CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA

FACULDADE DE TECNOLOGIA DE INDAIATUBA

DR. ARCHIMEDES LAMMOGLIA

BRUNA DE NOVAIS SCHULTZ

CAIO FRANSON DA SILVA

CAMILA BARCARO CORRÊA

PEDRO DE OLIVEIRA SILVA

VITÓRIA CAROLINE DA SILVA FARIAS FATURI

**Projeto integrador - 2º semestre: solução acadêmica para modernização de processos em cartórios**

Indaiatuba, junho de 2025

**Sumário**

[**1.** **ENGENHARIA DE SOFTWARE I** 3](#_Toc200455705)

[**1.1. Extração de requisitos** 3](#_Toc200455706)

[**1.2. Análise de riscos** 4](#_Toc200455707)

[**1.3. Projeto** 4](#_Toc200455708)

[**1.4. *Function Point*** 7](#_Toc200455709)

[**1.4.1. Tabela de Contagem dos Itens** 7](#_Toc200455710)

[**1.4.2 Fator de Ajuste (FA)** 8](#_Toc200455711)

[**1.4.3 Cálculo Final dos *Function Points*** 8](#_Toc200455712)

[**1.4.4 Custo Estimado** 8](#_Toc200455713)

[**1.5. Codificação** 8](#_Toc200455714)

[**1.5.1. Estrutura do Projeto** 9](#_Toc200455715)

[**1.5.2. Funcionalidades Implementadas** 9](#_Toc200455716)

[**1.5.3. Boas Práticas** 9](#_Toc200455717)

[**1.5.4. Testes** 10](#_Toc200455718)

[**1.5.5. Qualidade do Sistema (PDCA)** 10](#_Toc200455719)

[**1.5.6. Manutenção** 11](#_Toc200455720)

[**1.5.6.1. Principais dificuldades identificadas:** 12](#_Toc200455721)

[**1.5.6.2. Ações adotadas e planejadas para amenizar essas dificuldades:** 12](#_Toc200455722)

[**1.5.6.3. Indicadores para acompanhar a manutenção:** 12](#_Toc200455723)

[**1.5.6.4. Facilidade para atualizações futuras:** 12](#_Toc200455724)

[**2.** **SISTEMAS DE INFORMAÇÃO** 12](#_Toc200455725)

[**2.1. Conceitos e Componentes de SI** 13](#_Toc200455726)

[**2.3. Modelagem e Banco de Dados** 13](#_Toc200455727)

[**2.4. Dados, Informação e Conhecimento** 14](#_Toc200455728)

[**2.5. PDCA e melhoria contínua** 15](#_Toc200455729)

[**2.6. Papel estratégico dos SI** 15](#_Toc200455730)

[**2.7. Modelagem de Dados – Diagrama Entidade-Relacionamento (DER)** 15](#_Toc200455731)

[**2.8. Tabelas de apoio:** 16](#_Toc200455732)

[**2.9. Relacionamentos:** 16](#_Toc200455733)

[**3.** **CONCLUSÃO** 16](#_Toc200455734)

[**4.** **ANEXO** 17](#_Toc200455735)

[**ANEXO A – Roteiro flexível planejado para a entrevista com o responsável pela área de TI do cartório** 17](#_Toc200455736)

[**ANEXO B – Relatório da entrevista com o responsável pela área de TI do cartório** 18](#_Toc200455737)

[**ANEXO C – Código SQL do banco de dados do projeto** 19](#_Toc200455738)

[**5.** **REFERÊNCIAS** 24](#_Toc200455739)

# **ENGENHARIA DE SOFTWARE I**

A disciplina de Engenharia de Software I tem como objetivo “Aplicar os princípios e conceitos da Engenharia de Software na implementação do componente software, como parte dos Sistemas de Informação e iniciar a modelagem de software (requisitos)”(CENTRO, [s. d.]).

Assim, os alunos que cursam essa matéria adquirem conhecimento sobre técnicas de extração de requisitos, que, segundo Tiago de Melo (2009), é “o processo de transformação das ideias que estão na mente dos usuários (a entrada) em um documento formal (saída)”.

## **1.1. Extração de requisitos**

A extração de requisitos é uma etapa fundamental no processo de desenvolvimento de software. Segundo Tiago de Melo, trata-se da transformação das ideias que estão na mente dos usuários (entrada) em um documento formal (saída). Dentre as diversas técnicas informais existentes, como JAD, PIECES, brainstorming e questionários, optamos pela entrevista, por se adequar melhor à natureza do nosso projeto.

Essa técnica consiste em um conjunto de encontros com os usuários, nos quais eles relatam seu ambiente de trabalho, suas atividades e suas necessidades em relação ao sistema. Ela exige mais habilidades sociais do que técnicas, especialmente nas fases iniciais, em que o objetivo é compreender o contexto geral e identificar os pontos críticos do sistema.

A entrevista seguiu as seguintes fases:

* **Fase 1 – Identificação dos candidatos:** Foi selecionado como principal entrevistado o responsável pelo setor de TI do cartório, por seu conhecimento abrangente sobre o sistema atual e sua vivência com as demandas de diferentes setores da empresa.
* **Fase 2 – Preparação:** A entrevista foi agendada com antecedência e conduzida com base em um roteiro flexível (Anexo A), permitindo a adaptação conforme novas informações surgiam durante a conversa.
* **Fase 3 – Condução:** O encontro ocorreu no dia 24 de março com Vitor Hugo Oliveira, responsável pelo setor de TI do Cartório Oficial de Registro de Imóveis e Anexos de Indaiatuba/SP. A entrevista foi conduzida por Bruna de Novais Schultz, Camila Barcaro Corrêa e Vitória Caroline da Silva Farias Faturi, e teve como foco compreender o funcionamento do sistema de Títulos, Documentos e Pessoas Jurídicas.
* **Fase 4 – Finalização e registro:** Ao final da conversa, foi elaborado um relatório detalhado, que pode ser consultado no Anexo B deste trabalho.

Durante a entrevista, foram relatados diversos desafios, como a sobreposição de dados entre janelas abertas, a impossibilidade de atualizar ou corrigir o sistema descontinuado e a ausência de capacitação formal dos funcionários. Ao mesmo tempo, aspectos positivos como o suporte técnico ágil e a infraestrutura de backup foram destacados.

A entrevista se mostrou uma técnica eficiente, pois permitiu compreender o cenário real de uso do sistema e identificar os requisitos necessários com clareza. Seguiremos com entrevistas contínuas, conforme indicado na disciplina, a fim de validar e complementar as informações obtidas.

## **1.2. Análise de riscos**

Durante o desenvolvimento do Projeto Integrador, foi realizada uma análise de riscos com base nos conceitos aprendidos na disciplina de Engenharia de Software. Essa análise tem como objetivo prever possíveis obstáculos que possam comprometer o sucesso do projeto e elaborar estratégias de prevenção e mitigação.

Os principais riscos identificados foram:

* **Tempo:** O cronograma limitado é um fator de risco relevante. Para garantir a entrega dentro do prazo, optamos por focar inicialmente nas funcionalidades essenciais do sistema, como cadastro, consultas, login, atualização e deleção de dados. Essa priorização visa garantir um produto funcional mesmo que outras partes precisem ser postergadas.
* **Desconhecimento técnico:** Como somos estudantes em formação, há limitações técnicas que podem impactar o andamento do projeto. Para mitigar esse risco, nos comprometemos a buscar apoio constante do professor e a utilizar materiais de referência confiáveis. A divisão clara de tarefas entre os membros da equipe também é uma medida preventiva.
* **Legislação:** A Lei Geral de Proteção de Dados (LGPD) representa um desafio, pois exige cuidados específicos com o tratamento de dados pessoais. Além disso, o desenvolvimento de funções obrigatórias no contexto de um cartório deve respeitar exigências legais, mesmo que o projeto seja simulado. Isso requer pesquisa e atenção às normas em vigor.
* **Comunicação com o cliente:** Embora o projeto tenha um cliente fictício, as entrevistas com o responsável pelo setor de TI do cartório foram fundamentais para o levantamento de requisitos. A continuidade dessas entrevistas é essencial para esclarecer dúvidas, validar soluções e ajustar o escopo. A interrupção nesse contato representa um risco que pode afetar a aderência do sistema às necessidades reais.
* **Riscos financeiros e de infraestrutura (complementar):** Mesmo não sendo um projeto comercial, simulamos um ambiente real. Em contextos reais, atrasos, custos adicionais e falhas de infraestrutura (como indisponibilidade de internet ou máquinas inadequadas) são riscos importantes. No nosso caso, usamos recursos educacionais gratuitos, mas limitações de ferramentas também foram consideradas.

Com base nessa análise, será elaborado um plano de contingência, definindo estratégias para minimizar os impactos caso os riscos se concretizem.

## **1.3. Projeto**

Após a fase de levantamento de requisitos, o grupo decidiu iniciar o desenvolvimento do sistema com foco nas operações básicas de CRUD (Create, Read, Update, Delete), a fim de garantir que as funcionalidades essenciais estivessem implementadas de forma funcional e segura.

A escolha pelo modelo CRUD se deu pela necessidade de permitir aos usuários do cartório realizar, com facilidade, o cadastro, consulta, edição e exclusão de informações relacionadas a protocolos, títulos, registros, dados de apresentantes, entre outros. Essa estrutura cobre as principais rotinas operacionais do cartório, promovendo a organização e integridade das informações.

Para orientar a construção do sistema, desenvolvemos telas de interface inspiradas nas rotinas diárias do cartório, simulando os principais módulos de consulta e interação com o sistema. Essas telas, anexadas a este trabalho, serviram de base para identificar as entidades, os campos e os fluxos de navegação do sistema. Elas incluem funcionalidades como:

* Registro de usuários;

Interface gráfica do usuário, Aplicativo, Teams

O conteúdo gerado por IA pode estar incorreto.

Figura 1 – Tela de cadastro de usuários

Fonte: Elaborado pelos autores (2025)

* Tela de login com autenticação de usuário;

Interface gráfica do usuário, Aplicativo

O conteúdo gerado por IA pode estar incorreto.

Figura 2 – Tela de login

Fonte: Elaborado pelos autores (2025)

* Tela de consulta de protocolos com filtros detalhados;

Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

O conteúdo gerado por IA pode estar incorreto.

Figura 3 – Tela de pesquisa de protocolo

Fonte: Elaborado pelos autores (2025)

* Seções específicas para módulos como Certificados, Notas de devolução, Registros, Andamento de Títulos, entre outros;
* Campos para visualização e edição de dados, como tipo de contato, número de protocolo, dados do apresentante e documentos anexos.

Com base nessas telas, foi possível estruturar o banco de dados inicial e planejar o desenvolvimento dos módulos com clareza. Essa abordagem prática contribuiu para alinhar as expectativas do cliente com o produto que está sendo desenvolvido, além de facilitar a identificação de melhorias e funcionalidades futuras.

## **1.4. *Function Point***

A contagem de Pontos de Função (*function points* – FP) foi utilizada para estimar o esforço de desenvolvimento do sistema com base em sua complexidade funcional, independentemente da tecnologia adotada. Essa técnica permite medir o tamanho funcional do software considerando as funcionalidades efetivamente entregues ao usuário.

### **1.4.1. Tabela de Contagem dos Itens**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Item** | **Detalhamento dos processos** | **Complexidade** | **Quantidade** | **Subtotal** |
| Entradas do usuário | * Formulários de cadastro de usuários (4) * Protocolos (4) * Grupos e natureza de título (3) * Parte (3) * Apresentante (3) | Médio | 6 | 102 |
| Saídas do usuário | * Emissão de relatórios de protocolos (4) * Visualização de dados em tela com filtros aplicados (3) * Impressão de protocolos (2) | Médio | 4 | 36 |
| Consultas do usuário | Filtros para busca de protocolos por:   * grupo (3); * natureza (2); ou * data (2) | Médio | 3 | 21 |
| Arquivos internos | Armazenamento e manipulação de dados de:   * protocolos (4); e * tipos de documentos (3) | Médio | 3 | 21 |
| Interfaces externas | * Integração futura com sistemas de autenticação ou cartórios externos (3) | Baixa | 1 | 3 |
| **Total de Pontos Não Ajustados** |  |  |  | 183 |

### **1.4.2 Fator de Ajuste (FA)**

Ajustamos a contagem com base em 14 questões de complexidade, avaliadas com notas de 0 a 5. As perguntas e suas respectivas notas atribuídas foram:

* O sistema requer backup e recuperação confiável? → 4
* São exigidas comunicações de dados? → 3
* Há funções de processamento distribuídas? → 2
* O desempenho é crítico? → 4
* O sistema funcionará em um ambiente operacional existente, intensivamente utilizado? → 4
* O sistema requer entrada de dados on-line? → 4
* A entrada de dados on-line exige que a transação seja elaborada em múltiplas telas ou operações? → 3
* Os arquivos mestres são atualizados on-line? → 3
* A entrada, saída, arquivos ou consultas são complexos? → 4
* O processamento interno é complexo? → 3
* O código é projetado de forma a ser reutilizável? → 5
* A conversão e instalação estão incluídas no projeto? → 5
* O sistema é projetado para múltiplas instalações em diferentes organizações? → 2
* A aplicação é projetada de forma a facilitar mudanças e o uso pelo usuário? → 3

Soma(Fi) = 55

Fator de Ajuste (FA) = 0,65 + (0,01 × 55) = 1,20

### **1.4.3 Cálculo Final dos *Function Points***

Com base nos 183 pontos não ajustados e no fator de ajuste calculado:

FP = 183 × 1,20 = 219,6 ≈ 220

### **1.4.4 Custo Estimado**

Considerando o valor de R$ 250,00 por ponto de função, o custo estimado para o desenvolvimento do sistema é:

220 × R$ 250,00 = R$ 55.000,00

Esse valor representa uma simulação de orçamento com base em métricas amplamente utilizadas na engenharia de software, considerando tanto a complexidade funcional quanto os fatores de impacto no desenvolvimento e manutenção da aplicação.

## **1.5. Codificação**

A etapa de codificação do sistema foi desenvolvida utilizando o framework Laravel, com foco na implementação das funcionalidades básicas do cartório por meio de operações CRUD. O código foi organizado de maneira modular, seguindo os padrões MVC (*Model-View-Controller*), o que facilita a manutenção, o reuso e a escalabilidade do sistema.

### **1.5.1. Estrutura do Projeto**

* **Controllers:** Localizados em app/Http/Controllers, os controladores lidam com a lógica de negócio e fazem a ponte entre os modelos e as views. Foram implementados controladores específicos para entidades como ProtocoloController, DocumentoController, ParteController, TipoAndamentoController, entre outros. Esses controladores são responsáveis por operações como criação, listagem, edição e exclusão de registros.
* **Models:** Os modelos estão organizados em app/Models e representam as entidades principais do sistema, como Protocolo, Documento, Parte, Apresentante, Natureza, Grupo, entre outros. Eles interagem diretamente com o banco de dados e definem os relacionamentos entre as tabelas.
* **Views (Blade):** As views estão localizadas em resources/views, utilizando a engine Blade para renderização das páginas. Foram criadas interfaces amigáveis e intuitivas para cada módulo do sistema, como:
  + autenticacao/index.blade.php para login;
  + protocolos/create.blade.php, index.blade.php e view.blade.php para o gerenciamento de protocolos;
  + andamento/index.blade.php para o andamento de títulos;
  + auth/login.blade.php, register.blade.php, reset-password.blade.php para autenticação e recuperação de senha.
* **Componentes reutilizáveis:** Foram criados diversos componentes Blade personalizados em resources/views/components, como input-label, input-select, modal, primary-button, entre outros. Isso garante padronização visual e facilita a construção de novas interfaces.
* **Rotas:** As rotas estão definidas nos arquivos routes/web.php e routes/auth.php, organizando os acessos às funcionalidades do sistema conforme o tipo de usuário e a necessidade de autenticação.

### **1.5.2. Funcionalidades Implementadas**

Até o momento, o sistema conta com:

* Cadastro e visualização de protocolos com formulários e filtros;
* Gerenciamento de apresentantes, documentos, partes, espécies, natureza e grupos;
* Controle de tipos de andamento e tipos de parte;
* Páginas de navegação e layout padronizado, incluindo barra de navegação, botões reutilizáveis e feedback de ações.

### **1.5.3. Boas Práticas**

Durante a codificação, foram adotadas boas práticas como:

* Separação de responsabilidades (princípio SRP);
* Uso de *migrations* e *seeders* para controle do banco;
* Componentização e reutilização de trechos de interface;
* Comentários e nomes descritivos para facilitar a leitura do código.

Essa base sólida permitirá a ampliação futura do sistema, com a adição de relatórios, integrações externas e módulos avançados como envio de notificações e assinatura digital.

### **1.5.4. Testes**

A fase de testes do sistema foi planejada de acordo com os princípios abordados na disciplina, visando garantir a funcionalidade, a segurança e a usabilidade da aplicação.

Foram aplicadas as seguintes abordagens:

* **Testes de Unidade:** Realizados diretamente nos métodos dos controladores e modelos, garantindo que operações como cadastro, atualização e exclusão estejam funcionando corretamente de forma isolada. Métodos como store(), update() e destroy() seriam verificados com diferentes entradas.
* **Testes de Integração:** Executados para validar o funcionamento de fluxos completos do sistema, como o cadastro de um novo protocolo com associação a um apresentante e posterior consulta. Esses testes asseguram a comunicação correta entre as diferentes camadas (*Model-View-Controller*).
* **Testes de Caixa Preta:** Utilizados para simular a interação do usuário final com o sistema, testando campos obrigatórios, mensagens de erro, autenticação, validação de formulários e comportamento de botões nas *views* Blade.
* **Testes de Caixa Branca (estruturais):** A lógica interna dos controladores seria revisada manualmente para confirmar o uso de estruturas condicionais corretas, validação de dados, redirecionamentos e consistência de retornos.

Embora os testes foram conduzidos de forma manual durante esta etapa inicial, a estrutura do Laravel permite no futuro a implementação de testes automatizados com ferramentas como PHPUnit, Laravel Dusk e PespPHP.

Além disso, outros testes importantes foram considerados para garantir a qualidade do sistema:

* **Teste de Funcionalidade**: Verifica se as funcionalidades implementadas atendem exatamente aos requisitos definidos no levantamento, como o correto armazenamento e exibição dos protocolos, grupos e apresentantes.
* **Teste de Interface do Usuário**: Avalia se a interface atende às exigências de clareza, usabilidade e organização visual, garantindo uma boa experiência ao usuário.
* **Teste de Segurança**: Garante que dados sensíveis, como login e sessões, não sejam acessados indevidamente, assegurando a integridade do sistema.

### **1.5.5. Qualidade do Sistema (PDCA)**

A qualidade do sistema foi gerida com base no ciclo PDCA (*Plan, Do, Check, Act*), uma metodologia amplamente utilizada para melhoria contínua de processos. Aplicamos esse ciclo em diferentes fases do desenvolvimento para garantir um produto funcional, seguro e alinhado às expectativas do cliente.

* ***Plan* (Planejar)**
  + Realizamos o levantamento de requisitos por meio de entrevistas com o responsável técnico do cartório.
  + Mapeamos os módulos essenciais e definimos um escopo viável com foco inicial nas funcionalidades CRUD.
  + Estabelecemos critérios de segurança, usabilidade e rastreabilidade para guiar o desenvolvimento.
* ***Do* (Executar)**
  + Desenvolvemos o sistema em Laravel, seguindo o padrão MVC.
  + Implementamos funcionalidades de autenticação, cadastro, consulta e visualização de protocolos e documentos.
  + Criamos componentes reutilizáveis e interfaces intuitivas para facilitar o uso por funcionários do cartório.
* ***Check* (Verificar)**
  + Realizamos testes manuais de unidade, integração e aceitação, verificando o correto funcionamento de cada funcionalidade.
  + A equipe simulou cenários reais de uso e identificou oportunidades de melhoria na navegação, validação e segurança.
  + Validamos o sistema com base nas necessidades reais relatadas na entrevista.
* ***Act* (Agir)**
  + Corrigimos inconsistências encontradas nos testes e melhoramos pontos como feedback visual e organização das views.
  + Documentamos o processo de desenvolvimento e definimos sugestões para versões futuras, como a inclusão de relatórios personalizados e envio de notificações automáticas.
  + Planejamos a continuidade do ciclo PDCA para as próximas iterações do sistema, mantendo o foco na melhoria contínua.

Essa abordagem permitiu garantir que o sistema atendesse aos requisitos identificados, com foco na qualidade técnica e na experiência do usuário.

### **1.5.6. Manutenção**

A manutenção do sistema foi considerada desde o início do projeto como uma etapa essencial para garantir sua evolução contínua e a correção eficaz de eventuais falhas. Durante o desenvolvimento, surgiram também reflexões importantes sobre os desafios e estratégias para manter a qualidade do software ao longo do tempo.

#### **1.5.6.1. Principais dificuldades identificadas:**

* A complexidade dos relacionamentos entre entidades exige atenção para evitar impactos colaterais em alterações pontuais.
* A ausência inicial de testes automatizados dificulta a verificação rápida da integridade do sistema após modificações.
* A necessidade de padronização visual nas *views* torna qualquer ajuste de layout sensível à consistência geral da interface.
* O código ainda exige conhecimento específico da equipe atual, o que pode dificultar a manutenção futura por novos membros.

#### **1.5.6.2. Ações adotadas e planejadas para amenizar essas dificuldades:**

* **Código limpo e modular:** A separação entre controladores, modelos e *views* facilita a localização de trechos específicos para manutenção.
* **Componentes *blade* reutilizáveis:** Elementos visuais padronizados como botões, *inputs* e modais foram criados como componentes, permitindo edições centralizadas.
* **Uso de *migrations* e *seeders*:** A versão controlada do banco de dados permite reestruturações seguras.
* **Documentação e comentários explicativos:** Foram incluídas descrições nos arquivos do sistema para apoiar futuras manutenções.

#### **1.5.6.3. Indicadores para acompanhar a manutenção:**

* Número de falhas registradas após atualizações;
* Tempo médio de resolução de erros;
* Cobertura de testes automatizados;
* Frequência de *commits* corretivos;
* Feedback dos usuários sobre usabilidade pós-manutenção.

#### **1.5.6.4. Facilidade para atualizações futuras:**

A arquitetura atual permite a adição de novos módulos – como relatórios personalizados, integração com notificações automáticas e suporte à assinatura digital – sem prejudicar a estrutura já implementada. Essa escalabilidade foi garantida por boas práticas de engenharia e versionamento.

Com base nessa reflexão, foi elaborado um relatório para solicitar oficialmente a realização de ajustes e melhorias pontuais no sistema, com o objetivo de manter sua performance e alinhamento com as necessidades operacionais do cartório (ver Anexo).

# **SISTEMAS DE INFORMAÇÃO**

A disciplina de Sistemas de Informação foi fundamental para o desenvolvimento do nosso Projeto Integrador, pois forneceu a base conceitual e prática para compreender como um sistema pode ser estruturado para apoiar processos organizacionais, tomada de decisão e ganho competitivo.

## **2.1. Conceitos e Componentes de SI**

Aprendemos que um Sistema de Informação é composto por três pilares: tecnologia, pessoas e processos. Aplicamos esse conceito ao longo de todo o projeto:

* **Tecnologia**: Banco de dados relacional para construir a aplicação.
* **Pessoas**: Identificamos os diferentes perfis de usuários do cartório (escreventes, atendentes, responsáveis técnicos) e suas necessidades.
* **Processos**: Mapeamos o fluxo de atendimento do cartório, desde a entrada do protocolo até a finalização, passando por autenticações, pagamentos e geração de relatórios.

## **2.3. Modelagem e Banco de Dados**

A modelagem do banco de dados foi realizada com base nas funcionalidades levantadas durante a análise de requisitos do sistema. O Diagrama Entidade-Relacionamento (DER) apresentado a seguir representa a estrutura lógica do sistema, evidenciando as entidades principais envolvidas nas operações do cartório, seus atributos e os relacionamentos entre elas.

Diagrama, Esquemático

O conteúdo gerado por IA pode estar incorreto.

Figura 4 – DER do projeto elaborado no software Tupiniquim

Fonte: Elaborado pelos autores (2025)

No centro do modelo está a entidade protocolo, que se relaciona com diversas outras entidades essenciais, como apresentante, natureza, grupo, usuário, entre outras. Essa estrutura permite o controle eficiente de registros, autenticações, documentos, partes envolvidas e tipos de movimentações, assegurando a integridade e a consistência dos dados.

A modelagem foi desenvolvida utilizando um enfoque relacional, com normalização adequada para evitar redundâncias e garantir a escalabilidade do sistema. O DER serviu como base para a criação das tabelas no banco de dados MySQL.

Diagrama

O conteúdo gerado por IA pode estar incorreto.

Figura 5 – Modelo físico do banco de dados elaborado no MySQL

Fonte: Elaborado pelos autores (2025)

O código SQL completo para criação das tabelas pode ser visualizado no Anexo C.

## **2.4. Dados, Informação e Conhecimento**

Durante o projeto, lidamos com:

* **Dados**: Informações brutas como número de protocolo, CPF, datas, tipos de documentos.
* **Informações**: Exibição organizada de registros para o usuário final com filtros e relatórios.
* **Conhecimento**: Utilização dessas informações para tomada de decisão no cartório, como acompanhamento de andamentos e identificação de falhas ou atrasos.

## **2.5. PDCA e melhoria contínua**

Aplicamos o **ciclo PDCA** para organizar e revisar nosso processo de desenvolvimento, reforçando a ideia de melhoria contínua: planejamos com base em entrevistas, desenvolvemos o sistema, testamos com foco em usabilidade e corrigimos falhas com base no feedback do grupo e do cliente simulado.

## **2.6. Papel estratégico dos SI**

Ao final, entendemos que nosso sistema não é apenas um software, mas um verdadeiro Sistema de Informação, pois automatiza processos, melhora a eficiência do cartório, garante a rastreabilidade das informações e poderá ser expandido futuramente com novas integrações, como assinatura digital ou notificações automatizadas.

## **2.7. Modelagem de Dados – Diagrama Entidade-Relacionamento (DER)**

Para estruturar o banco de dados do sistema desenvolvido para o cartório de Títulos, Documentos e Pessoas Jurídicas, foi elaborado um Diagrama Entidade-Relacionamento (DER) completo, representando todas as entidades, atributos e relacionamentos necessários para atender às operações de negócio identificadas durante a etapa de levantamento de requisitos.

O modelo segue os princípios da normalização de dados e prioriza a integridade referencial entre as tabelas. As principais entidades representadas são:

#### 

* **Usuário**: Armazena dados dos funcionários que utilizam o sistema, incluindo nome, e-mail, senha, setor, entre outros. Está relacionada a várias funcionalidades, como andamento, protocolo e autenticação.
* **Protocolo**: Representa a entrada de documentos no cartório. Contém dados como número, data de abertura, tipo de documento, grupo, espécie, natureza, entre outros.
* **Apresentante**: Pessoa ou entidade responsável pela entrega do documento. Contém informações como nome, e-mail e número de contato.
* **Parte**: Representa as partes envolvidas em um protocolo, vinculadas a um tipo específico definido pela tabela tipo\_parte.
* **Andamento**: Registro de ações e movimentações do protocolo ao longo do processo cartorial.
* **Autenticação**: Registra ações de autenticação de documentos e é vinculada à forma de pagamento.
* **Documento**: Associado ao apresentante, define o tipo de documento apresentado.
* **Pagamento** (entidade implícita via autenticação + forma\_pagamento): Indica que a autenticação envolve um pagamento realizado via determinada forma (dinheiro, PIX ou cheque).

## **2.8. Tabelas de apoio:**

* **Grupo, Espécie e Natureza**: Utilizadas para classificar os documentos/protocolos com base em regras do cartório.
* **Tipo de Parte**: Define se a parte envolvida é requerente, requerido, procurador etc.
* **Tipo de Andamento**: Classifica os tipos de movimentações possíveis durante o fluxo de um protocolo.
* **Forma de Pagamento**: Define as formas permitidas no sistema, podendo ser utilizada tanto por autenticações quanto, futuramente, por outros serviços.

## **2.9. Relacionamentos:**

A modelagem contempla relacionamentos do tipo 1:N e N:1 entre entidades como usuário, protocolo, andamento e parte, garantindo flexibilidade no cadastro e rastreabilidade dos dados.

As entidades seguem o modelo relacional com integridade referencial garantida por chaves estrangeiras.

Esse DER serviu de base para a implementação do banco de dados e estruturação dos modelos Eloquent do Laravel, garantindo coerência entre a modelagem lógica e a camada de aplicação.

# **CONCLUSÃO**

O desenvolvimento do sistema para o Cartório de Títulos, Documentos e Pessoas Jurídicas representou uma aplicação prática e integrada dos conhecimentos adquiridos ao longo do semestre nas disciplinas de Engenharia de Software, Sistemas de Informação e Modelagem de Dados. Desde a extração de requisitos até a implementação em Laravel, o projeto permitiu à equipe vivenciar as etapas fundamentais do ciclo de vida de um software, com foco na qualidade, na usabilidade e na aderência às necessidades do cliente.

O uso de entrevistas como técnica de levantamento de requisitos proporcionou uma compreensão aprofundada do ambiente de negócio, evidenciando problemas reais como limitações do sistema atual, falta de treinamentos e necessidade de melhorias na usabilidade. A partir disso, o grupo planejou e desenvolveu um sistema funcional com operações básicas de CRUD, interface intuitiva e estrutura modular baseada no padrão MVC.

A contagem de Pontos de Função permitiu estimar o esforço de desenvolvimento e simular o custo do projeto, enquanto a aplicação do ciclo PDCA garantiu a gestão da qualidade durante todo o processo. A estrutura adotada facilita a manutenção e expansão do sistema, prevendo funcionalidades futuras como relatórios personalizados, notificações automáticas e integração com certificados digitais.

Concluímos que o projeto alcançou seus objetivos principais: criar uma solução personalizada, segura e escalável para as demandas de um cartório, ao mesmo tempo em que promoveu o aprendizado prático das técnicas e metodologias de engenharia de software. A continuidade do desenvolvimento poderá consolidar ainda mais essa base, oferecendo um sistema robusto, confiável e alinhado às exigências legais e operacionais do setor.

# **ANEXO**

## **ANEXO A – Roteiro flexível planejado para a entrevista com o responsável pela área de TI do cartório**

Roteiro flexível de entrevista – Cartório de títulos, documentos e pessoas jurídicas

* **Contexto e objetivos do sistema**
  + Quais os principais problemas atuais do sistema?
  + O que é esperado do sistema a ser desenvolvido?
  + Quais as funcionalidades do sistema e quais seriam de interesse adicionar ou remover?
  + O atual layout agrada a todos os usuários?
* **Treinamento e suporte**
  + Necessidade e usabilidade de treinamento de funcionários?
  + Como funciona o suporte?
* **Usuários e perfis de acesso**
  + Quem usará o sistema?
  + Quais permissões cada perfil deve ter?
* **Serviços prestados e requisitos funcionais**
  + Quais tipos de documentos são registrados no cartório?
  + Como ocorre o protocolo e a tramitação dos documentos?
  + Quais dados precisam ser capturados?
  + O sistema deve permitir o upload de documentos digitalizados?
* **Fluxo de trabalho e atendimento**
  + Como é feito o atendimento ao público (presencial, online, ambos)?
  + Como é realizado o protocolo de documentos?
  + Existe um prazo máximo para cada tipo de serviço?
  + Como ocorre o acompanhamento do status de cada processo?
* **Integrações e conformidade legal**
  + O sistema deve se integrar com alguma base de dados oficial?
  + O software precisa emitir documentos com **assinatura digital** ou certificado ICP-Brasil?
* **Armazenamento, segurança e backup**
  + O sistema precisa de criptografia para proteger documentos e dados sensíveis?
  + Como será feito o backup dos registros e documentos?
* **Relatórios e auditoria**
  + Quais relatórios o cartório precisa gerar?
  + O sistema deve manter logs detalhados de cada ação realizada pelos usuários?
* **Pagamentos e controle financeiro**
  + Como as taxas são calculadas para cada tipo de serviço?
  + O sistema deve gerar boletos, notas fiscais ou permitir pagamentos online?
  + Como é feito o controle de pagamentos e inadimplência?

## **ANEXO B – Relatório da entrevista com o responsável pela área de TI do cartório**

Indaiatuba, 26 de março de 2025

Relatório 01: Sistema do Cartório de Títulos e Documentos e Pessoas Jurídicas

Entrevistado: Vitor Hugo Oliveira, responsável pelo setor de TI Entrevistador(es): Bruna de Novais Schultz, Camila Barcaro Corrêa e Vitória Caroline da Silva Farias Faturi

No dia 24 de março de 2025, às 21h00 foi realizada a primeira entrevista com o responsável do setor técnico do Cartório Oficial de Registro de Imóveis e Anexos de Indaiatuba/SP, empresa em que atualmente possui 16 funcionários que utilizam o sistema referente a títulos e documentos e pessoas jurídicas.

O sistema atual é um Web utilizado para fins de cadastramento e registro de documentos, contando com mais de 300 funções diferentes, e foi informado que cerca de 70% das funcionalidades são frequentemente utilizadas; já os outros 30% devem ser mantidos para cumprir normas jurídicas obrigatórias à empresa.

Os três principais problemas apontados pelo entrevistado são:

* A questão da alteração de informações quando há duas janelas do sistema abertas com protocolos diferentes – se houver uma mudança em um protocolo serão sobrescritos os dados do outro: Atualmente, essa falha compromete a confiabilidade dos dados registrados, podendo levar a inconsistências e perda de informações;
* O layout que não agrada a maioria dos usuários: Isso dificulta a usabilidade e torna o processo de cadastro de informações menos eficiente. e
* Por ser um sistema descontinuado, a não possibilidade de melhorias nem correção de rotina: Isso resulta em dificuldades operacionais para os funcionários, aumento da incidência de erros no processamento de dados e impossibilidade de adaptação às novas exigências legais e tecnológicas, comprometendo a eficiência e a segurança das operações do cartório.

Outro ponto discutido foi a necessidade de treinamento para os funcionários. Foi relatado que o sistema não é intuitivo e que novos usuários podem levar até três meses para compreender seu funcionamento adequado. Atualmente, não há um programa de capacitação formal oferecido pelo cartório, o que impacta diretamente a produtividade dos funcionários.

No que se refere ao suporte técnico, foi relatado que funciona bem para a empresa, pois eles têm contato direto com os responsáveis, assim não há a necessidade de abrir chamados. Dessa forma, os problemas são resolvidos rapidamente.

Quanto aos perfis de acesso, foi destacado que o sistema será utilizado por funcionários do setor de atendimento, escreventes e pelo menos um profissional do setor de arquivo. A diferenciação de permissões entre escreventes responsáveis e auxiliares deve ser implementada para garantir segurança e controle sobre as informações.

Em relação aos protocolos, será necessário estruturar um banco de dados que relacione números de protocolo – número sequencial entregue ao cliente após ele chegar à recepção, ser atendido e ter seu pedido de serviço analisado; junto com o protocolo é indicada uma previsão de data de finalização do trabalho – e números de registro – número também sequencial que é recebido quando o processo de registro é concluído.

No âmbito do fluxo de trabalho, foi esclarecido que o protocolo de documentos é um processo sequencial, com prazos definidos por normativas legais. Atualmente, os clientes podem acompanhar a situação do seu protocolo através do site do cartório, mas há necessidade de automatizar as mensagens de devolutiva para maior eficiência no atendimento.

Sobre armazenamento e segurança, o entrevistado informou que o cartório realiza cinco backups, sendo armazenados em mídia física, *storage*, nuvem, na Associação dos Registrados de Imóveis do Estado de São Paulo, além de dois servidores locais com replicação em tempo integral. Apesar disso, não há necessidade de criptografia adicional para proteção de documentos e dados sensíveis.

No que se refere aos relatórios, o sistema atual já permite a geração de relatórios pré-definidos, que podem ser acessados por qualquer setor do cartório. Foi enfatizada a importância de manter *logs* detalhados de todas as ações realizadas pelos usuários, garantindo maior rastreabilidade e segurança nas operações.

Quanto às questões financeiras, o entrevistado mencionou que os valores cobrados pelos serviços seguem uma tabela fixa. Não há inadimplência, pois o pagamento ocorre antes da prestação do serviço. O cartório não trabalha com boletos nem aceita pagamentos via cartão de crédito, sendo avaliada a possibilidade de implementação de pagamentos via QR Code ou Pix.

Por fim, foi reforçada a necessidade de integração do novo sistema com bases de dados oficiais e a possibilidade de emissão de documentos com assinatura digital ou certificado ICP-Brasil, garantindo conformidade legal com as exigências normativas.

O próximo passo será a análise detalhada das funcionalidades a serem implementadas, com foco na otimização dos processos críticos apontados durante a entrevista.

## **ANEXO C – Código SQL do banco de dados do projeto**

CREATE TABLE grupo(

id SERIAL NOT NULL,

tipo varchar(50) NOT NULL,

updated\_at timestamp without time zone,

PRIMARY KEY(id)

);

CREATE TABLE natureza(

id SERIAL NOT NULL,

tipo varchar(50) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

id\_grupo bigint NOT NULL,

PRIMARY KEY(id),

CONSTRAINT natureza\_id\_grupo\_fkey FOREIGN key(id\_grupo) REFERENCES grupo(id)

);

CREATE TABLE especie(

id SERIAL NOT NULL,

tipo varchar(20) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id)

);

CREATE TABLE documento(

id SERIAL NOT NULL,

tipo varchar(64) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id)

);

CREATE TABLE apresentante(

id SERIAL NOT NULL,

id\_documento bigint NOT NULL,

numero\_documento varchar(20) NOT NULL,

nome varchar(64) NOT NULL,

numero\_contato varchar(15) NOT NULL,

email varchar(100) NOT NULL,

tipo\_contato varchar(100) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id),

CONSTRAINT apresentante\_id\_documento\_foreign FOREIGN key(id\_documento) REFERENCES documento(id)

);

CREATE TABLE tipo\_parte(

id SERIAL NOT NULL,

tipo varchar(20) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

id\_grupo bigint NOT NULL,

PRIMARY KEY(id),

CONSTRAINT tipo\_parte\_id\_grupo\_fkey FOREIGN key(id\_grupo) REFERENCES grupo(id)

);

CREATE TABLE parte(

id SERIAL NOT NULL,

identificacao varchar(100) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

id\_tipo\_parte bigint NOT NULL,

id\_protocolo bigint NOT NULL,

PRIMARY KEY(id),

CONSTRAINT parte\_id\_protocolo\_foreign FOREIGN key(id\_protocolo) REFERENCES protocolo(id),

CONSTRAINT parte\_id\_tipo\_parte\_foreign FOREIGN key(id\_tipo\_parte) REFERENCES tipo\_parte(id)

);

CREATE TABLE forma\_pagamento(

id SERIAL NOT NULL,

tipo varchar(20) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id)

);

CREATE TABLE usuario(

id SERIAL NOT NULL,

nome varchar(255) NOT NULL,

email varchar(255) NOT NULL,

email\_verified\_at timestamp without time zone,

password varchar(255) NOT NULL,

telefone varchar(255) NOT NULL,

endereco varchar(255) NOT NULL,

setor varchar(255) NOT NULL,

usuario varchar(255) NOT NULL,

foto varchar(255),

remember\_token varchar(100),

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id)

);

CREATE UNIQUE INDEX usuario\_email\_unique ON public.usuario USING btree (email);

CREATE TABLE protocolo(

id SERIAL NOT NULL,

numero\_documento integer NOT NULL,

data\_documento date NOT NULL,

data\_abertura date NOT NULL DEFAULT CURRENT\_DATE,

data\_cancelamento date,

numero\_protocolo integer,

numero\_registro integer,

data\_retirada timestamp without time zone,

data\_registro timestamp without time zone,

id\_usuario bigint NOT NULL,

id\_apresentante bigint NOT NULL,

id\_grupo bigint NOT NULL,

id\_especie bigint NOT NULL,

id\_natureza bigint NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id),

CONSTRAINT protocolo\_id\_apresentante\_foreign FOREIGN key(id\_apresentante) REFERENCES apresentante(id),

CONSTRAINT protocolo\_id\_especie\_foreign FOREIGN key(id\_especie) REFERENCES especie(id),

CONSTRAINT protocolo\_id\_grupo\_foreign FOREIGN key(id\_grupo) REFERENCES grupo(id),

CONSTRAINT protocolo\_id\_natureza\_foreign FOREIGN key(id\_natureza) REFERENCES natureza(id),

CONSTRAINT protocolo\_id\_usuario\_foreign FOREIGN key(id\_usuario) REFERENCES usuario(id)

);

CREATE TABLE autenticacao(

id SERIAL NOT NULL,

valor numeric(10,2) NOT NULL DEFAULT '0'::numeric,

data\_autenticacao date,

numero\_cheque integer,

agencia varchar(6),

conta varchar(15),

banco varchar(30),

id\_usuario bigint NOT NULL,

id\_protocolo bigint NOT NULL,

id\_forma\_pagamento bigint NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id),

CONSTRAINT autenticacao\_id\_forma\_pagamento\_foreign FOREIGN key(id\_forma\_pagamento) REFERENCES forma\_pagamento(id),

CONSTRAINT autenticacao\_id\_protocolo\_foreign FOREIGN key(id\_protocolo) REFERENCES protocolo(id),

CONSTRAINT autenticacao\_id\_usuario\_foreign FOREIGN key(id\_usuario) REFERENCES usuario(id)

);

CREATE TABLE tipo\_andamento(

id SERIAL NOT NULL,

tipo varchar(50) NOT NULL,

possui\_valor varchar(3) NOT NULL,

created\_at timestamp without time zone,

updated\_at timestamp without time zone,

PRIMARY KEY(id)

);

CREATE TABLE andamento(

id SERIAL NOT NULL,

data\_hora timestamp without time zone NOT NULL DEFAULT '2025-06-03 23:25:19'::timestamp without time zone,

valor numeric(10,2) NOT NULL DEFAULT '0'::numeric,

observacao varchar(255),

id\_tipo\_andamento bigint NOT NULL,

id\_usuario bigint NOT NULL,

id\_protocolo bigint NOT NULL,

updated\_at timestamp with time zone,

created\_at timestamp with time zone,

PRIMARY KEY(id),

CONSTRAINT andamento\_id\_protocolo\_foreign FOREIGN key(id\_protocolo) REFERENCES protocolo(id),

CONSTRAINT andamento\_id\_tipo\_andamento\_foreign FOREIGN key(id\_tipo\_andamento) REFERENCES tipo\_andamento(id),

CONSTRAINT andamento\_id\_usuario\_foreign FOREIGN key(id\_usuario) REFERENCES usuario(id)

);

CREATE OR REPLACE FUNCTION public.atribui\_numero\_protocolo()

RETURNS trigger

LANGUAGE plpgsql

SET search\_path TO 'public'

AS $function$

BEGIN

IF NEW.numero\_protocolo IS NULL THEN

IF NEW.id\_grupo = 1 THEN

NEW.numero\_protocolo := nextval('seq\_grupo\_1');

ELSIF NEW.id\_grupo = 2 THEN

NEW.numero\_protocolo := nextval('seq\_grupo\_2');

END IF;

END IF;

RETURN NEW;

END;

$function$

INSERT INTO grupo (id, tipo, updated\_at) VALUES

(1, 'Títulos e Documentos', NOW()),

(2, 'Pessoa Jurídica', NOW());

INSERT INTO natureza (id, tipo, created\_at, updated\_at, id\_grupo) VALUES

(1, 'Ata de Condomínio', NOW(), NOW(), 1),

(2, 'Cedula de Crédito', NOW(), NOW(), 1),

(3, 'Conservação', NOW(), NOW(), 1),

(4, 'Notificação', NOW(), NOW(), 1),

(5, 'Tradução', NOW(), NOW(), 1),

(6, 'Ata de Assembleia', NOW(), NOW(), 2),

(7, 'Abertura de Filial', NOW(), NOW(), 2),

(8, 'Contrato Social', NOW(), NOW(), 2),

(9, 'Distrato', NOW(), NOW(), 2),

(10, 'Estatuto', NOW(), NOW(), 2);

INSERT INTO especie (id, tipo, created\_at, updated\_at) VALUES

(1, 'Registro', NOW(), NOW()),

(2, 'Averbação', NOW(), NOW());

INSERT INTO documento (id, tipo, created\_at, updated\_at) VALUES

(1, 'RG', NOW(), NOW()),

(2, 'CPF', NOW(), NOW()),

(3, 'CNH', NOW(), NOW()),

(4, 'CNPJ', NOW(), NOW());

INSERT INTO tipo\_parte (id, tipo, created\_at, updated\_at, id\_grupo) VALUES

(1, 'Condomínio', NOW(), NOW(), 1),

(2, 'Destinatário', NOW(), NOW(), 1),

(3, 'Emitente', NOW(), NOW(), 1),

(4, 'Parte', NOW(), NOW(), 1),

(5, 'Remetente', NOW(), NOW(), 1),

(6, 'Síndico', NOW(), NOW(), 1),

(7, 'Associação', NOW(), NOW(), 2),

(8, 'Diretor Executivo', NOW(), NOW(), 2),

(9, 'Presidente', NOW(), NOW(), 2),

(10, 'Secretário', NOW(), NOW(), 2),

(11, 'Sócio', NOW(), NOW(), 2);

# **REFERÊNCIAS**

CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA. **Conteúdo Programático: Curso de Análise e Desenvolvimento de Sistemas – Fatec Indaiatuba**. Indaiatuba: Fatec Indaiatuba, [s.d.]. Disponível em: <https://www.fatecid.com.br/site/wp-content/uploads/downloads/secretaria/conteudo/conteudo_ads.pdf>. Acesso em: 7 jun. 2025.

MELO, Tiago de. **Extração de Requisitos**. CEFET–PB, 2009. Disponível em: <http://tiagodemelo.info/aulas/cefet/2009/asoo/extracao-requisitos.pdf>. Acesso em: 7 jun. 2025.