Nome dos Integrantes:

- Erick Gama RM:561951
- Matheus Nascimento RM: 563765
- Bruno Jacob RM: 565249

Nome da Equipe:

- EMB

Nome da Solução (Sistema Desenvolvido):

- Prompt Medic: Um sistema de gerenciamento de consultas médicas otimizado para o Hospital das Clínicas, com foco em acessibilidade para idosos e pessoas com dificuldades.

Sumário

Nome dos Integrantes: 1
Nome da Equipe: 1
Nome da Solução (Sistema Desenvolvido): 1
Sumário 2
2. Objetivo e Escopo do Projeto 3
3. Descrição das Funcionalidades 3
4. Tabela de Endpoints (API Restful) 5
5. Modelo de Entidade-Relacionamento 7
6. Diagrama de Classes 8
7. Protótipo 9
8. Conclusão 11

2. Objetivo e Escopo do Projeto

O projeto PromptMedic é uma aplicação Java desenvolvida especificamente para o Hospital das Clínicas, com o objetivo principal de melhorar o atendimento para consultas médicas, otimizando processos como cadastro, visualização de cadastros e outras funcionalidades relacionadas ao gerenciamento de pacientes, médicos, endereços, consultas e receitas.

O sistema implementa uma API RESTful com CRUD completo (Cadastrar, Consultar, Atualizar e Excluir) para todas as entidades, além de login de pacientes e filtros personalizados. Ele utiliza classes de domínio (Paciente, Médico, Endereço, Consulta, Receita), DTOs para transferência de dados com validações, DAOs para persistência em banco de dados Oracle e endpoints REST via JAX-RS. As melhorias incluem validações automáticas com Bean Validation para evitar erros, mapeamento com ModelMapper e tratamento de exceções personalizado.

3. Descrição das Funcionalidades

As principais funções do sistema são mostradas de forma simples, em uma listinha com explicações rápidas. A ideia é deixar claro o que ele oferece e como isso ajuda quem vai usar, principalmente idosos ou pessoas com mais dificuldades.

- **Cadastro de Pacientes**: Permite inserir novos registros com validação de campos obrigatórios (ex: nome, CPF, email), utilizando DTOs para reduzir digitação e erros.
- **Listagem de Entidades**: Exibe listas de pacientes, médicos, endereços, consultas e receitas, com filtros opcionais por ID ou status.
- **Agendamento e Atualização de Consultas**: Cria consultas vinculando médico e paciente, com status (Marcada, Finalizada, Cancelada). Suporta atualizações com validações para evitar violações de integridade.
- **Busca e Exclusão**: Métodos para buscar por ID e remover registros, com exceção personalizada para entidades não encontradas, promovendo segurança no uso.
- **Login de Paciente**: Autenticação via CPF + Email, retornando detalhes do paciente.
- **Filtros Específicos**: Listar consultas por paciente ou receitas por consulta.

- **Integração com Banco de Dados**: Persistência via JDBC/DAO em Oracle, garantindo dados confiáveis.

Cada funcionalidade é acessível via endpoints REST, garantindo usabilidade simples e acessível. Por exemplo, o consumidor (Postman ou frontend) envia JSON e recebe feedback imediato sobre sucessos ou erros, com mensagens claras.

4. Tabela de Endpoints (API Restful)

/pacientes/criar, POST, "201, 400"

/pacientes/listar, GET, 200

/pacientes/buscar/{id}, GET, "200, 404"

/pacientes/atualizar/{id}, PUT, "200, 400, 404"

/pacientes/deletar/{id}, DELETE, "204, 404"

/medicos/criar, POST, "201, 400"

/medicos/listar, GET, 200

/medicos/buscar/{id}, GET, "200, 404"

/medicos/atualizar/{id}, PUT, "200, 400, 404"

/medicos/deletar/{id}, DELETE, "204, 404"

/enderecos/criar, POST, 201

/enderecos/listar, GET, 200

/enderecos/buscar/{id}, GET, "200, 404"

/enderecos/atualizar/{id}, PUT, "200, 400, 404"

/enderecos/deletar/{id}, DELETE, "204, 404"

/consultas/criar, POST, 201

/consultas/listar, GET, 200

/consultas/buscar/{id}, GET, "200, 404"

/consultas/atualizar, PUT, "200, 404"

/consultas/deletar/{id}, DELETE, "204, 404"

/consultas/consultaPaciente/{idPaciente}, GET, 200

/receitas/criar, POST, 201

/receitas/listar, GET, 200

/receitas/buscar/{id}, GET, "200, 404"

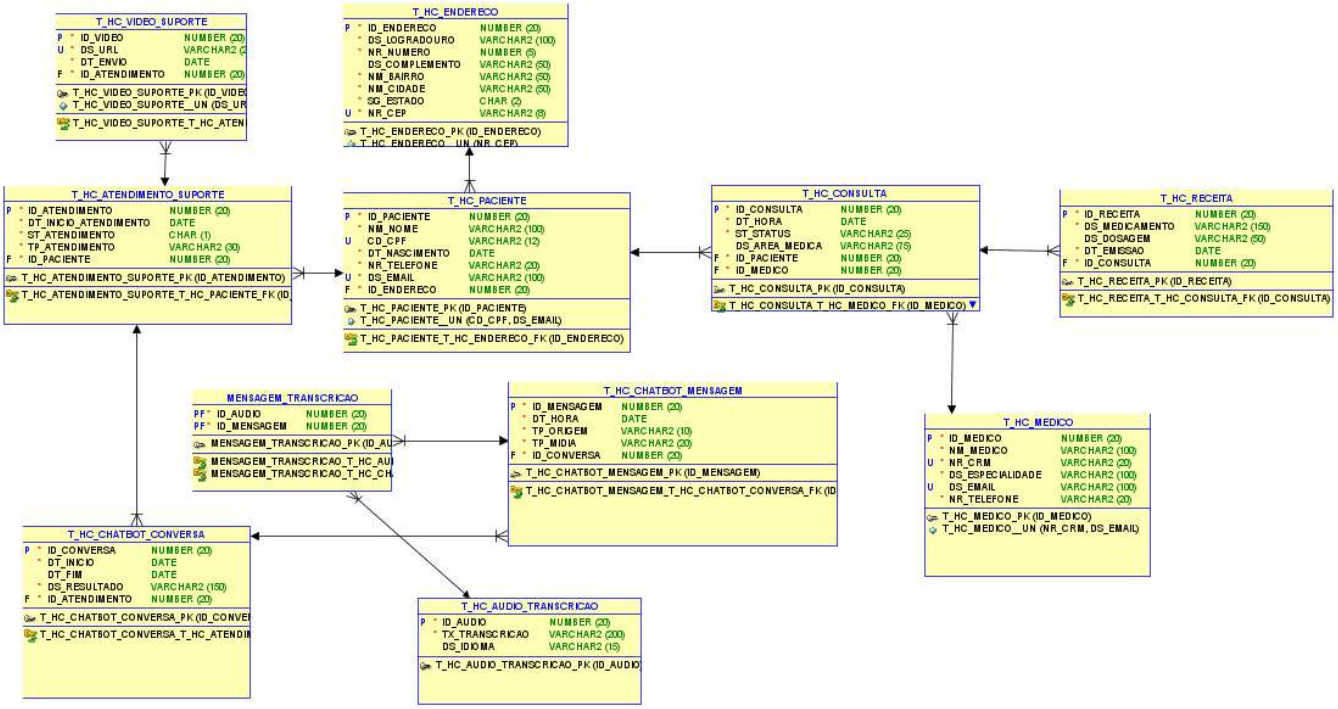
/receitas/atualizar/{id}, PUT, "200, 400, 404"

/receitas/deletar/{id}, DELETE, "204, 404"

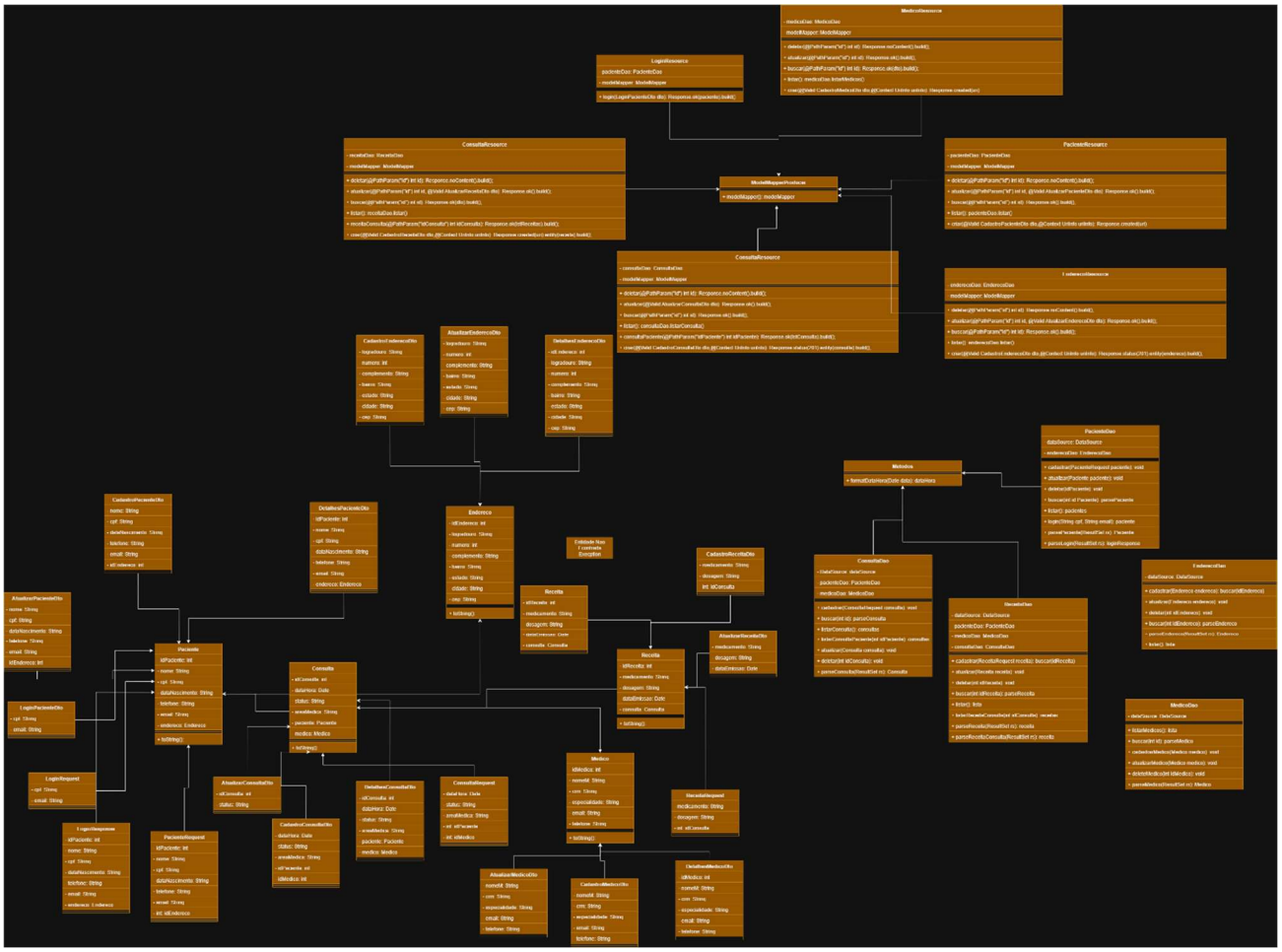
/receitas/receitaConsulta/{idConsulta}, GET, 200

/login, POST, "200, 401"

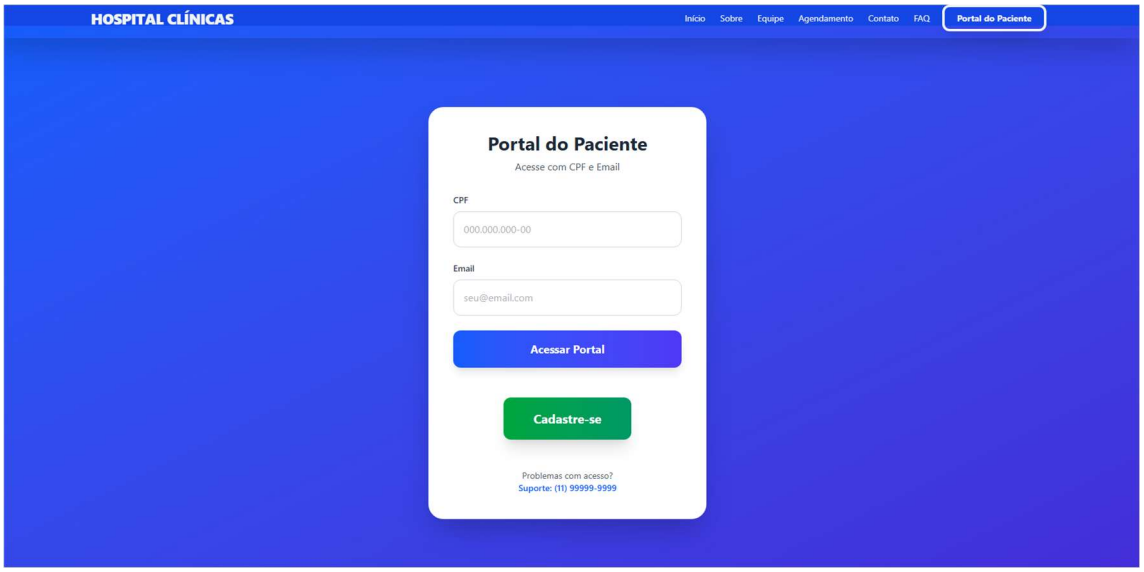
5. Modelo de Entidade-Relacionamento



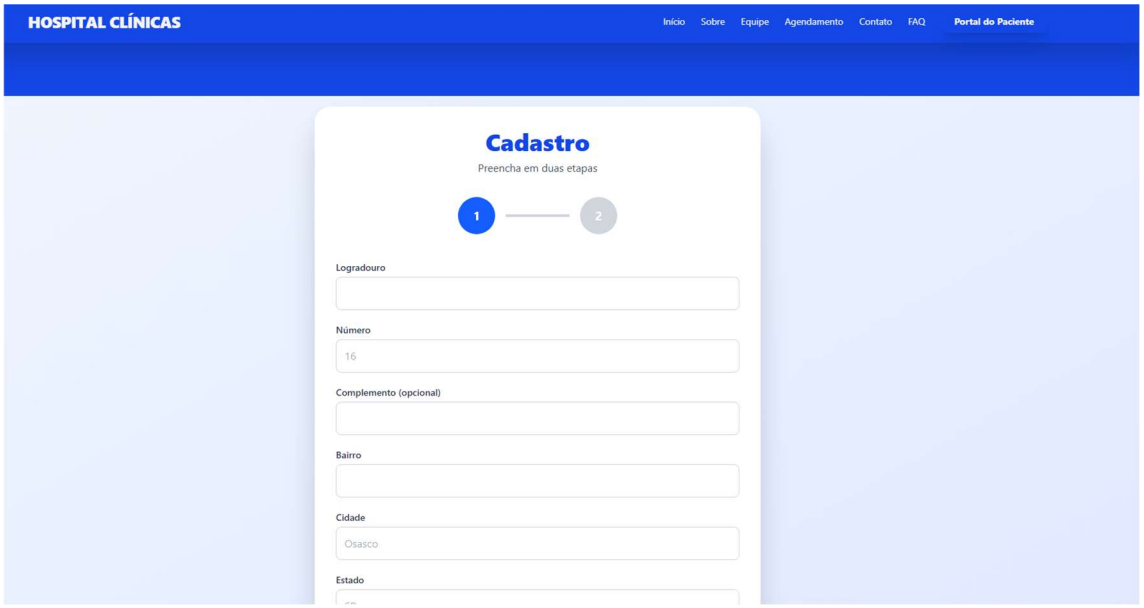
6. Diagrama de Classes



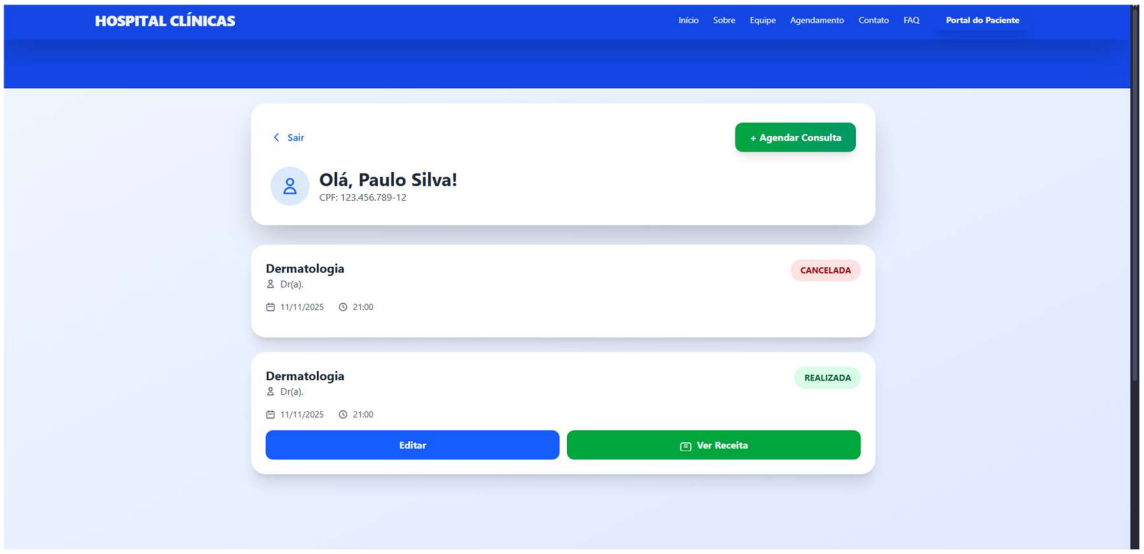
7. Protótipo



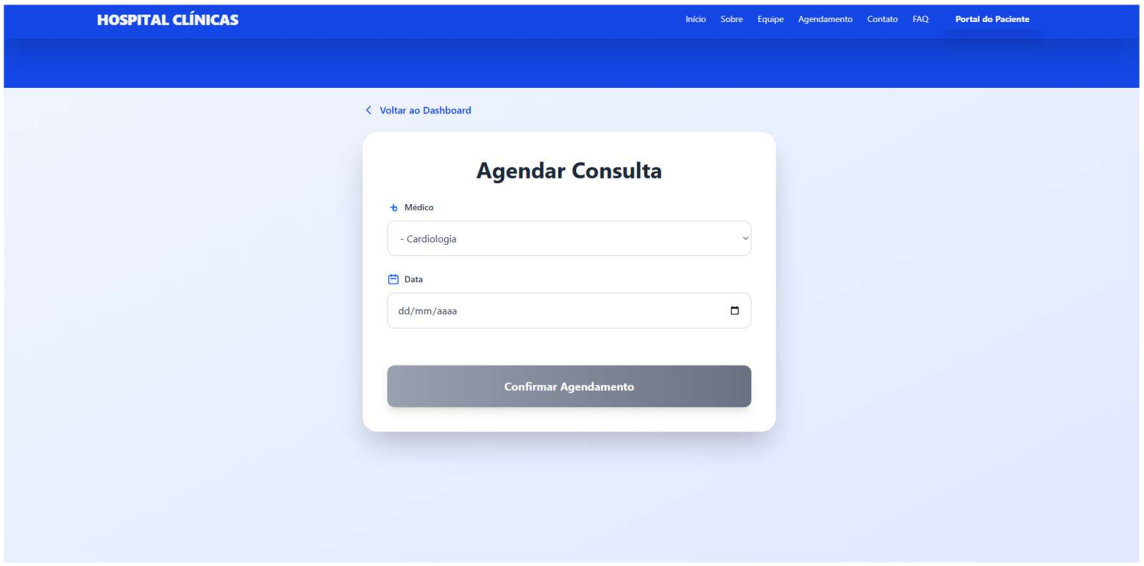
Essa seria a parte onde os Pacientes fariam login utilizando seus CPF e EMAIL



Essa seria onde os Pacientes fariam o cadastro caso não tenha algum login.



Essa seria a parte que mostra as consultas após o paciente realizar o login



Essa seria a pagina para agendar a consulta para o paciente que fez o login

Essa pagina seria onde fica a receita, após terminar uma consulta

Todas as paginas mostradas tem acesso as API que foram desenvolvidas no Java.

8. Conclusão

O **PromptMedic** mostrou na prática como dá pra aplicar POO e persistência em Java de um jeito organizado e funcional. O sistema pensa em atender ao que foi proposto para o Hospital das Clínicas, pensando na acessibilidade para idosos e pessoas com dificuldades. Os pontos mais fortes ficam por conta das validações (que ajudam a manter a integridade dos dados) e da estrutura modular com DAOs, que deixa o código mais limpo e fácil de manter.