sessoes` e `execucoes` se uma delas for vinculada ao agendamento.
- [ ] Opcional: Adicionar flag `agendamento_link_manual_necessario`.
- [x] 4.8: (DB View) Criar a VIEW `vw_agendamentos_com_status_vinculacao` juntando `agendamentos` com `sessoes` e `execucoes` via `agendamento_id` e adicionando colunas de status (`possui_sessao_vinculada`, `possui_execucao_vinculada`, `status_vinculacao`).

Fase 5: Backend API - Endpoints para Integração Agendamentos

- [x] 5.1: (Backend API) Criar endpoint `POST /api/vinculacoes/agendamentos/batch` para chamar a função `vincular_agendamentos()`.
- [x] 5.2: (Backend API) Ajustar endpoint `GET /api/agendamentos` para consultar a VIEW `vw_agendamentos_com_status_vinculacao` e retornar os novos campos de status.

Fase 6: Frontend - Interface de Agendamentos Atualizada

- [x] 6.1: (Frontend/Agendamentos) Atualizar a página para exibir as novas colunas de status (`possui_sessao_vinculada`, `possui_execucao_vinculada`, `status_vinculacao`) usando ícones/badges.
- [x] 6.2: (Frontend/Agendamentos) Adicionar filtros para `status_vinculacao`.
- [ ] 6.3: (Frontend/Agendamentos) Opcional: Adicionar botão "Tentar Vinculação Automática" para chamar o endpoint batch.
- [ ] 6.4: (Frontend/Agendamentos) Opcional: Adicionar links nas linhas para navegar para detalhes da Ficha/Sessão/Execução vinculada.

Fase 7: Testes e Refinamentos - Integração Agendamentos

- [ ] 7.1: (Testing) Testar a vinculação de agendamentos em diversos cenários: agendamento sem ficha/execução, com ambos, com apenas um, etc.
- [ ] 7.2: (Testing) Testar a função batch `vincular_agendamentos()` e a atualização da VIEW.
- [ ] 7.3: (Testing) Testar a exibição e filtros na página de Agendamentos no frontend.
- [ ] 7.4: (UAT) Realizar Testes de Aceitação do Usuário focando na clareza dos status de vinculação na tela de Agendamentos.
- [ ] 7.5: (Refinement) Ajustar lógicas, SQL e interface com base nos resultados dos testes e feedback.