



Sistema para Gerenciamento de Empréstimos / Biblioteca

Leonardo Nascimento Dias

UENF - CCT - LCMAT - CC

15 de novembro de 2022



Copyright © 2022 Leonardo Nascimento Dias

UENF - UNIVERSIDADE ESTADUAL DO NORTE FLUMINENSE DARCY RIBEIRO

CCT - CENTRO DE CIÊNCIA E TECNOLOGIA
LCMAT - LABORATÓRIO DE MATEMÁTICAS
CC - CURSO DE CIÊNCIA DA COMPUTAÇÃO



Sumário

1	Introdução	1
1.1	Contextualização do Sistema OO	1
1.1.1	Referencial Teórico	2
1.2	Objetivo do Sistema	2
1.3	Objetivos específicos do Sistema	2
1.4	Justificativa	2
2	Requisitos do Sistema OO	5
2.1	Requisitos Funcionais	5
2.2	Requisitos Não-Funcionais	6
2.2.1	Requisitos de Usabilidade	6
2.2.2	Requisitos de Confiabilidade	6
2.2.3	Requisitos de Disponibilidade	7
2.2.4	Requisitos de Privacidade	7
2.2.5	Requisitos de Acesso	7
2.2.6	Requisitos de portabilidade	7
2.3	Requisitos de Negócios	8
3	Modelagem do Sistema	9
3.1	Diagramas DFD	9
3.1.1	Cadastrar Usuário	9
3.1.2	Empréstimo	10
3.1.3	Devolução	11
3.1.4	Multas	11

3.1.5	Consulta por Autor	12
3.1.6	Consulta por Título	13
3.1.7	Consulta Disponibilidade Livro	13
3.1.8	Consulta tabela de Empréstimos	14
3.1.9	Fornecedor envia livros	15
3.1.10	Diagrama Gerir Empréstimos	15
3.1.11	Diagrama Gerir Livros	16
3.1.12	Diagrama Sistema	17
3.2	Diagramas E-R	18
3.3	Diagramas de Classes	20
3.4	Diagramas Casos de uso	20
3.5	Diagramas Sequência	21
3.5.1	Diagrama de Sequencia Login	22
3.5.2	Diagrama de Sequencia Devolução	22
3.5.3	Diagrama de Sequencia Emprestimo	23
3.6	Diagramas de Atividades	23
3.6.1	Diagrama de Atividades Devolução	24
3.6.2	Diagrama de Atividades Empréstimo	24
3.6.3	Diagrama de Atividades Cadastro Usuário	25
3.7	Diagramas Estado	25
3.7.1	Diagrama de Estado Login	26
3.7.2	Diagrama de Estado Consulta	26
3.7.3	Diagrama de Estado Cadastro Usuário	27
3.7.4	Diagrama de Estado Empréstimo	27
3.7.5	Diagrama de Estado Devolução	28
4	Projeto do Sistema OO	29
4.1	Arquitetura do Sistema - Classes	29
4.2	Interfaces do Usuário	33
4.2.1	Interfaces do Usuário - Modais	35
4.3	Tabelas de Dados	38
5	Implementação do Sistema OO	43
5.1	Programação	43
5.1.1	Classes	45
5.1.2	Códigos Importantes	45
5.2	Documentação do Software	49
6	Considerações Finais	51
	Bibliografia	53



1. Introdução

Paradigma de Desenvolvimento de Sistemas Orientado a Objetos é uma disciplina orientada a desenvolver um sistema utilizando a metodologia orientada a Objetos em todas as etapas do Ciclo de Vida de Desenvolvimento de um Sistema (CVDS). As referências bibliográficas básicas a serem consultadas são: [DWR14], [Hel13], [Gue11], [Som18] e [Waz11]. Como bibliografia complementar serão considerados: [SJB12], [SR12] e [Fur13].

Neste documento serão apresentadas as principais atividades realizadas para o desenvolvimento COMPLETO de uma aplicação OO.

O sistema a ser desenvolvido é um sistema de gerenciamento de uma Biblioteca.

Atualmente o uso de um software de gerenciamento vem crescendo muito nas empresas e/ou qualquer tipo de entidade, isso se deve pelo fato de haver inúmeras informações a serem guardadas e manipuladas e a dificuldade de isto ser realizado manualmente. Com isso, a utilização de um software de Gerenciamento resulta na exatidão e agilidade nos resultados obtidos, que são de grande valia para qualquer tipo de organização. O software desenvolvido trata-se de um Sistema Gerencial de Biblioteca.

1.1 Contextualização do Sistema OO

O Sistema ajudará a reduzir muito o tempo gasto em algumas operações, tais como levantamento total de acervo existente na biblioteca emitido através de relatórios.

Ele permitirá aos funcionários cadastrar acervos e clientes, efetuar empréstimos, devoluções, reservas e consultas de todo o acervo existente, além de obter por meio de relatórios resultados importantes para ter total controle e eficiência para uma eventual tomada de decisão.

O sistema para gerenciamento de biblioteca deverá conter cadastro de usuários, no qual dados serão colhidos dos usuários, funcionários e livros para possíveis empréstimos e devoluções de livros. Para que um empréstimo de livros seja efetuado será necessário que o livro esteja devidamente cadastrados no sistema, lembrando que o aluno deverá estar sem multas em seu registro para fazer um empréstimo. Caso o livro esteja Cadastrado, o empréstimo será efetuado. Logo após o lançamento do empréstimo poderá gerar uma tabela de empréstimos de cada usuário contendo a

data de devolução.

1.1.1 Referencial Teórico

Como auxilio para esse projeto, foram analisados os trabalhos de conclusão de curso: "Sistema de gerenciamento para biblioteca" por Giovanne Marangoni Martins [Mar15]

"Uma abordagem de implementação de uma biblioteca eletrônica utilizando ferramentas de domínio público" por Augusto Eduardo Pôrto Paes [Pae03]

"Desenvolvimento de software de gestão para a biblioteca do cesec juscelino kubitschek de oliveira do município de diamantina-mg" por Eduardo Augusto Costa Trindade [Tri18]

"Sistema de biblioteca" por Verusca Cristina Inácio [Iná08]

1.2 Objetivo do Sistema

Tem como objetivos principais, o desenvolvimento de um sistema gerenciador de biblioteca, que possibilite os funcionários a fazerem pesquisa de quais livros estão reservados ou livres e fazer consulta para indicar aonde está localizado o livro desejado, ver histórico de cada cliente, etc, tornando assim, mais eficaz a consulta de um livro desejado. Para ter acesso livre para pegar livros, o aluno deverá ter um cadastro na biblioteca. Já os interessados que não tiverem cadastro, deverão se dirigir a biblioteca e se cadastrar, fazendo com que aumente o interesse e futuramente todas as pessoas possam ter seu cadastro.

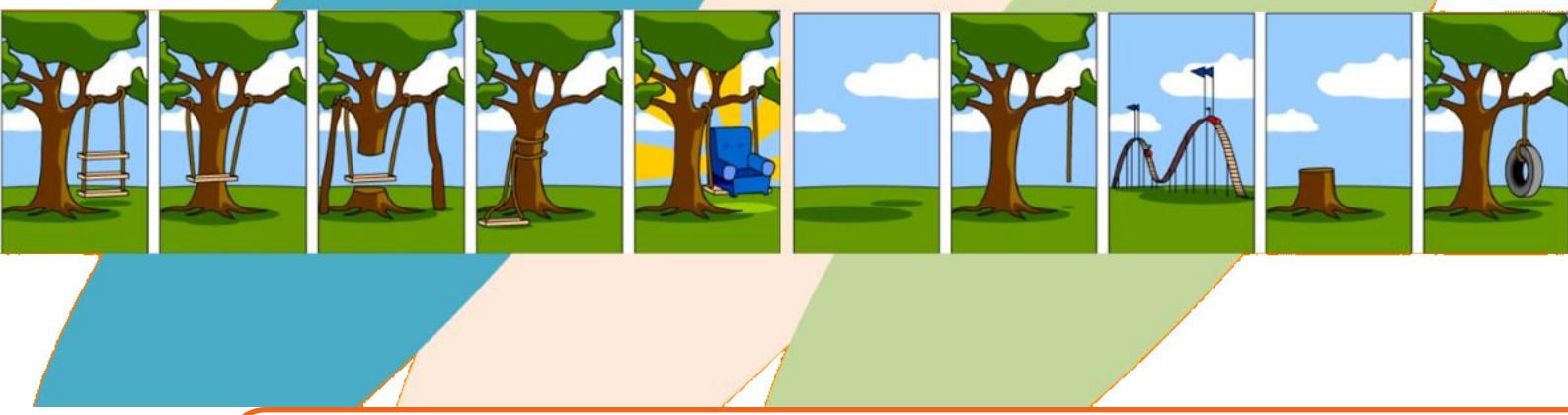
1.3 Objetivos específicos do Sistema

- Autenticação de Usuário
- Manter usuário, funcionário, livro, editora, periódico
 - Armazenar dados de cada processo para que não sejam perdidos.
- Relatório clientes, empréstimos, livros, funcionários, periódicos
 - Gerar relatórios para manter controle do que ocorre dentro da empresa e evitar futuros problemas
- Consulta clientes, livros funcionários editora, periódico
- Gerar multas
- Cadastro de itens
 - Para cada item deve se informar o título, autor, editora, assunto, e para cada item podem ser cadastrados diversos exemplares.
- Cadastro de usuários
 - O sistema deve permitir cadastrar Funcionários e usuários Requisitos não funcionais.
- Empréstimo de itens
 - Sistema deve fazer o empréstimo de itens para os usuários cadastrado na biblioteca.
- Negar Empréstimo
 - Impedir que o usuário com multas solicite livros.

1.4 Justificativa

A justificativa de implementação de um Sistema de Gerenciamento de Biblioteca se da em pela necessidade de controle e gerenciamento de todo acervo pertencente a instituição, alem de controlar

e emitir relatórios para uma melhor administração, e com isso reduzir o tempo gasto na execução de serviços.



2. Requisitos do Sistema OO

Neste capítulo é apresentado listas, definições e especificações de Requisitos do sistema ser desenvolvido. Os requisitos são declarações abstratas de alto nível sobre os *serviços* que o sistema deve prestar à organização, e as *restrições* sobre as quais deve operar. Os requisitos sempre refletem as necessidades dos clientes do sistema.

Sobre os requisitos, Raul S. Wazlawick afirma:

A etapa de levantamento de requisitos corresponde a buscar todas as informações possíveis sobre as funções que o sistema deve executar e as restrições sobre as quais o sistema deve operar. O produto dessa etapa será o documento de requisitos, principal componente do anteprojeto de software.

A etapa de análise de requisitos serve para estruturar e detalhar os requisitos de forma que eles possam ser abordados na fase de elaboração para o desenvolvimento de outros elementos como casos de uso, classes e interfaces.

O levantamento de requisitos é o processo de descobrir quais são as *funções* que o sistema deve realizar e quais são as *restrições* que existem sobre estas funções [Waz11].

Este projeto visa dinamizar, facilitar e agilizar os processos e atividades realizadas pelo sistema, promovendo um melhor serviço para os clientes e mais eficiência no trabalho dos funcionários.

2.1 Requisitos Funcionais

Um requisito de sistema de software que especifica uma função que o sistema ou componente deve ser capaz de realizar. Estes são requisitos de software que definem o comportamento do sistema, ou seja, o processo ou transformação que componentes de software ou hardware efetuam sobre as entradas para gerar as saídas. Esses requisitos capturam as funcionalidade sob o ponto de vista do usuário.

- Acervo
 - 1. Exibir livros
 - 2. Consultar livro
 - 3. Mostrar imagem do livro

- 4. Excluir livro
- 5. Alterar dados
- Empréstimo
 - 1. Tabela de registro
 - 2. Empréstimar para usuários cadastrados
 - 3. Empréstimar para usuário não cadastrados
 - 4. Avaliar usuário
 - 5. Não permitir empréstimos de livro indisponíveis
- Devolução
 - 1. Fazer devolução do livro
 - 2. tabela de empréstimos
 - 3. Gerar multa
 - 4. Conferir tempo de empréstimo
 - 5. Não devolver livro de outro usuário
- Cadastro
 - 1. Cadastrar usuários
 - 2. Cadastrar Categoria
 - 3. Cadastrar livros
 - 4. Inserir imagens

2.2 Requisitos Não-Funcionais

Requisitos não funcionais são aqueles que não estão diretamente relacionados à funcionalidade de um sistema. O termo requisitos não funcionais é também chamado de atributos de qualidade. Os requisitos não funcionais têm um papel de suma importância durante o desenvolvimento de um sistema, podendo ser usados como critérios de seleção na escolha de alternativas de projeto, estilo arquitetural e forma de implementação. Desconsiderar ou não considerar adequadamente tais requisitos é dispendioso, pois torna difícil a correção uma vez que o sistema tenha sido implementado. Os requisitos não funcionais abordam aspectos de qualidade importantes em sistemas de software. Se tais requisitos não são levados em consideração, então o sistema de software poderá ser inconsistente e de baixa qualidade.

2.2.1 Requisitos de Usabilidade

Usabilidade é um dos atributos de qualidade ou requisitos não funcionais de qualquer sistema interativo, ou seja, no qual ocorre interação entre o sistema e seres humanos. A noção de usabilidade vem do fato que qualquer sistema projetado para ser utilizado pelas pessoas deveria ser fácil de aprender e fácil de usar, tornando assim fácil e agradável a realização de qualquer tarefa.

- 1. Uso de Design responsivo nas interfaces gráficas
- 2. Facil de aprender
- 3. O sistema deve ser rapido
- 4. Deve ser facil de usar
- 5. Deve ser acessivel

2.2.2 Requisitos de Confiabilidade

Confiabilidade de software é a probabilidade de o software não causar uma falha num sistema durante um determinado período de tempo sob condições especificadas. A probabilidade é uma

função da existência de defeitos no software. Assim, os estímulos recebidos por um sistema determinam a existência ou não de algum defeito. Em outras palavras, a confiabilidade de software, geralmente definida em termos de comportamento estatístico, é a probabilidade de que o software irá operar como desejado num intervalo de tempo conhecido. Também, a confiabilidade caracteriza-se um atributo de qualidade de software o qual implica que um sistema executará suas funções como esperado.

1. Deve armazenar a senha no banco de dados de forma segura
2. deve ter boa disponibilidade
3. Não pode permitir que usuários alterem informações de outros.
4. Não permitir alteração no código fonte da página que influenciam no funcionamento do sistema
5. Deve ser feito cópias (backup) de todos os dados do sistema a cada 24 horas

2.2.3 Requisitos de Disponibilidade

1. O sistema estará disponível pelo menos 99,7% do tempo em dias de semana e pelo menos 99,95% finais de semana
2. Um novo usuário deverá ser capaz de usar o sistema após não mais do que 30 minutos de orientação
3. Utilização do módulo de Informações Cadastrais em modo off-line

2.2.4 Requisitos de Privacidade

Em um sistema de software, este requisito não funcional caracteriza a segurança de que acessos não autorizados ao sistema e dados associados não serão permitidos. Portanto, é assegurada a integridade do sistema quanto a ataques intencionais ou acidentes. Dessa forma, a segurança é vista como a probabilidade de que a ameaça de algum tipo será repelida.

1. Senha criptografada no momento do cadastro
2. Acesso e modificação
3. Identificação dos participantes
4. Segurança dos dados
5. Notificação

2.2.5 Requisitos de Acesso

1. Autenticação de usuário para consumo de webservices do sistema por sistemas externos
2. Apenas usuários autenticados poderão visualizar informações.
3. As permissões de acesso ao sistema podem ser alteradas apenas pelo administrador de sistemas.

2.2.6 Requisitos de portabilidade

Portabilidade pode ser definida como a facilidade na qual o software pode ser transferido de um sistema computacional ou ambiente para outro. Em outras palavras, o software é dito portável se ele pode ser executado em ambientes distintos. Note que o termo ambiente pode referir-se tanto à plataforma de hardware quanto a um ambiente de software como, por exemplo, um sistema operacional específico.

1. O sistema deverá executar em qualquer plataforma

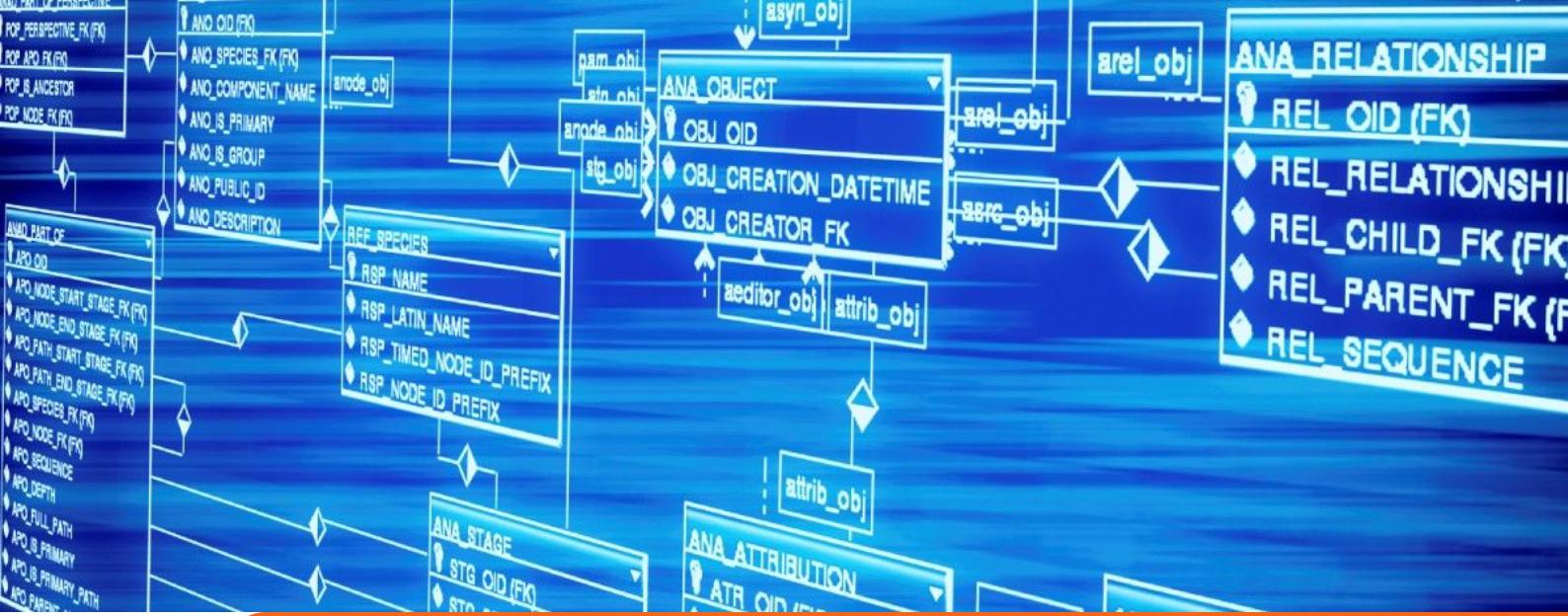
2.3 Requisitos de Negócios

Requisitos do negócio são requisitos de alto nível que explicam e justificam qualquer projeto. Os requisitos de negócios são as atividades críticas de uma empresa que devem ser executadas para atender ao(s) objetivo(s) organizacional(is) enquanto permanecem independentes do sistema solução.

Espera-se uma redução de gastos e tempo com o controle do acervo, que possibilita um melhor acompanhamento e gerenciamento dos acontecimentos da Biblioteca. Com análises dos dados de usuários cadastrados no sistema para adequação do acervo, é previsto uma expansão e fidelização da base de clientes, gerando um aumento consequente nos lucros da empresa.

1. O sistema deverá possibilitar que o funcionário da biblioteca possa registrar novos livros e atualizar o sistema para que os clientes possam ter acesso as mudanças;
2. Deve fazer o cadastro de clientes e de funcionários para que eles possam utilizar o sistema;
3. Permitir que clientes reservem os livros desejados, desde que estejam cumprindo os requisitos;
4. Consultar clientes, livros, funcionários possibilitando o gerenciamento dos mesmos, tendo um histórico de movimentação.
5. Gerar multas para clientes que não farão a devolução dos livros dentro do prazo, para que os mesmos não possam fazer outras reservas até que expliquem o motivo do atraso e paguem a multa.
6. Fornecer meios de comunicação, relacionamento com clientes, vendas online e entrega de produtos através da web.
7. Permitir sugestões dos clientes para modificações no acervo, fazendo com que a biblioteca não fique estagnada com conteudos fora de época.

Parte importante do trabalho será a estruturação da interface de usuário, com o intuito de facilitar a transição dos processos manuais para a utilização do sistema e poder efetivamente, agilizar todos os processos executados pela rede, assim como agilizar o relacionamento e trocar de dados entre a mesma.



3. Modelagem do Sistema

Neste capítulo veremos algumas formas de representar graficamente os processos do sistema e como eles se relacionam com agente externos e bases de dados. Os diagramas tem como finalidade representar alguns processos específicos bem como o sistema por inteiro de forma ampla, genérica, e objetiva, para norteamento do objetivo geral do sistema.

3.1 Diagramas DFD

O diagrama de fluxo de dados (DFD) é uma representação gráfica do "fluxo" de dados através de um sistema de informação, modelando seus aspectos de processo. Ele fornece apenas uma visão do sistema, a visão estruturada das funções, ou seja, o fluxo dos dados.

3.1.1 Cadastrar Usuário

Representando o cadastro de um novo cliente para se tornar um usuário da biblioteca.

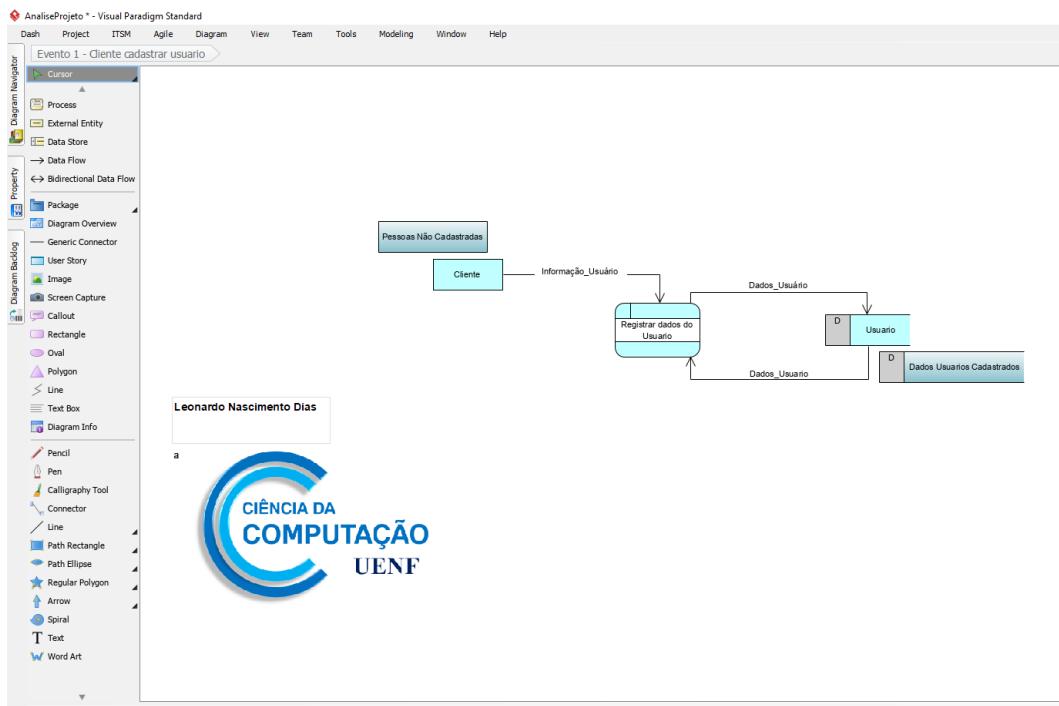


Figura 3.1: Cadastrar Usuário

3.1.2 Empréstimo

Representação do processo para um usuário fazer um empréstimo.

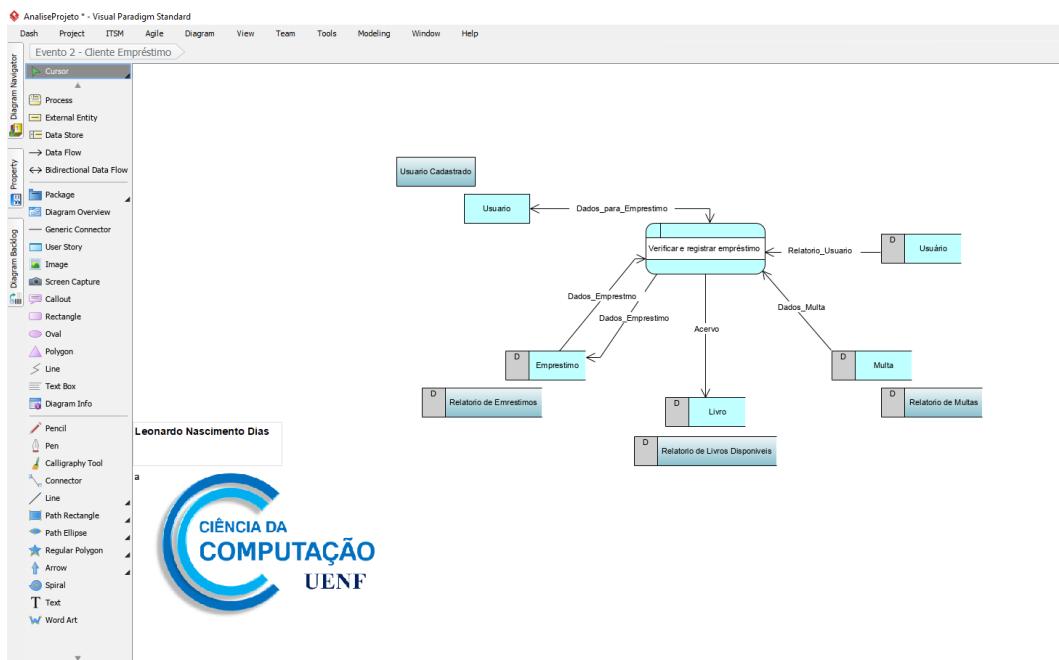


Figura 3.2: Empréstimo

3.1.3 Devolução

Representação do processo de devolução de livros pegos por usuário.

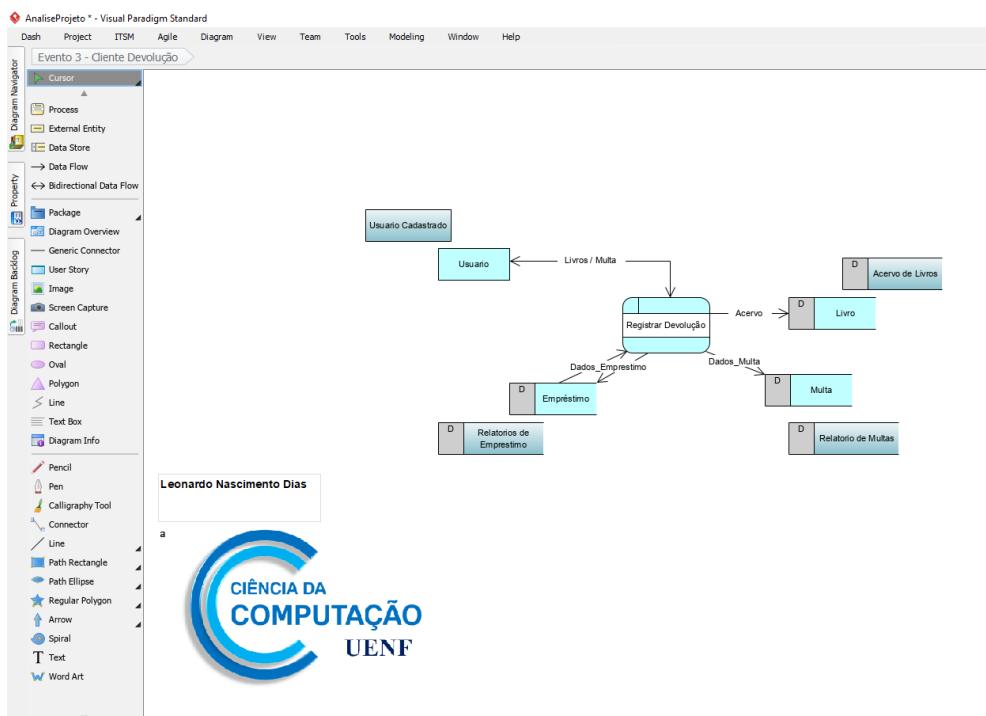


Figura 3.3: Devolução

3.1.4 Multas

Representação de Usuário que teve uma multa aplicada em sua conta, e tem de fazer o pagamento para continuar utilizando os serviços do sistema.

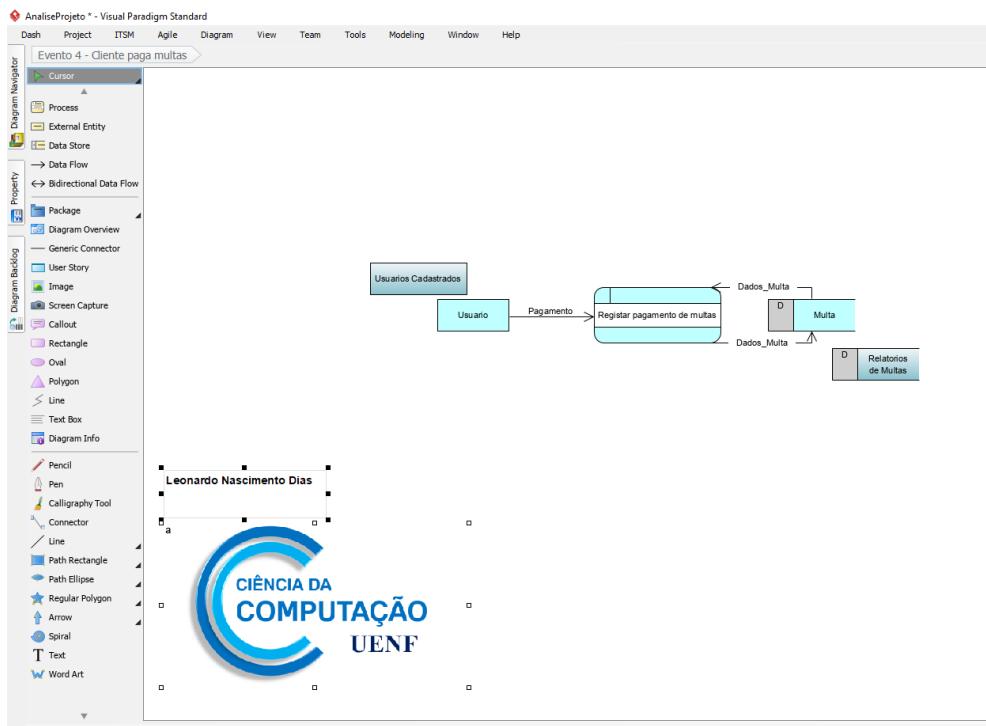


Figura 3.4: Pagamento Multa

3.1.5 Consulta por Autor

Processo onde Usuário faz a pesquisa do livro desejado baseado no autor do proprio.

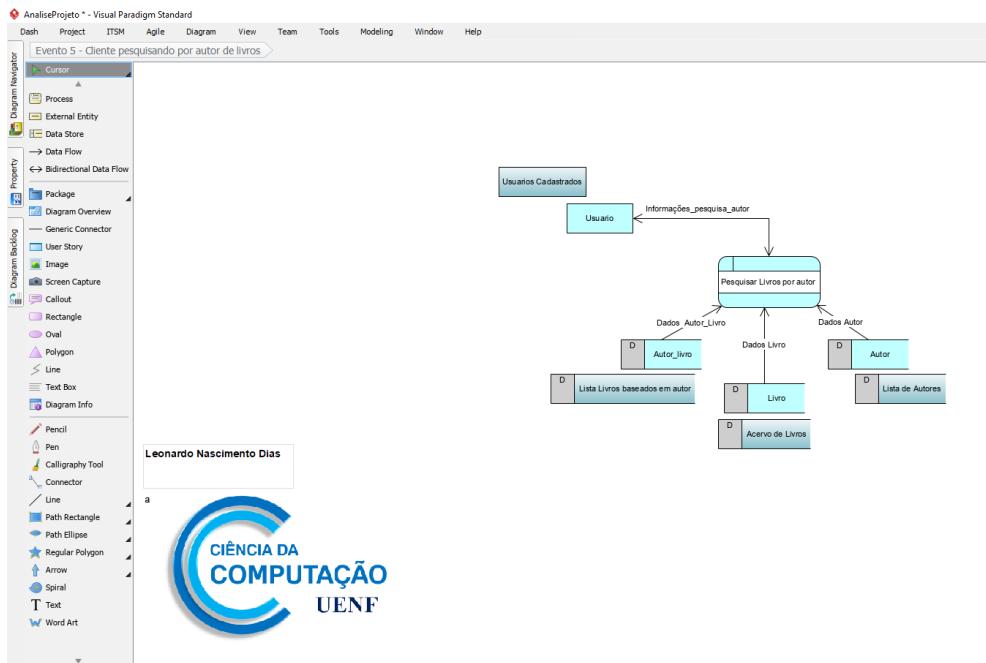


Figura 3.5: Pesquisa Autor

3.1.6 Consulta por Título

Processo onde Usuário pesquisa livro desejado inserindo seu título.

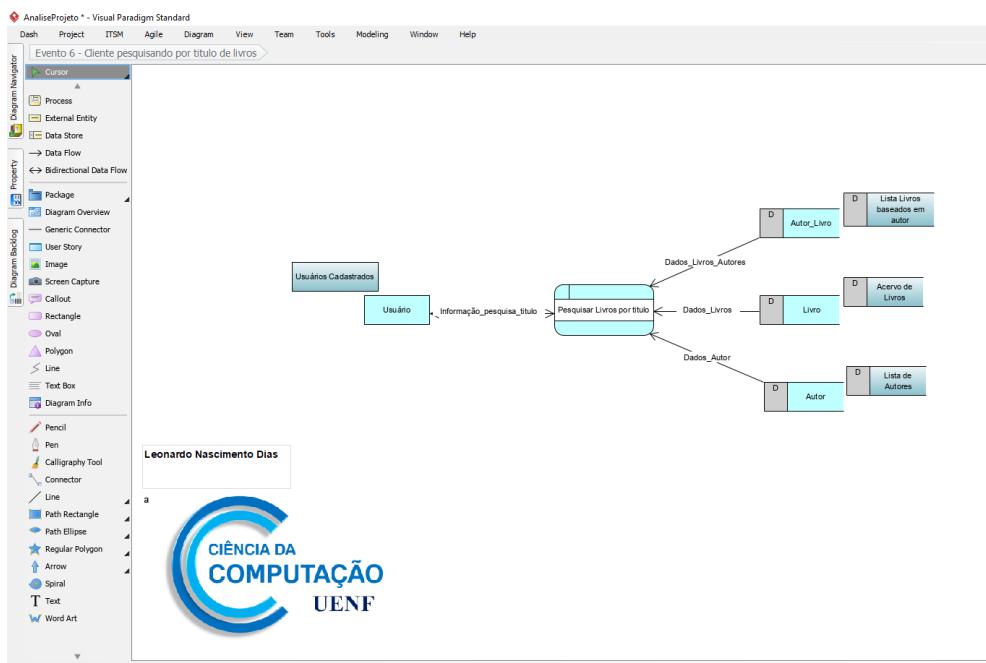


Figura 3.6: Pesquisa Título

3.1.7 Consulta Disponibilidade Livro

Processo para usuário descobrir se o livro que deseja está disponível para empréstimos ou quando estará caso já esteja com outro cliente.

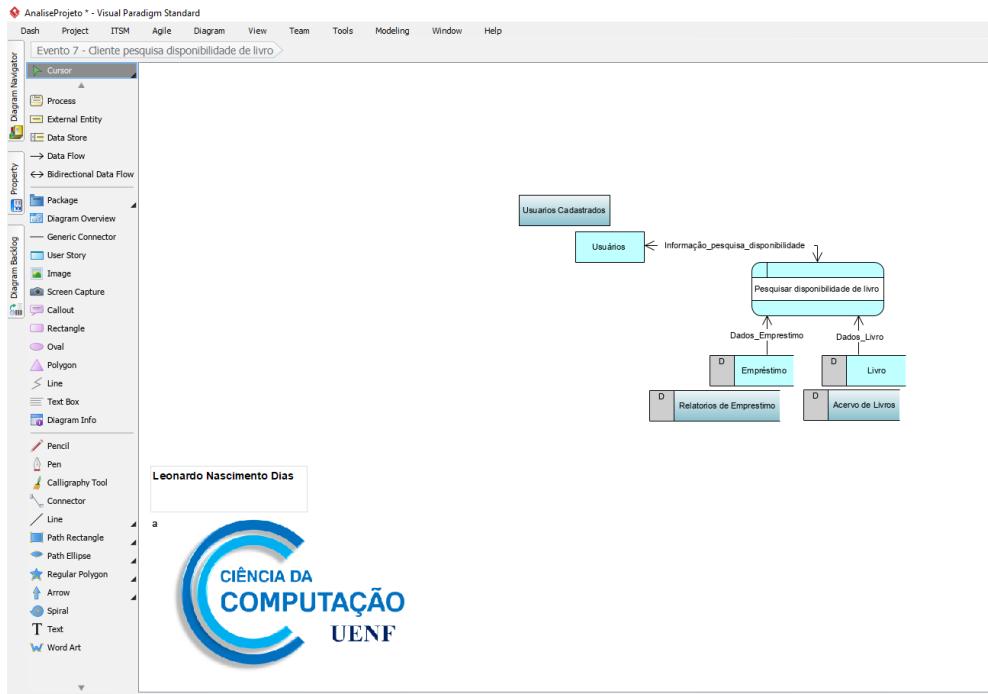


Figura 3.7: Pesquisa Disponibilidade

3.1.8 Consulta tabela de Empréstimos

Processo para funcionário gerar relatório de empréstimo de um usuário ou de um livro.

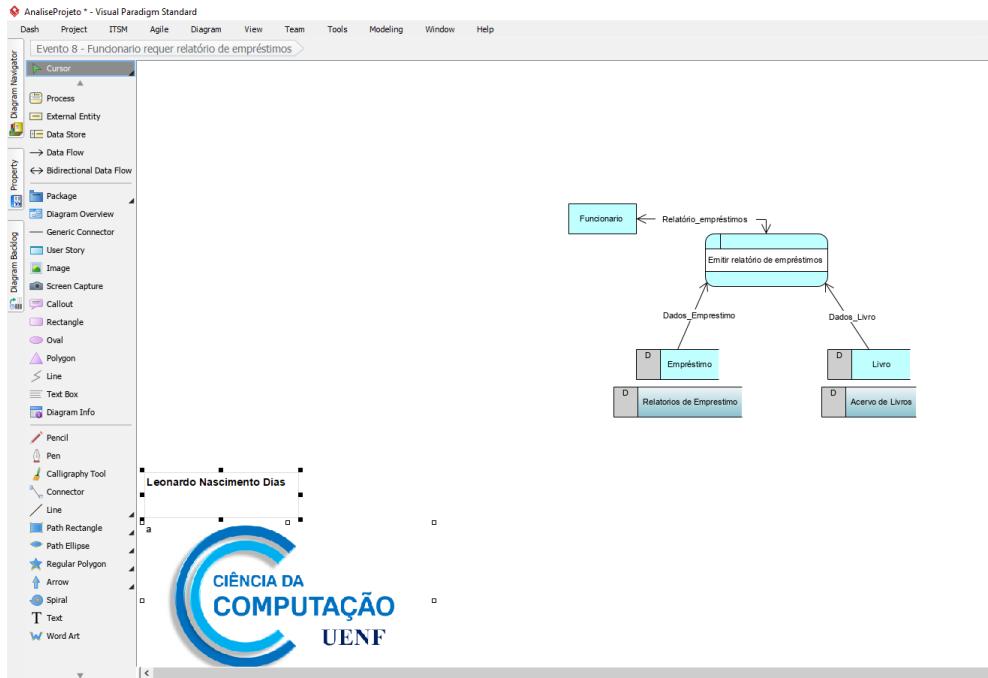


Figura 3.8: Relatório

3.1.9 Fornecedor envia livros

Processo para registro de livros enviados por fornecedores.

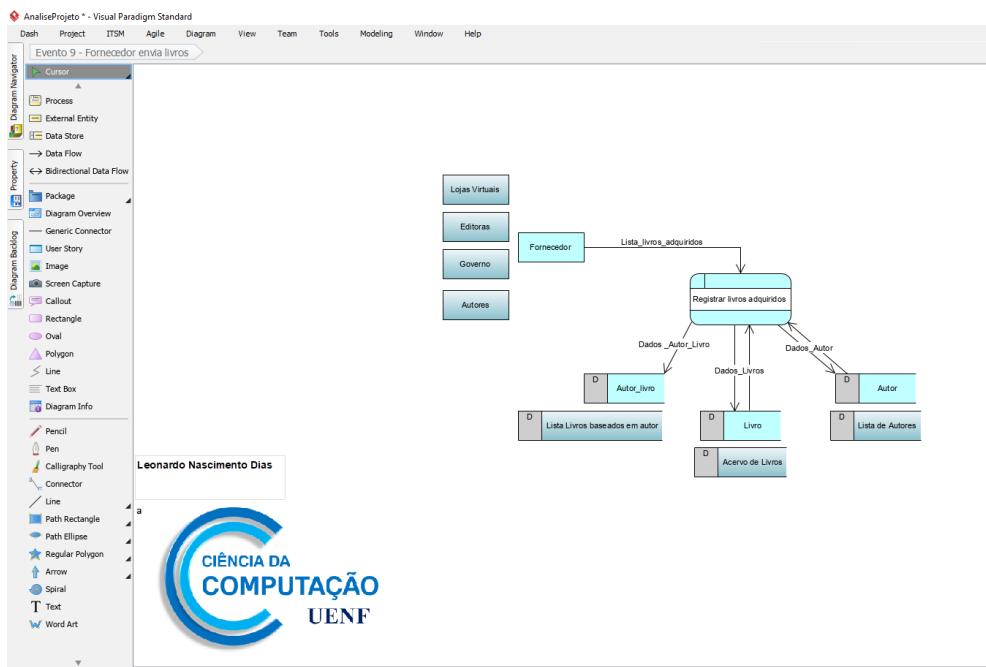


Figura 3.9: Envio Fornecedor

3.1.10 Diagrama Gerir Empréstimos

Diagrama representado o agrupamento dos processos Cadastrar Usuário, Empréstimo, Devolução, Multas.

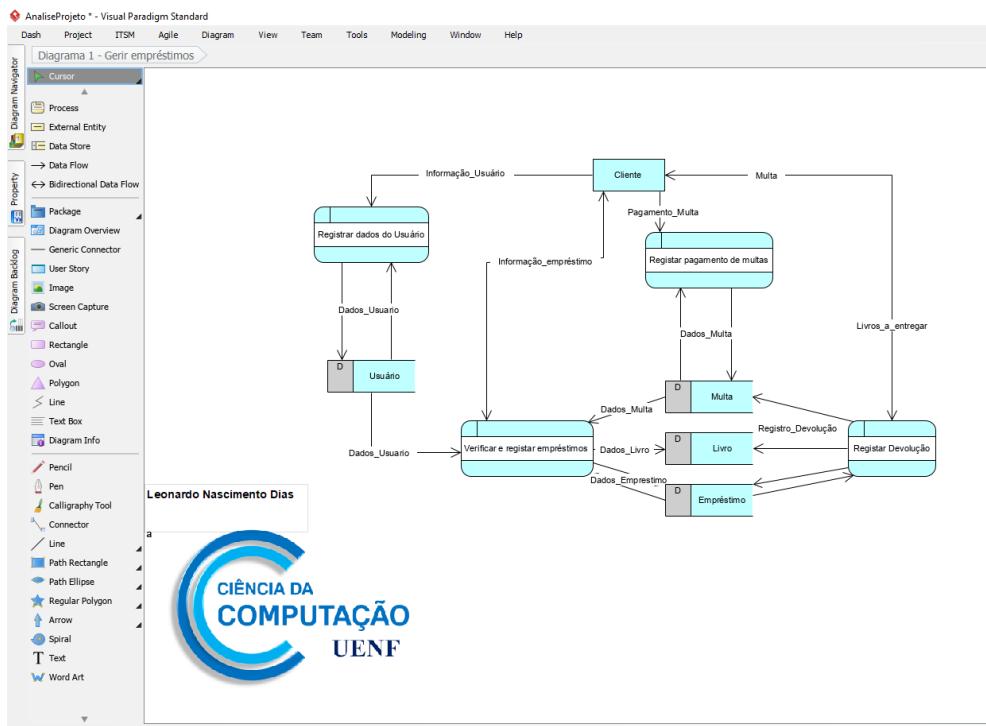


Figura 3.10: Empréstimo Geral

3.1.11 Diagrama Gerir Livros

Diagrama representando o agrupamento dos processos Consulta por Autor, Consulta por Título, Consulta Disponibilidade Livro, Envio Fornecedor.

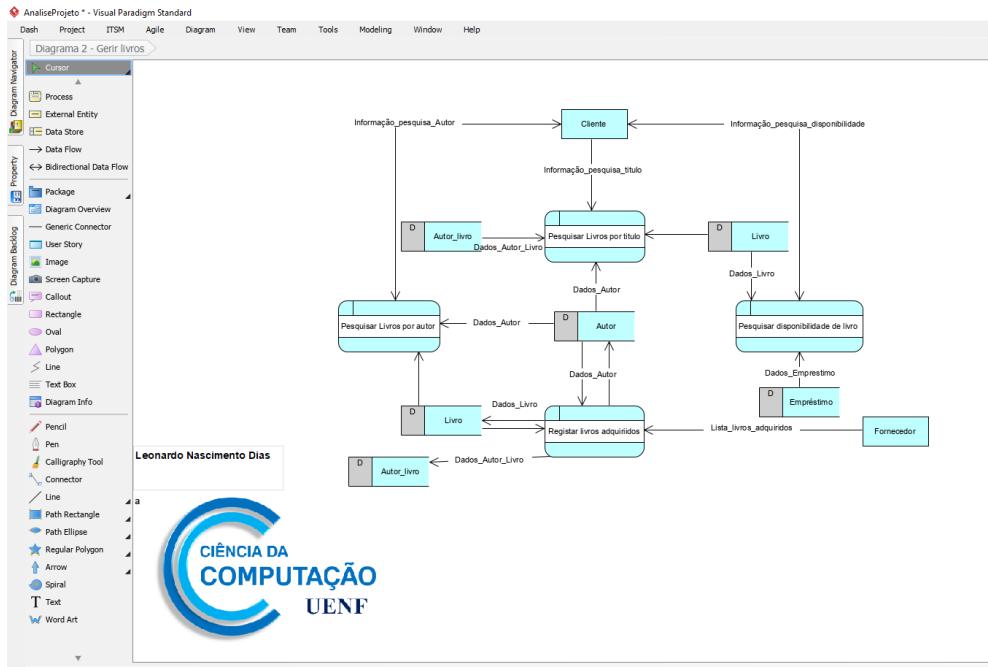


Figura 3.11: Livros Geral

3.1.12 Diagrama Sistema

Diagrama Geral do Sistema representando o agrupamento de Todos os processos.

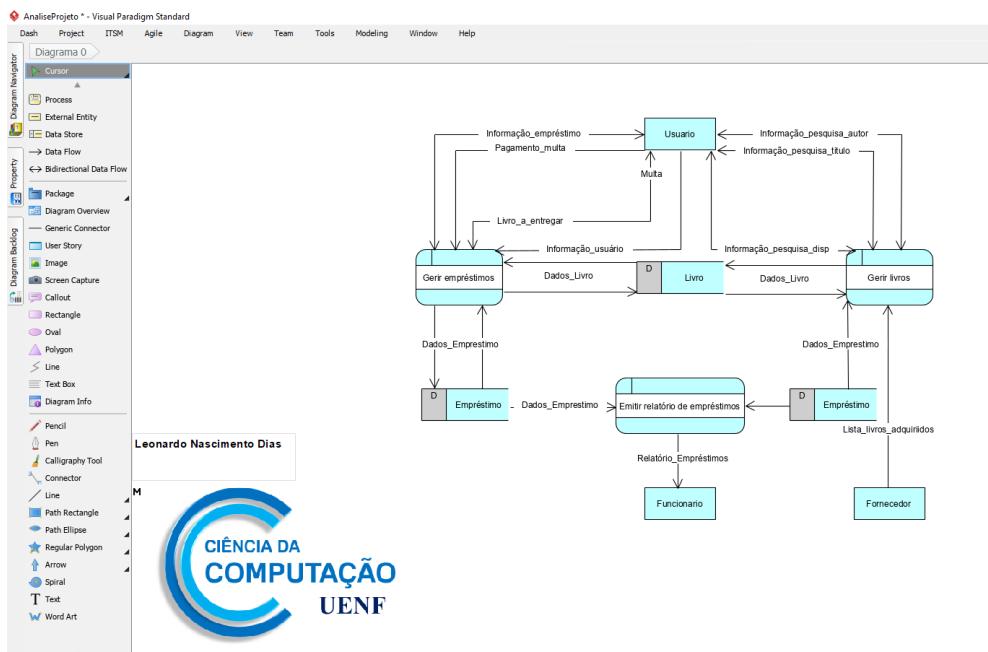


Figura 3.12: Sistema Geral

3.2 Diagramas E-R

Um modelo entidade relacionamento é um modelo de dados para descrever os dados ou aspectos de informação de um domínio de negócio ou seus requisitos de processo, de uma maneira abstrata que em última análise se presta a ser implementada em um banco de dados, como um banco de dados relacional.

Os diagramas a seguir representam a modelagem de relacionamento entre as entidades do sistema.

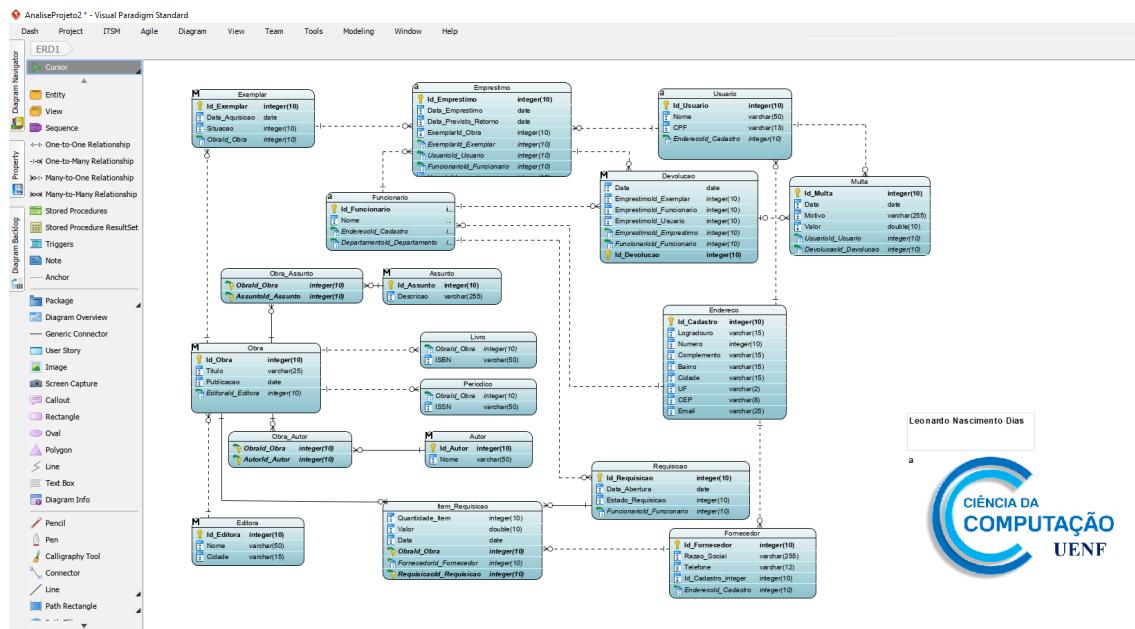


Figura 3.13: ER01 - Sistema Processos

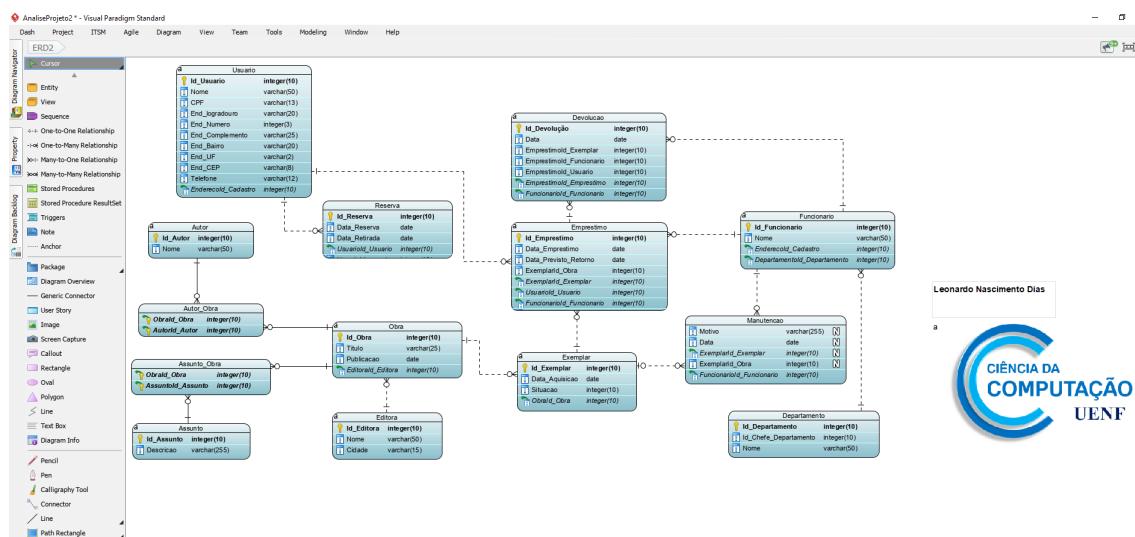


Figura 3.14: ER02 - Sistema Processos

Os Seguintes Diagramas representam a modelagem de relacionamento mais específicas do sistema.

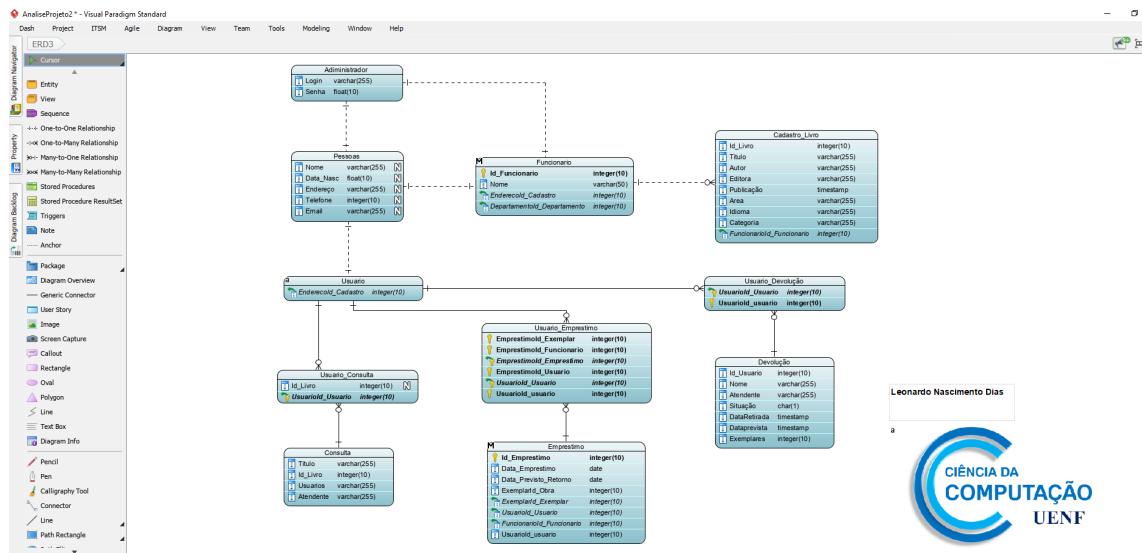


Figura 3.15: ER03 - Empréstimo / Devolução

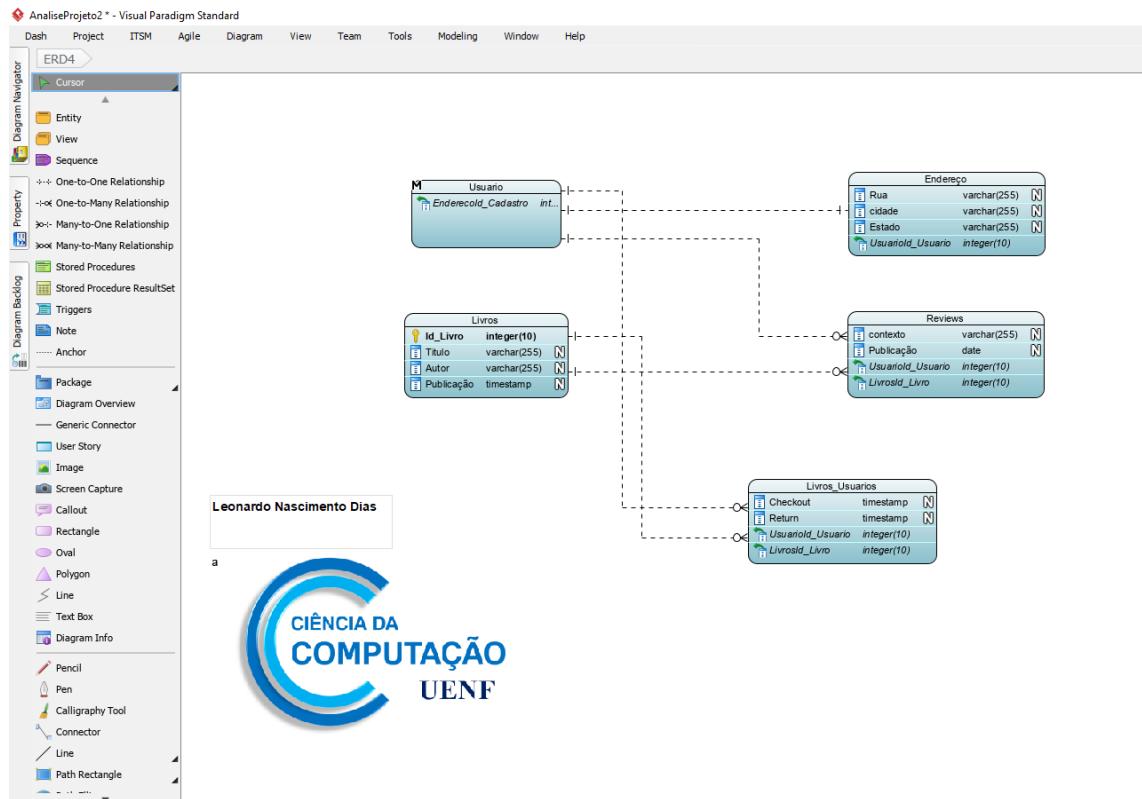


Figura 3.16: ER04 - Review

3.3 Diagramas de Classes

Um diagrama de classes é uma representação da estrutura e relações das classes que servem de modelo para objetos.

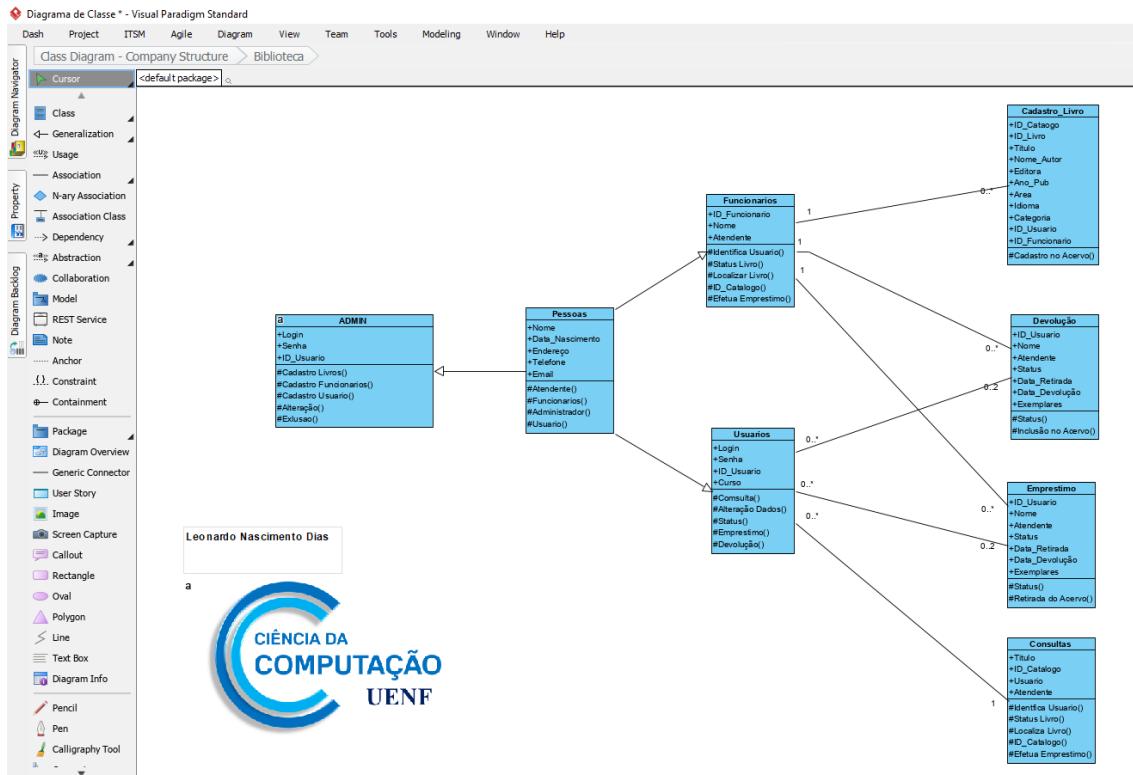


Figura 3.17: Diagrama de Classes

3.4 Diagramas Casos de uso

O diagrama de caso de uso descreve a funcionalidade proposta para um novo sistema que será projetado, é uma excelente ferramenta para o levantamento dos requisitos funcionais do sistema.

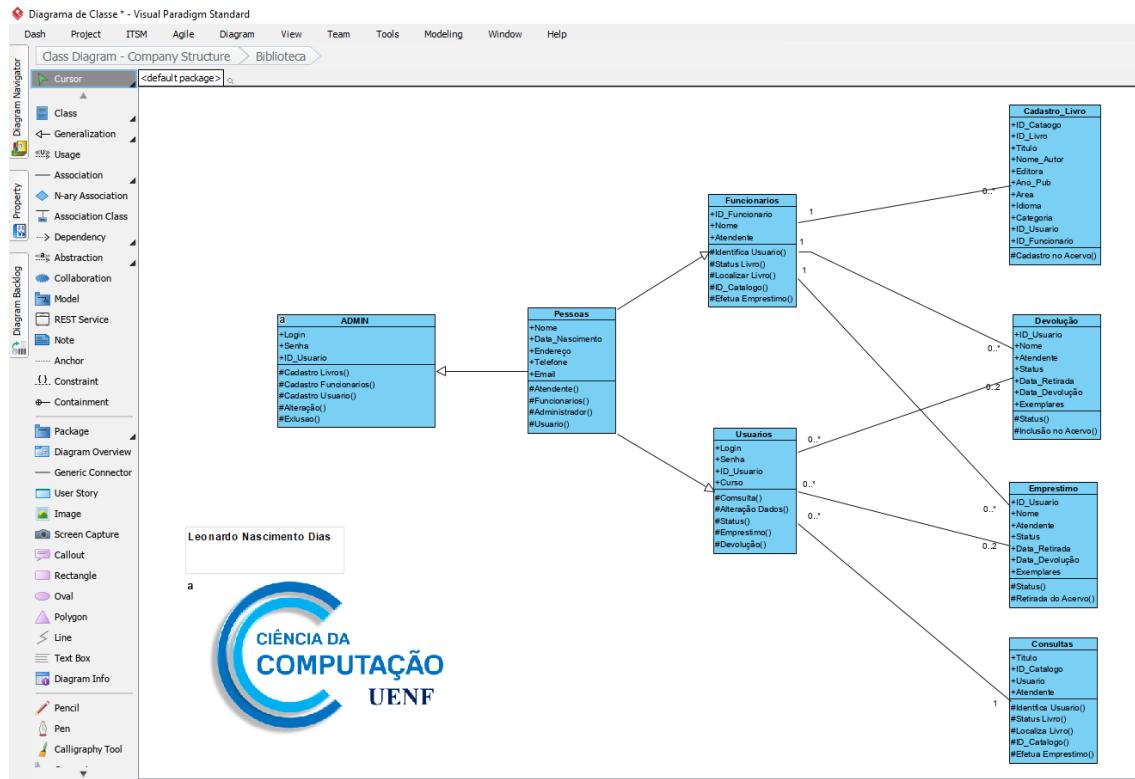


Figura 3.18: Diagrama de Casos de Uso

3.5 Diagramas Sequência

Diagrama de sequência é um diagrama usado em UML, representando a sequência de processos num programa de computador. Como um projeto pode ter uma grande quantidade de métodos em classes diferentes, pode ser difícil determinar a sequência global do comportamento.

3.5.1 Diagrama de Sequencia Login

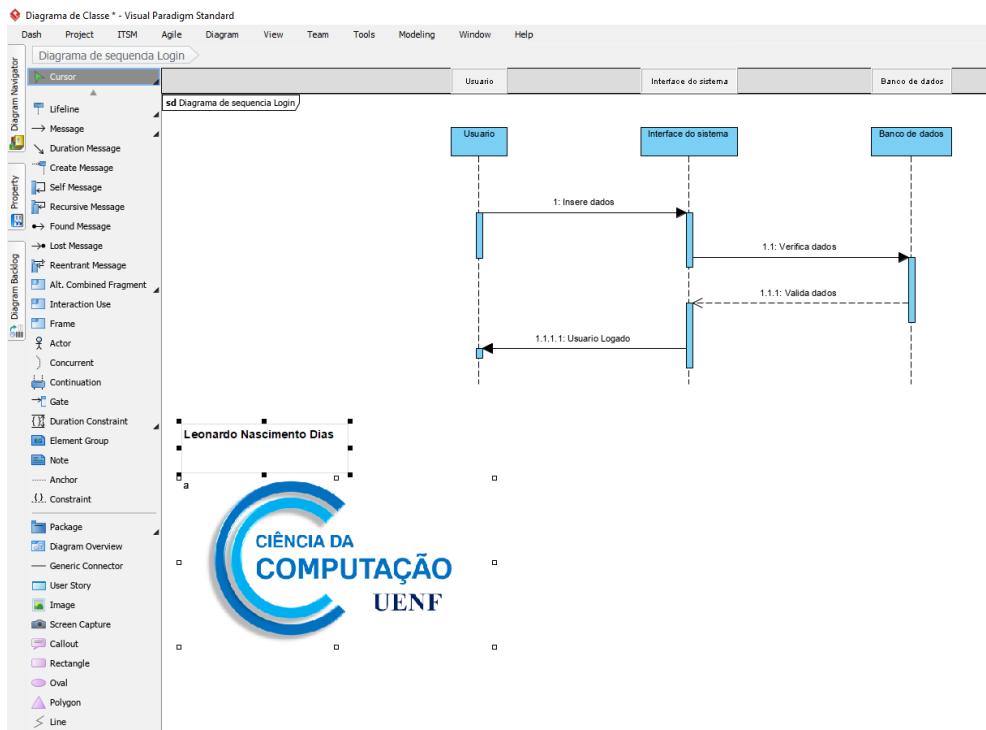


Figura 3.19: Login

3.5.2 Diagrama de Sequencia Devolução

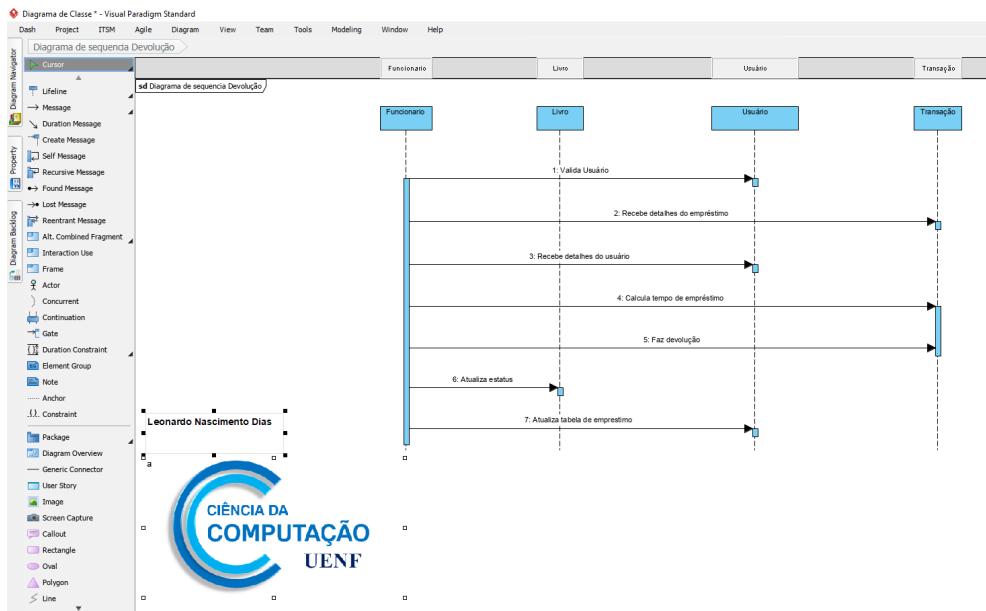


Figura 3.20: Devolução

3.5.3 Diagrama de Sequencia Emprestimo

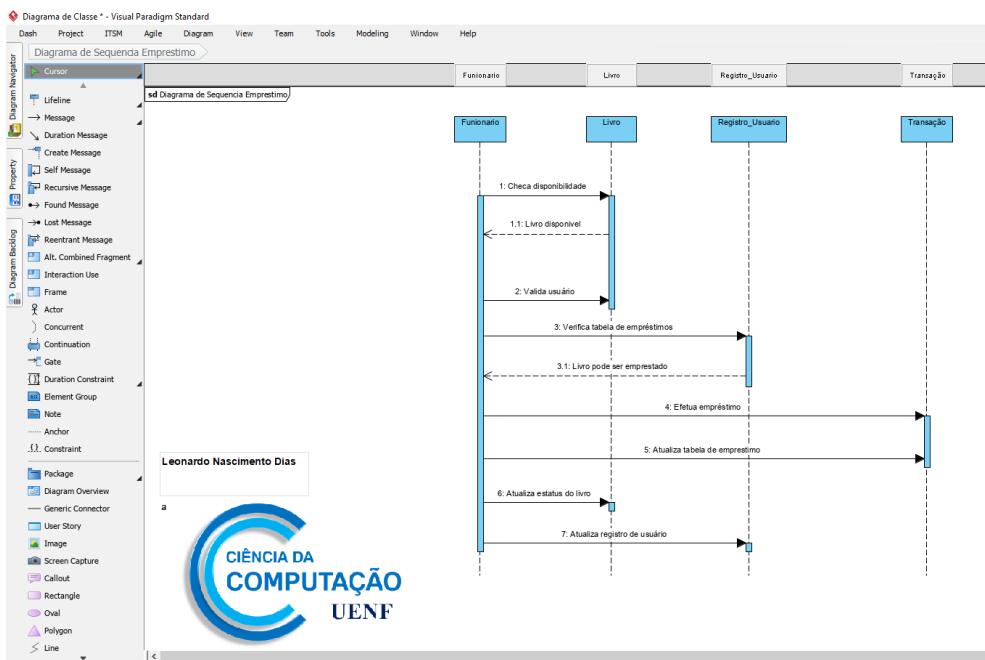


Figura 3.21: Emprestimo

3.6 Diagramas de Atividades

Um diagrama de atividade é essencialmente um gráfico de fluxo, mostrando o fluxo de controle de uma atividade para outra e serão empregados para fazer a modelagem de aspectos dinâmicos do sistema. Na maior parte, isso envolve a modelagem das etapas sequenciais em um processo computacional.

3.6.1 Diagrama de Atividades Devolução

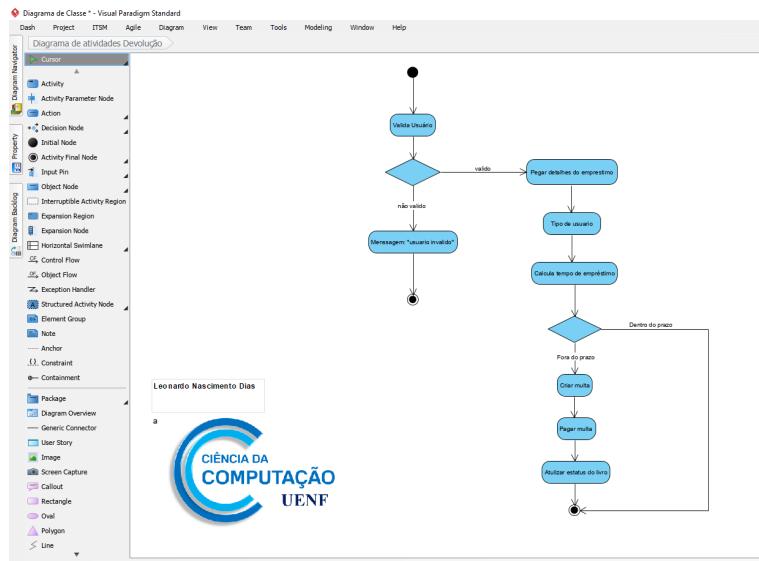


Figura 3.22: Devolução

3.6.2 Diagrama de Atividades Empréstimo

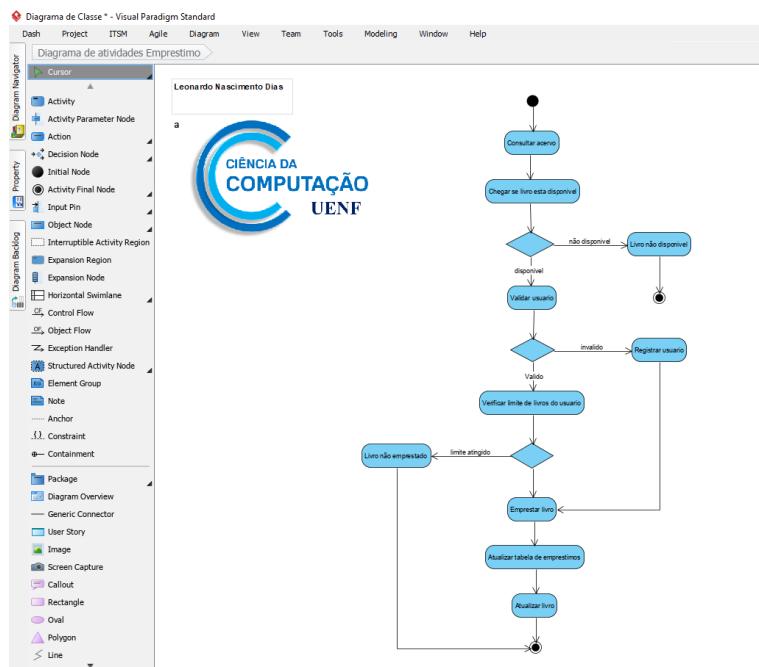


Figura 3.23: Empréstimo

3.6.3 Diagrama de Atividades Cadastro Usuário

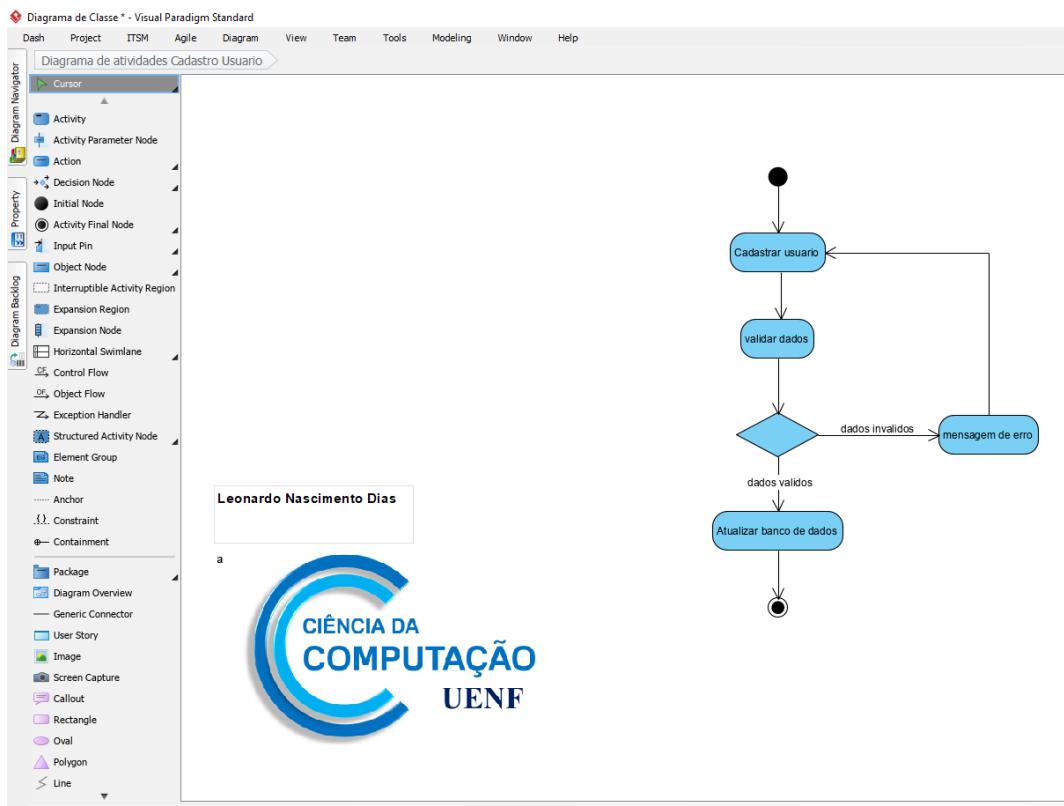


Figura 3.24: Cadastro Usuário

3.7 Diagramas Estado

Um Diagrama de Transição de Estados, ou Diagrama de Máquina de Estados, é uma representação do estado ou situação em que um objeto pode se encontrar no decorrer da execução de processos de um sistema.

3.7.1 Diagrama de Estado Login

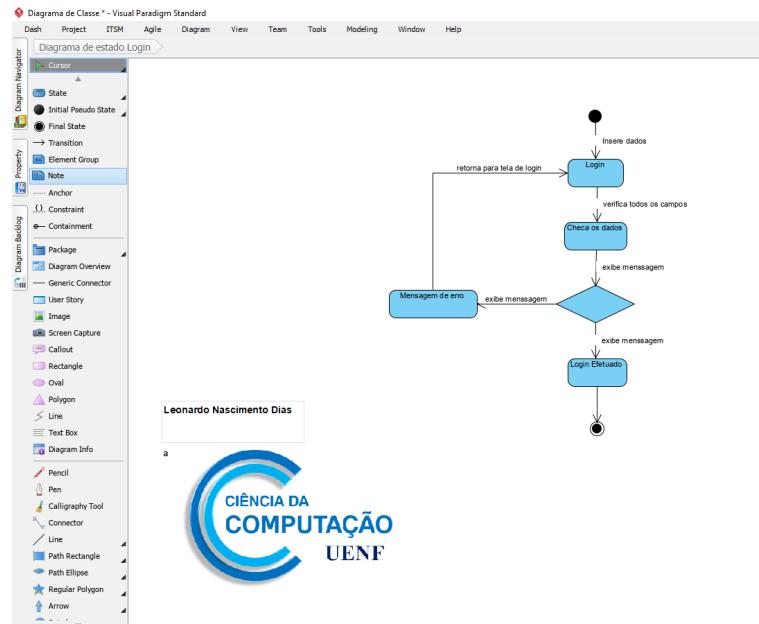


Figura 3.25: Login

3.7.2 Diagrama de Estado Consulta

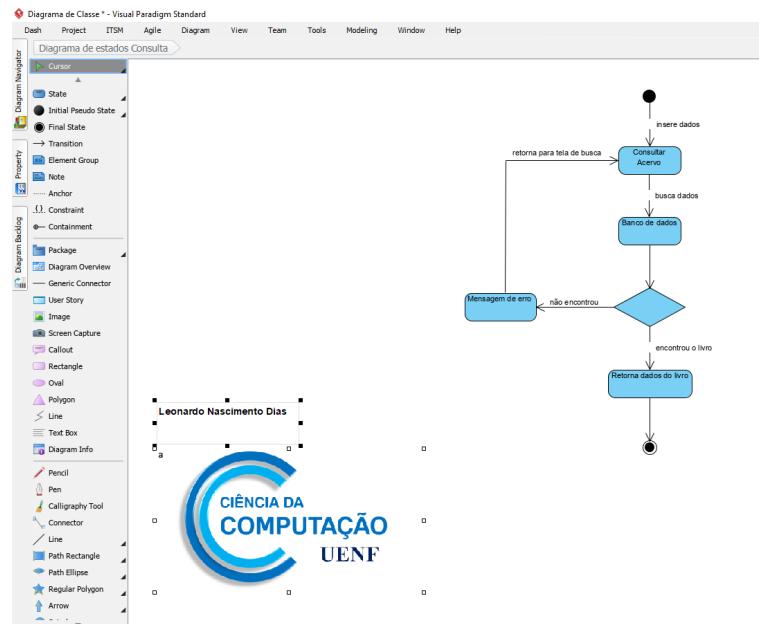


Figura 3.26: Consulta

3.7.3 Diagrama de Estado Cadastro Usuário

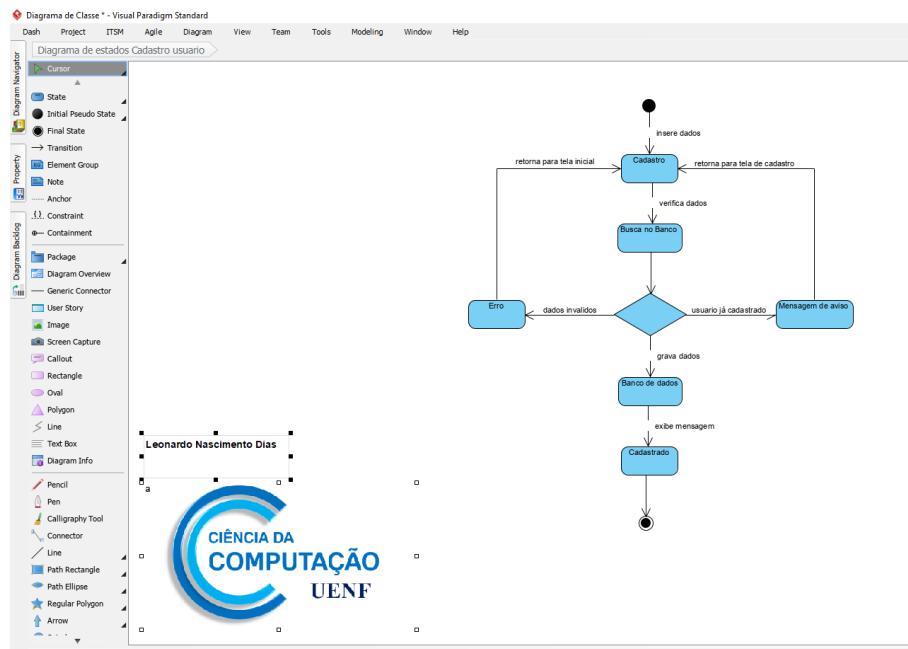


Figura 3.27: Cadastro Usuário

3.7.4 Diagrama de Estado Empréstimo

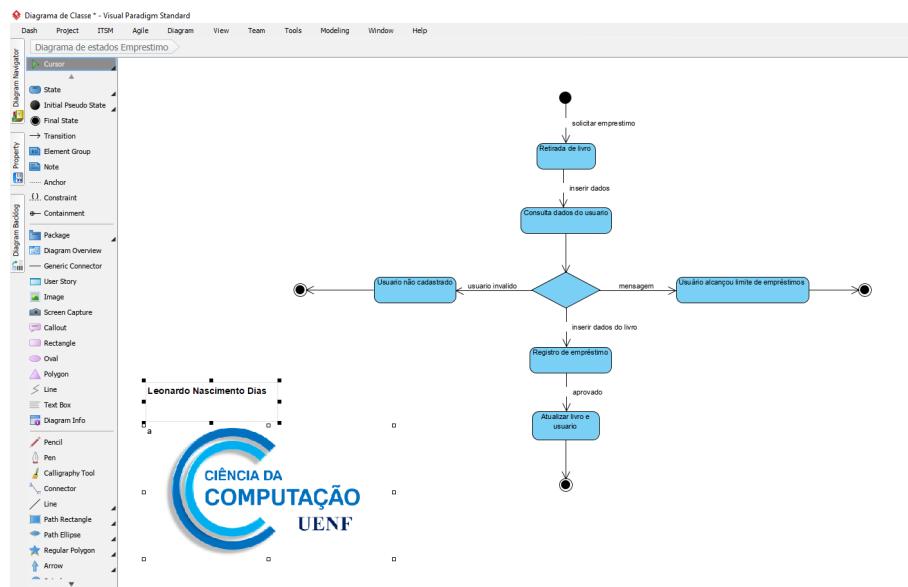


Figura 3.28: Empréstimo

3.7.5 Diagrama de Estado Devolução

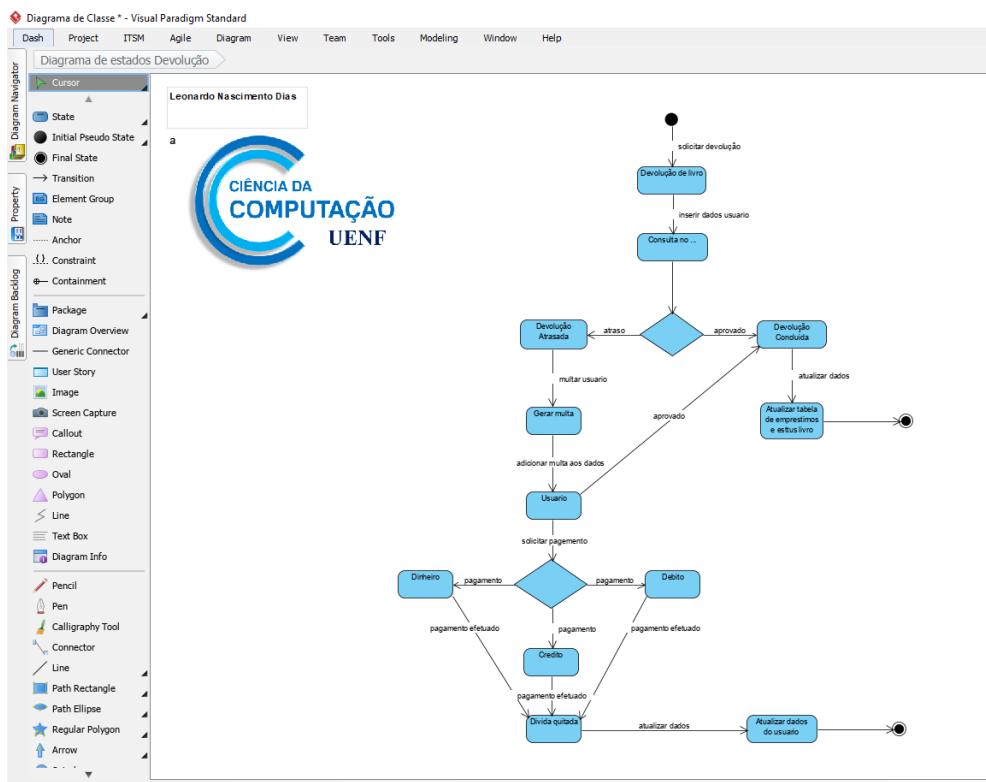


Figura 3.29: Devolução



4. Projeto do Sistema OO

Um Projeto Orientado a Objetos é um processo por meio do qual um conjunto de modelos de projeto orientados a objetos são construídos pra posteriormente, ser utilizados por programadores para escrever e testar o novo sistema sendo desenvolvido [SJB12].

4.1 Arquitetura do Sistema - Classes

- Arquitetura do sistema completo

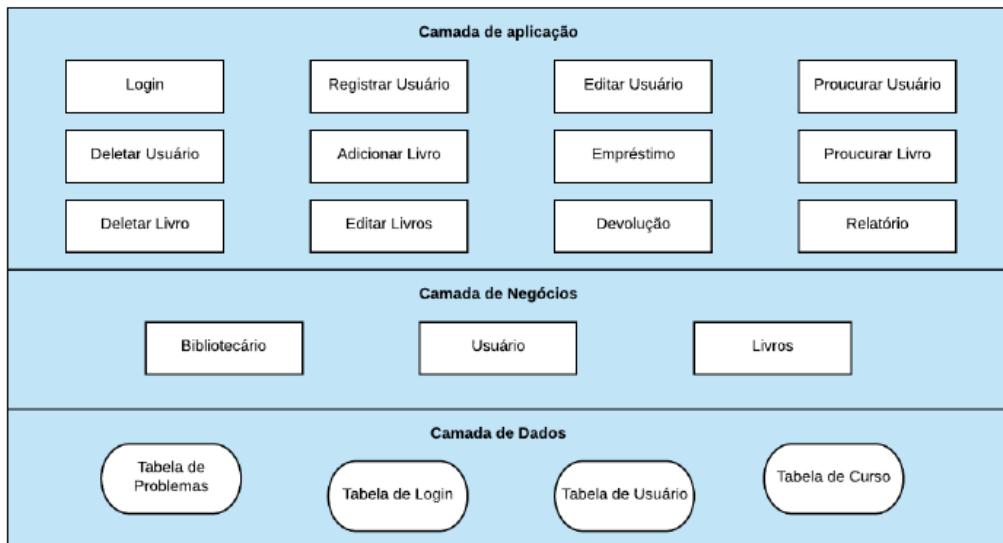


Figura 4.1: Modelo de Arquitetura de Sistema 01

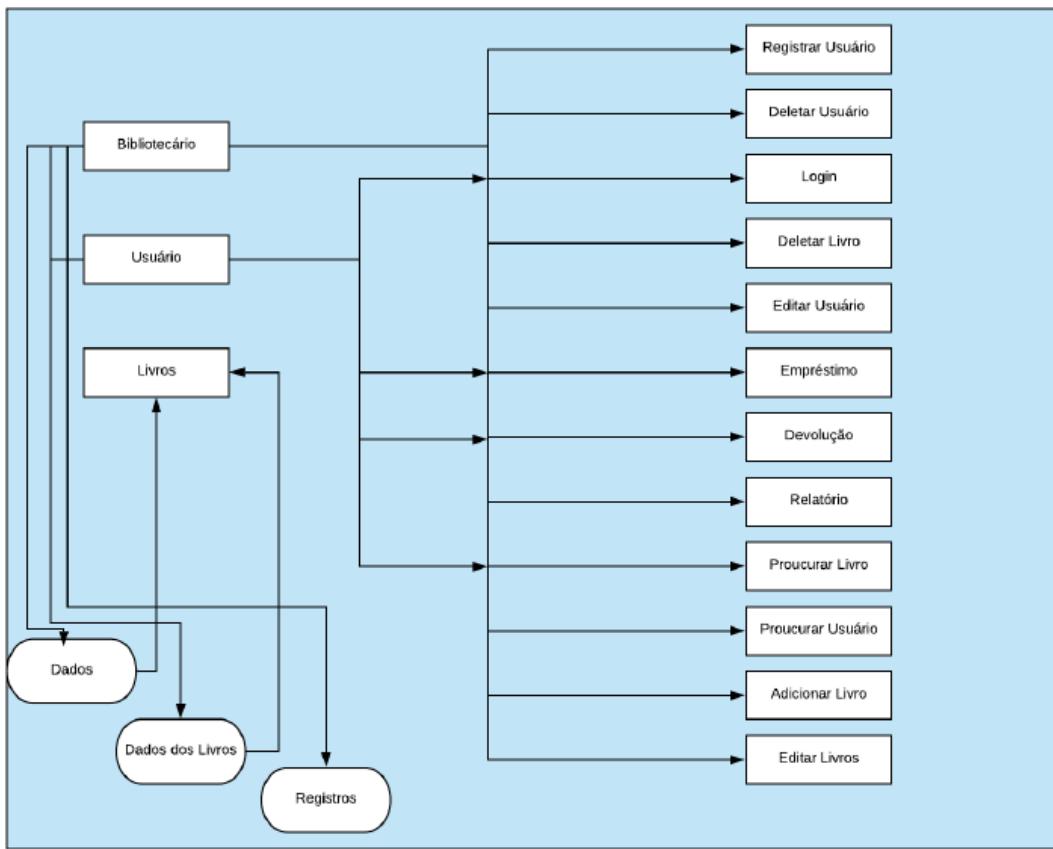


Figura 4.2: Modelo de Arquitetura de Sistema 02

- Arquitetura de hardware

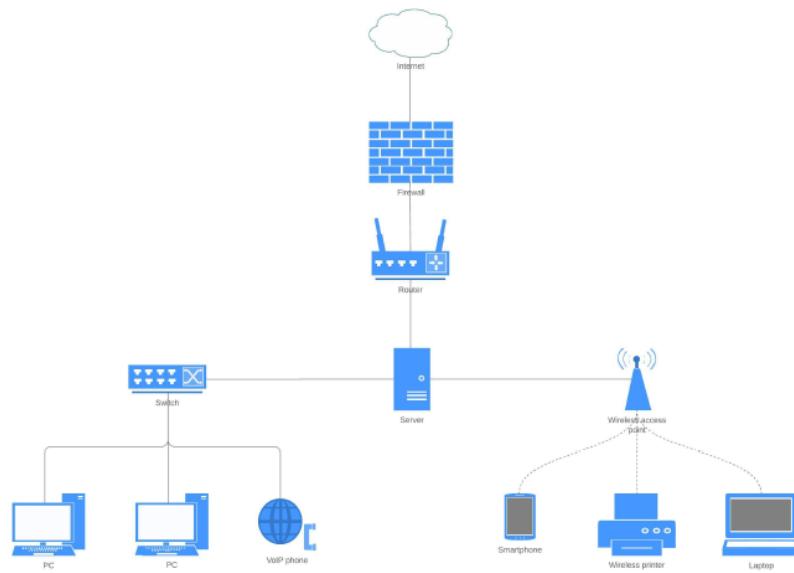


Figura 4.3: Modelo de Arquitetura de Hardware 01

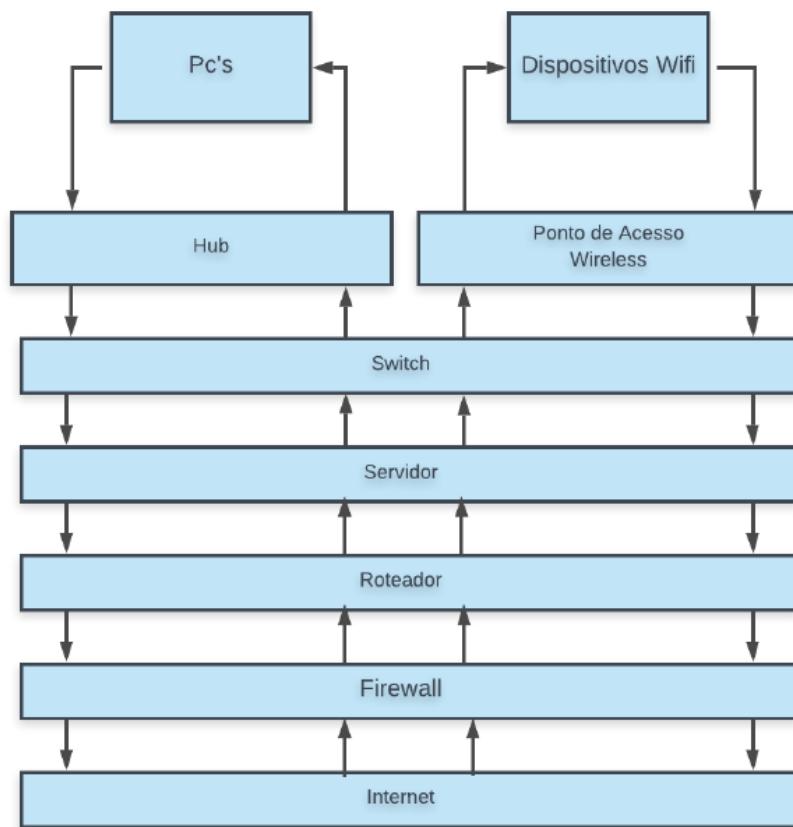


Figura 4.4: Modelo de Arquitetura de Hardware 02

- Arquitetura de software

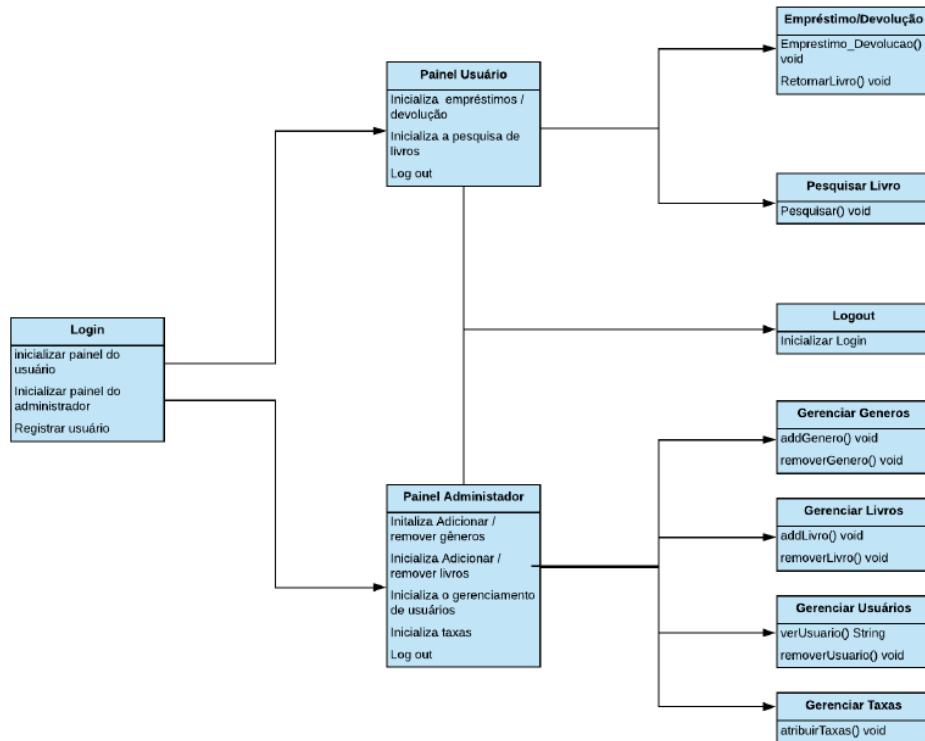


Figura 4.5: Modelo de Arquitetura de Software 01

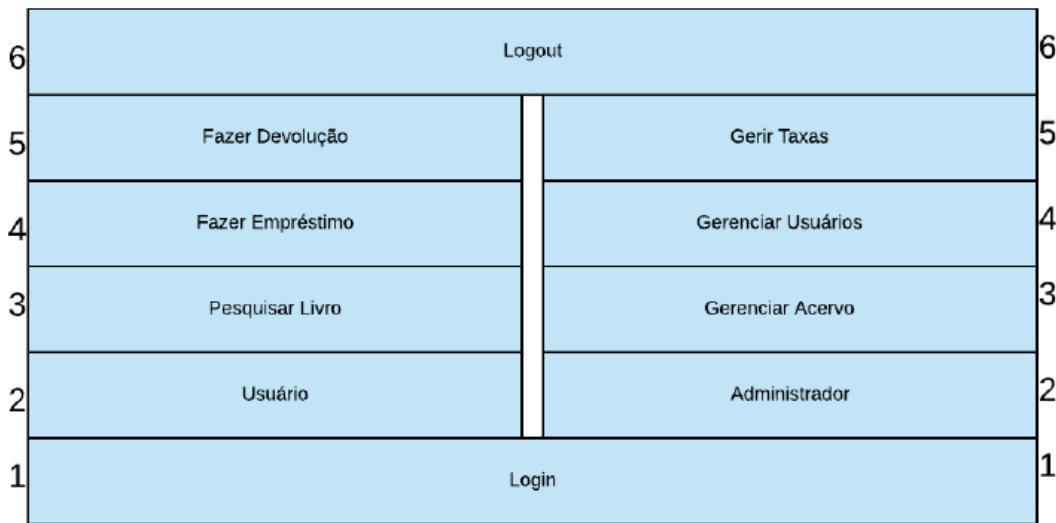


Figura 4.6: Modelo de Arquitetura de Software 02

4.2 Interfaces do Usuário

- Tela de Login (figura 4.7) feita de forma padrão, contendo campos de email e senha para o usuário preencher e acessar o sistema, contem alguns botões adicionais sendo eles e de "Ainda não sou cadastrado" que leva o usuário para tela de cadastro e "esqueci minha senha" que está somente de forma estética e sem funcionamento.

The screenshot shows a login form titled 'Login' with a light blue header bar containing a library icon and the word 'Biblioteca'. Below the title, there are two input fields: 'Email:' and 'Senha:', each with a placeholder 'Email...' and 'Senha...'. A teal-colored 'Enviar' button is positioned below the inputs. At the bottom of the form, there are two links: 'Esqueci minha senha' and 'Ainda não sou cadastrado'.

Figura 4.7: Login

- Tela de Cadastro 4.8 foi feita da mesma forma que a tela de login, contem os campos de nome, email e senha para o usuário preencher, o campo de senha foi feito para esconder a senha durante a digitação por motivos de segurança e privacidade. Possui botões para enviar o formulario de cadastro e para voltar para a pagina de login.

The screenshot shows a registration form titled 'Cadastre-se' with a light blue header bar containing a library icon and the word 'Biblioteca'. Below the title, there are three input fields: 'Nome' (Name), 'Email', and 'Senha' (Password), each with a placeholder 'Nome...', 'Email...', and 'Senha...'. The 'Senha' field includes a small icon indicating it's a password field. A teal-colored 'Enviar' button is positioned below the inputs. At the bottom of the form, there is a link 'Já tenho uma conta'.

Figura 4.8: Cadastro

- Tela Inicial 4.9 do sistema contendo os livros cadastrados pelo usuário, possibitando acessar cada livro e um menu contendo as principais funcionalidades do sistema sendo elas Opções, Seus empréstimos, Sobre, Sair e a quantidade de livros cadastrados

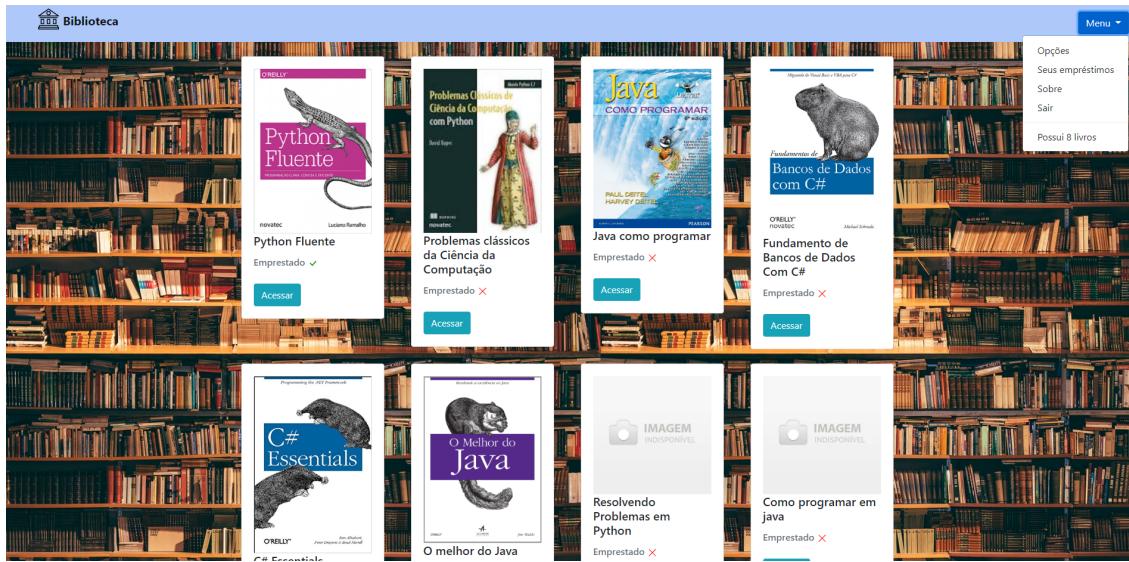


Figura 4.9: Home

- Tela 4.10 para mostrar lista de empréstimos do usuário, contem todos os livros que o usuário pegou emprestado, mostrando a data em que foi pego e a data que foi devolvido.

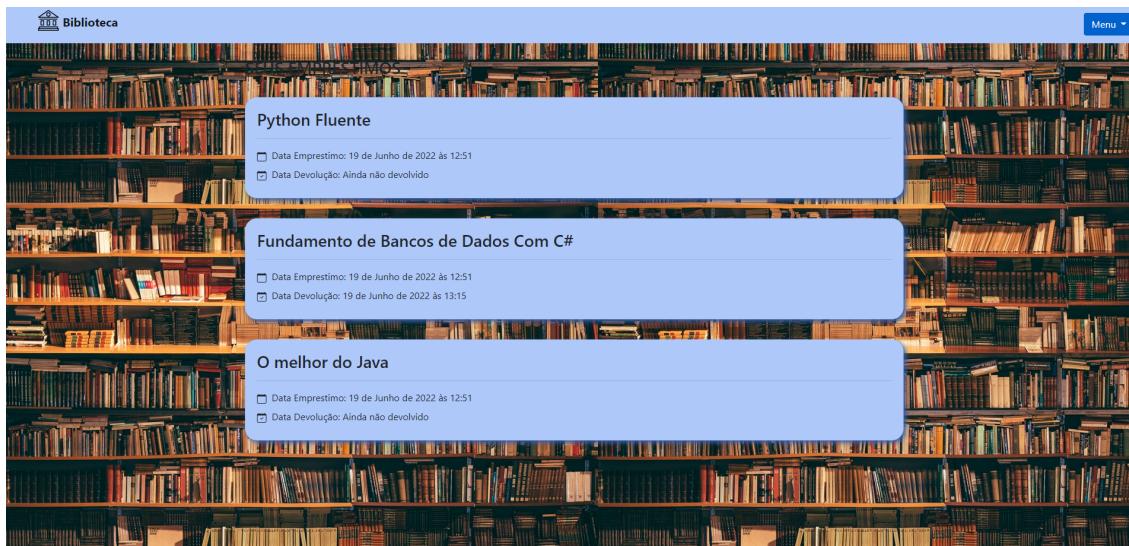


Figura 4.10: Empréstimos usuário

- Tela 4.11 Acessa os dados de um livro e permite que o usuário altere alguns dados ou exclua o livro, também possui uma tabela para o historio de empréstimo que mostrar dos os usuários que pegaram aquele determinado livro, mostrando a data em que foi pego e a data de devolução, tambem permite avaliar a pessoa que pegou o livro, pelo tempo de empréstimo ou situação física do livro.

Figura 4.11: Dados Livro

4.2.1 Interfaces do Usuário - Modais

Os modais foram utilizados usando bootstrap que é um framework web com código-fonte aberto para desenvolvimento de componentes de interface e front-end para sites e aplicações web usando HTML, CSS, baseado em modelos de design para a tipografia, melhorando a experiência do usuário em um site amigável e responsivo. Essas informações foram colocada em 'base.html' e foi herdado pelos outros templates com algumas modificações assim como o modal sobre que será apresentado a seguir.

Na Código a seguir é mostrado como foi organizado os botões dentro do modal para ficar mais organizado e responsivo.

```
<!-- Modal -->
{%
    if usuario_logado %}
<div class="modal fade" id="exampleModal" tabindex="-1" role="dialog" aria-labelledby="exampleModalLabel"
    aria-hidden="true">
    <div class="modal-dialog" role="document">
        <div class="modal-content">
            <div class="modal-header">
                <h5 class="modal-title" id="exampleModalLabel">Cadastrar / Emprestar / Devolver</h5>
                <button type="button" class="close" data-dismiss="modal" aria-label="Close">
                    <span aria-hidden="true">&times;</span>
                </button>
            </div>
            <div class="modal-body">

                <button style="margin-left:10px" onclick="mostrar_form(2)" class="btn btn-info">Livro</button>
                <button style="margin-left:10px" onclick="mostrar_form(1)" class="btn btn-info">Categoria</button>
                <button style="margin-left:10px" onclick="mostrar_form(3)" class="btn btn-info">Empréstimo</button>
                <button style="margin-left:10px" onclick="mostrar_form(4)" class="btn btn-info">Devolução</button>

                <br>
            </div>
        </div>
    </div>
</div>
```

A seguir mais uma parte do código mostrando a forma de como ele está responsivo onde de acordo com o botão apertado pelo usuário, a ação vai setar o formulário referente ao botão, e mudará os outros para 'hidden' no código fonte.

```

<div style="display:none;" id="livro">
    <form action="{% url 'cadastrar_livro' %}" method="POST" enctype="multipart/form-data">
        {%csrf_token %}
        <table>
            {{form.as_table}}
        </table>
        <input type="submit" class="btn btn-success" value="Cadastrar">
    </form>
</div>

<div style="display:none;" id="categoria">
    <form action="{% url 'cadastrar_categoria' %}" method="POST">
        {%csrf_token %}
        <table>
            {{ form_categoria.as_table }}
        </table>
        <input type="hidden" name="usuario" value="{{usuario_logado}}">
        <input type="submit" class="btn btn-success" value="Salvar">
    </form>
</div>

```

Dentro do modal ele tambem faz alguma verificações para autenticar o usuario e conferir os ids das tabelas para não gerar nenhum erro durante a execução do sistema.

- Modal 4.12 cadastro de Livros possui os campos para o usuário preencher e cadastrar o livro desejado, possibilitando que o usuário escolha uma imagem

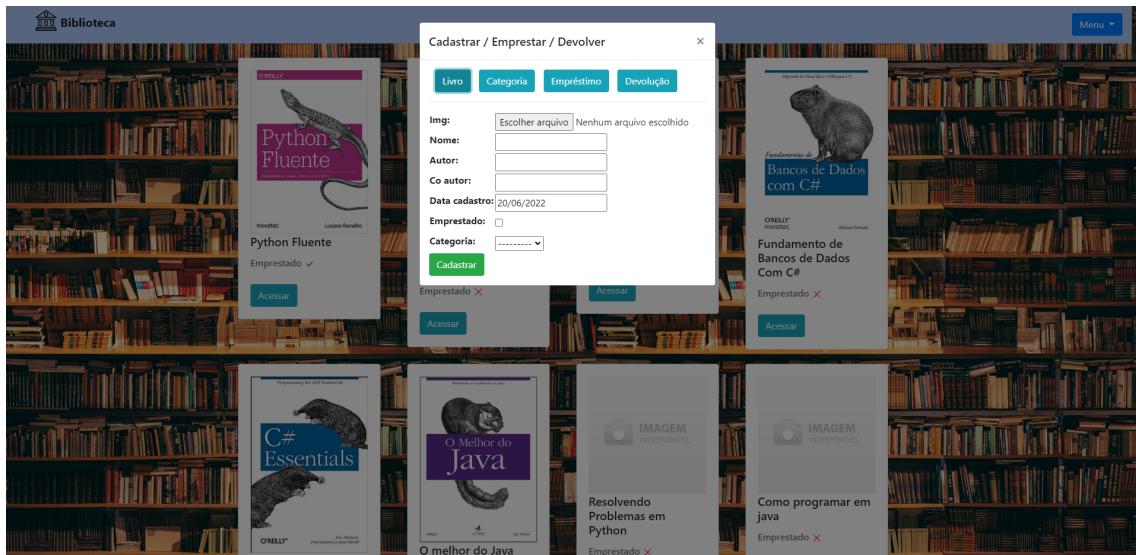


Figura 4.12: Cadastro Livro

- Modal 4.13 cadastro de Categoria possui os campos para o usuário preencher e cadastrar uma categoria, possuindo os campos de nome e descrição da categoria.

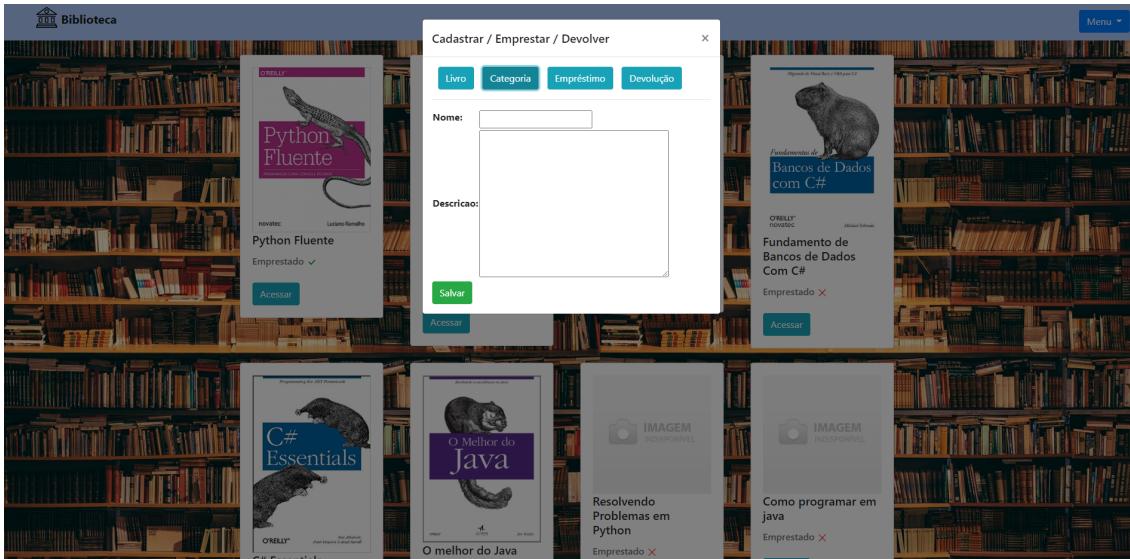


Figura 4.13: Cadastro Categoria

- Modal 4.14 Empréstimo possui dois botões para o usuário escolher se quer fazer o empréstimo para alguém já cadastrado no sistema ou para uma pessoa sem cadastro. Possui os campos para escolher quem vai receber o livro e qual livro será emprestado

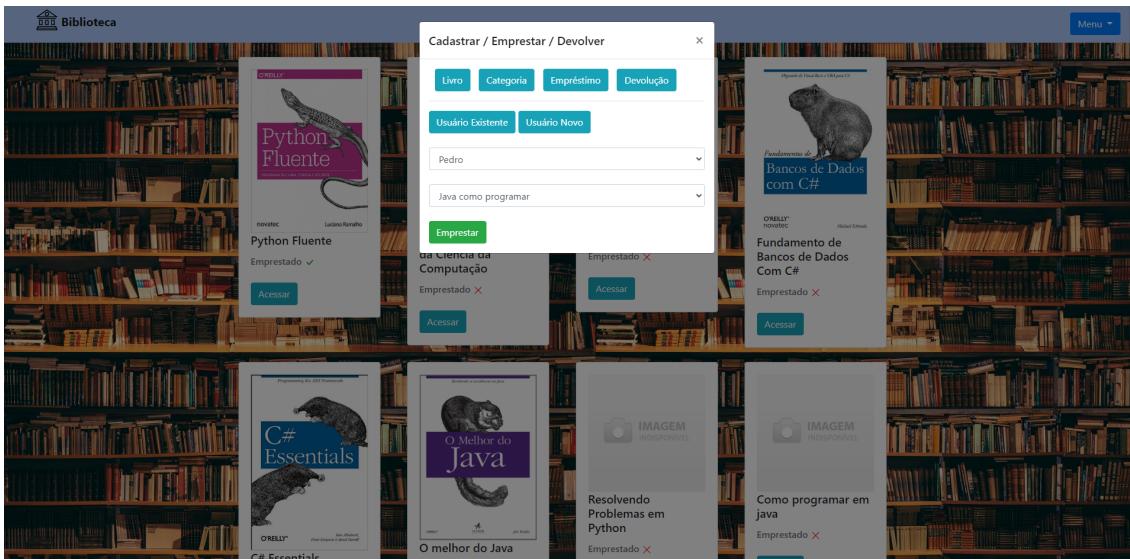


Figura 4.14: Empréstimos

- Modal 4.15 Devolução permite que o usuário devolva um livro que ele já recebeu de volta.

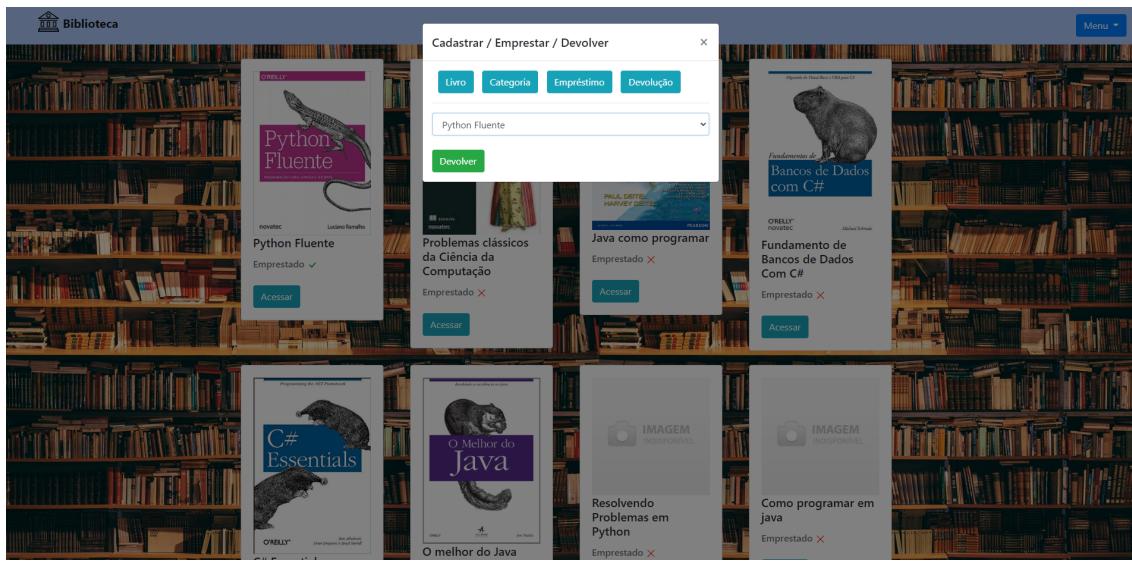


Figura 4.15: Devolução

- Modal 4.16 Sobre exibe informações do sistema.

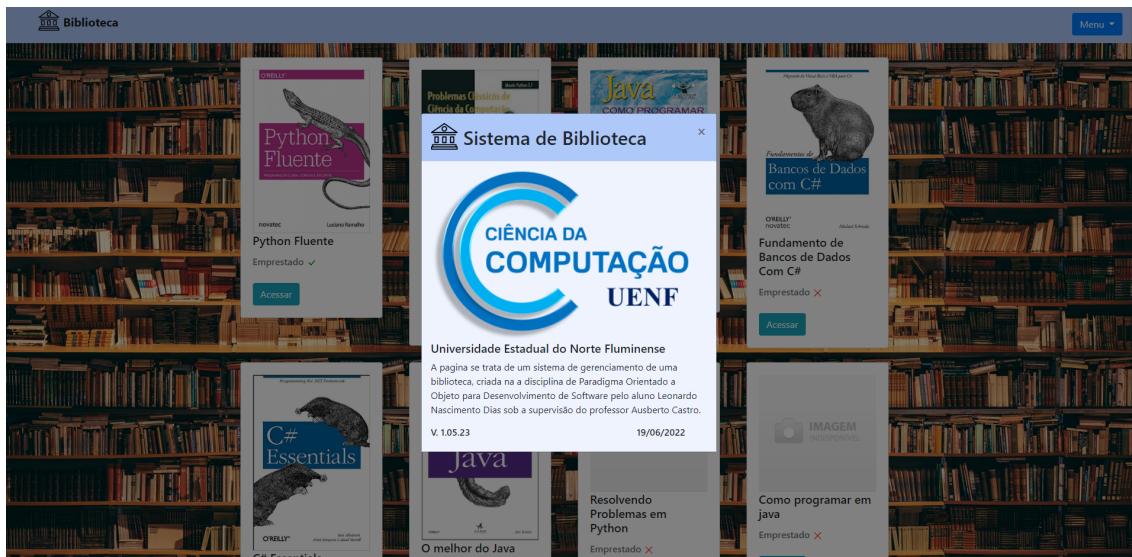


Figura 4.16: Sobre

4.3 Tabelas de Dados

Estruturas de dados que fazem parte da base de dados: cada uma com seus atributos e chaves principais e secundárias. Foi utilizado o software DB Brownsr para acessar as tabelas do banco de forma mais legivel.

Figura 4.17: Todas as tabelas do banco

The screenshot shows the DB Browser for SQLite interface. The title bar indicates the database is located at D:\Aulas\POO\Projeto\Biblioteca\db.sqlite3. The menu bar includes Arquivo, Editar, Exibir, Ferramentas, and Ajuda. The toolbar includes Novo banco de dados, Abrir banco de dados, Escrever modificações, Reverter modificações, Abrir projeto, Salvar projeto, Anexar banco de dados, and Fechar banco de dados. Below the toolbar are buttons for Criar tabela, Criar índice, and Imprimir. A navigation bar at the top has Estrutura do banco de dados, Navegar dados, Editar pragmas, and Executar SQL. The main area is divided into two panes. The left pane displays a tree view of the database structure:

- Tabelas (15)**
 - auth_group
 - auth_group_permissions
 - auth_permission
 - auth_user
 - auth_user_groups
 - auth_user_permissions
 - django_admin_log
 - django_content_type
 - django_migrations
 - django_session
 - livro_categoria
 - livro_emprestimos
 - livro_livros
 - sqlite_sequence
 - usuarios_usuario
- Índices (20)**
- Vistas (0)**
- Gatilhos (0)**

The right pane shows the schema for the selected table, which is currently 'livro_categoria'. The schema is as follows:

```

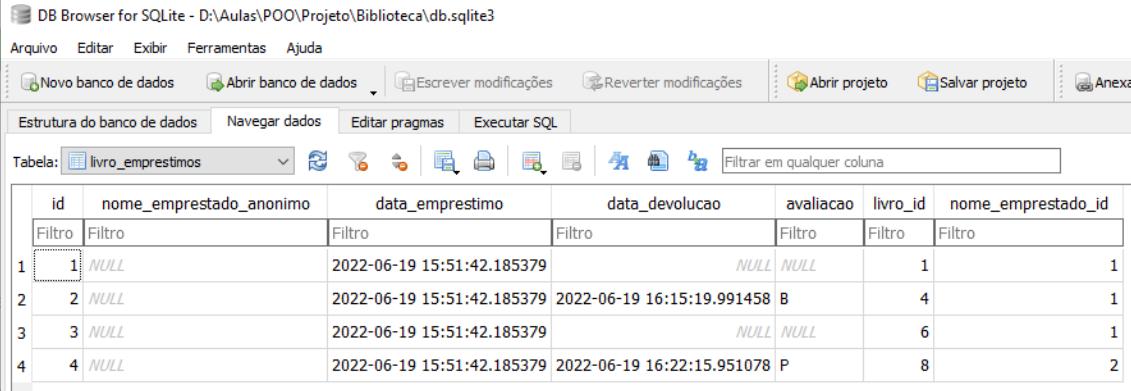
CREATE TABLE "livro_categoria" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "nome" varchar(30) NOT NULL);
CREATE TABLE "livro_emprestimos" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "livro_id" integer NOT NULL, "usuario_id" integer NOT NULL, "data_emprestimo" datetime NOT NULL, "data_devolucao" datetime NULL);
CREATE TABLE "livro_livros" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "img" varchar(100) NULL, "nome" varchar(100) NOT NULL);
CREATE TABLE "sqlite_sequence(name,seq)" ("name" varchar(30) NOT NULL, "seq" integer NOT NULL);
CREATE TABLE "usuarios_usuario" ("id" integer NOT NULL PRIMARY KEY AUTOINCREMENT, "nome" varchar(30) NOT NULL);
  
```

Figura 4.18: Tabela de Categorias

The screenshot shows the DB Browser for SQLite interface with the 'livro_categoria' table selected. The title bar and menu bar are identical to Figura 4.17. The toolbar includes Novo banco de dados, Abrir banco de dados, Escrever modificações, Revert, and other standard icons. The navigation bar includes Estrutura do banco de dados, Navegar dados, Editar pragmas, and Executar SQL. The main area shows the 'livro_categoria' table data:

	id	nome	descricao	usuario_id
1	1	Python	Livros referente a linguagem de programação Python	1
2	2	Java	Livros referentes a linguagem de programação Java	1
3	3	C#	Livros referentes a linguagem de programação C#	1

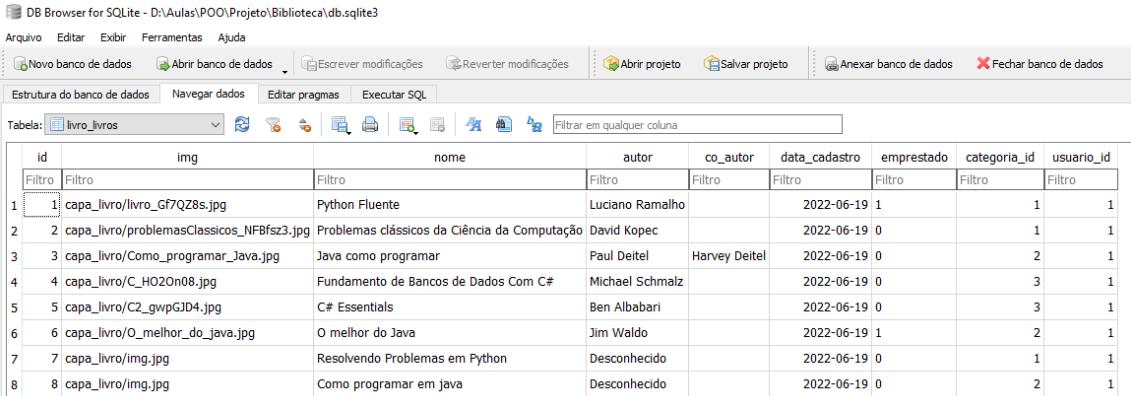
Figura 4.19: Tabela de Empréstimos



The screenshot shows the DB Browser for SQLite interface with the database file 'db.sqlite3' open. The 'livro_emprestimos' table is selected. The table has columns: id, nome_emprestado_anonimo, data_emprestimo, data_devolucao, avaliacao, livro_id, and nome_emprestado_id. There are four rows of data:

	id	nome_emprestado_anonimo	data_emprestimo	data_devolucao	avaliacao	livro_id	nome_emprestado_id
1	1	NULL	2022-06-19 15:51:42.185379	NULL	NULL	1	1
2	2	NULL	2022-06-19 15:51:42.185379	2022-06-19 16:15:19.991458	B	4	1
3	3	NULL	2022-06-19 15:51:42.185379	NULL	NULL	6	1
4	4	NULL	2022-06-19 15:51:42.185379	2022-06-19 16:22:15.951078	P	8	2

Figura 4.20: Tabela de Livros



The screenshot shows the DB Browser for SQLite interface with the database file 'db.sqlite3' open. The 'livro_livros' table is selected. The table has columns: id, img, nome, autor, co_autor, dataCadastro, emprestado, categoria_id, and usuario_id. There are eight rows of data:

	id	img	nome	autor	co_autor	dataCadastro	emprestado	categoria_id	usuario_id
1	1	capa_livro/livro_Gf7QZ8s.jpg	Python Fluente	Luciano Ramalho		2022-06-19	1	1	1
2	2	capa_livro/problemasClassicos_NFBfsz3.jpg	Problemas clássicos da Ciência da Computação	David Kopec		2022-06-19	0	1	1
3	3	capa_livro/Como_programar_Java.jpg	Java como programar	Paul Deitel	Harvey Deitel	2022-06-19	0	2	1
4	4	capa_livro/C_H02n08.jpg	Fundamento de Bancos de Dados Com C#	Michael Schmalz		2022-06-19	0	3	1
5	5	capa_livro/C2_gwpGJD4.jpg	C# Essentials	Ben Albabari		2022-06-19	0	3	1
6	6	capa_livro/O_melhor_do_java.jpg	O melhor do Java	Jim Waldo		2022-06-19	1	2	1
7	7	capa_livro/img.jpg	Resolvendo Problemas em Python	Desconhecido		2022-06-19	0	1	1
8	8	capa_livro/img.jpg	Como programar em java	Desconhecido		2022-06-19	0	2	1

Figura 4.21: Tabela de Usuários

The screenshot shows the DB Browser for SQLite interface with the title bar "DB Browser for SQLite - D:\Aulas\POO\Projeto\Biblioteca\db.sqlite3". The menu bar includes Arquivo, Editar, Exibir, Ferramentas, and Ajuda. The toolbar has buttons for Novo banco de dados, Abrir banco de dados, Escrever modificações, Reverter modificações, and others. Below the toolbar, tabs include Estrutura do banco de dados, Navegar dados, Editar pragmas, and Executar SQL. The Estrutura tab is selected. A dropdown menu "Tabela:" shows "usuarios_usuario". The main area displays a table with columns id, nome, email, and senha. The data is as follows:

	id	nome	email	senha
1	1	Leonardo	leonardondm@hotmail.com	15e2b0d3c33891ebb0f1ef609ec419420c20e320c...
2	2	Leonardo2	leonardondm2@hotmail.com	15e2b0d3c33891ebb0f1ef609ec419420c20e320c...
3	3	Pedro	pedro@gmail.com	15e2b0d3c33891ebb0f1ef609ec419420c20e320c...
4	4	Leonardo3	BigTazs123@hotmail.com	1a7f3e6a41a9e582ce211d013ac6216de122e002...


```
tic void Main(string[] args)  
  
for (int i = 0; i < 50; i++)  
{  
    Thread mythread = new Thread(new T  
    mythread.Start();  
  
    Task.Run(() =>  
    {  
        Console.WriteLine("starting task in th  
        Thread.CurrentThread.ManagedThreadId);
```



5. Implementação do Sistema OO

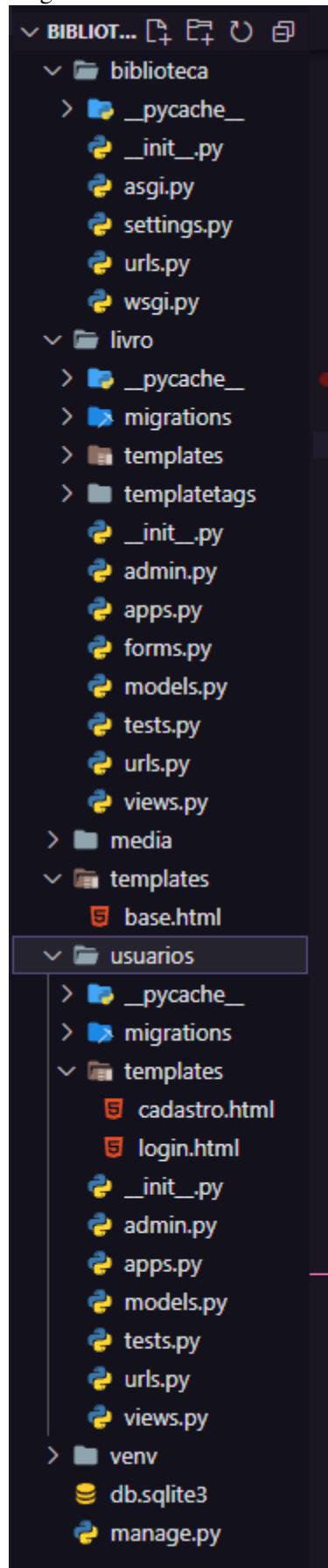
Nesse capítulo será mostrado como o sistema foi dividido e como foi implementado algumas partes importantes. Tal divisão é importante para respeitar a arquitetura de três níveis e isolar o Sistema de possíveis riscos. Também é uma boa estrutura organizacional, pois facilita o entendimento do sistema de forma mais clara e torna mais fácil encontrar erros.

5.1 Programação

Estrutura de pastas sendo mostrado na figura 5.1.

- A pasta biblioteca contem as configurações iniciais do django assim como as urls usadas no sistema.
- A pasta livro contem toda programação referente as funcionalidades que são relacionadas aos livros, os arquivos mais importantes são 'models.py' que contem as classes, 'urls' contendo as urls do sistema para redirecionamento e 'views.py' que contem as funções implementas.
- A pasta media contem imagens usadas durante o desenvolvimento.
- templates contem o modelo das paginas HTML que o sistema usa.
- A pasta usuário contem a programação referentes as funcionalidades dos usuarios como o login e o cadastro, Tambem contem os arquivos 'models.py' contendo as classes, 'views.py' possuindo as funções e 'urls.py' que possui as urls do sistema para redirecionamento.

Figura 5.1: Tabela de Usuários



5.1.1 Classes

- Classe Usuario: contem os atributos nome, email e senha que serao usados para cadastro e para o login.

```
class Usuario(models.Model):
    nome = models.CharField(max_length=30)
    email = models.EmailField()
    senha = models.CharField(max_length=64)
```

- Classe Categoria: contem os atributos nome, descrição e usuário que será a pessoa que esta criando a categoria.

```
class Categoria(models.Model):
    nome = models.CharField(max_length=30)
    descricao = models.TextField()
    usuario = models.ForeignKey(Usuario, on_delete=models.DO_NOTHING)
```

- Classe Livros: contem os atributos img para o usuário colocar uma imagem na hora de cadastrar um livro e caso ele não escolha uma será posto uma imagem padrão, nome, autor, co-autor, data de cadastro, se esta emprestado, categoria e usuário.

```
class Livros(models.Model):
    img = models.ImageField(upload_to='capa_livro', null=True, blank=True, default='capa_livro/img.jpg')
    nome = models.CharField(max_length=100)
    autor = models.CharField(max_length=30)
    co_autor = models.CharField(max_length=30, blank=True)
    dataCadastro = models.DateTimeField(default=date.today)
    emprestado = models.BooleanField(default=False)
    categoria = models.ForeignKey(Categoria, on_delete=models.DO_NOTHING)
    usuario = models.ForeignKey(Usuario, on_delete=models.DO_NOTHING)
```

- Classe Emprestimos: contem os atributos relacionas ao emprestimo de um livro.

```
class Emprestimos(models.Model):
    nome_emprestado = models.ForeignKey(Usuario, on_delete=models.DO_NOTHING, blank=True, null=True)
    nome_emprestado_anonimo = models.CharField(max_length=30, blank=True, null=True)
    data_emprestimo = models.DateTimeField(default=datetime.datetime.now())
    data_devolucao = models.DateTimeField(blank=True, null=True)
    livro = models.ForeignKey(Livros, on_delete=models.SET_NULL, null=True)
    avaliacao = models.CharField(max_length=1, choices=choices, null=True)
```

5.1.2 Códigos Importantes

- Validação de Login (Figura 5.2) recebe os valores inseridos pelo usuario no momento do login e faz a comparação com o banco de dados para autenticar a entrada no sistema.

```

def valida_login(request):
    email = request.POST.get('email')
    senha = request.POST.get('senha')
    senha = sha256(senha.encode()).hexdigest()

    usuario = Usuario.objects.filter(senha=senha, email=email)

    if len(usuario) == 0:
        return redirect('/auth/login/?status=1')
    elif len(usuario) == 1:
        request.session['usuario'] = usuario[0].id
        return redirect ('/livro/home/')

```

Figura 5.2: Código Autenticação Login

- Validação de Cadastro (Figura 5.3) recebe valores inseridos pelo usuário, autentica se os campos de email e nome não estão vazios, verifica se a senha tem mais que oito caracteres e confere se o usuário já não existe, se algo estiver errado ele retorna um erro especificando o problema, caso esteja tudo certo ele salva o novo usuário no banco de dados.

```

def valida_cadastro(request):
    nome = request.POST.get('nome')
    senha = request.POST.get('senha')
    email = request.POST.get('email')

    usuario = Usuario.objects.filter(email=email)

    if len(nome.strip()) == 0 or len(email.strip()) == 0:
        return redirect('/auth/cadastro/?status=1')

    if len(senha) < 8:
        return redirect('/auth/cadastro/?status=2')

    if len(usuario) > 0:
        return redirect('/auth/cadastro/?status=3')

    try:
        senha = sha256(senha.encode()).hexdigest()
        usuario = Usuario(nome=nome, senha=senha, email=email)
        usuario.save()

        return redirect('/auth/cadastro/?status=0')
    except:
        return redirect('/auth/cadastro/?status=4')

```

Figura 5.3: Código Autenticação Cadastro

- Ver dados de um livro (Figura 5.4) busca o id do usuário e id do livro para conferir se o livro pertence ao usuário que está utilizando o sistema evitando que um usuário diferente tente achar e modificar livros que não o pertencem. Caso as informações estiverem corretas abre o

form contendo as informações do livro e do historico de emprestimo, exibindo para o usuário e permitindo fazer atualizações ou excluir o livro.

```
def ver_livros(request, id):
    if request.session.get('usuario'):
        livro = Livros.objects.get(id = id)
        if request.session.get('usuario') == livro.usuario.id:
            usuario = Usuario.objects.get(id=request.session['usuario'])
            categoria_livro = Categoria.objects.filter(usuario=request.session.get('usuario'))
            emprestimos = Emprestimos.objects.filter(livro = livro)

            form = CadastroLivro()
            form.fields['usuario'].initial = request.session['usuario']
            form.fields['categoria'].queryset = Categoria.objects.filter(usuario=usuario)
            form_categoria = CategoriaLivro()

            usuarios = Usuario.objects.all()
            livros = Livros.objects.filter(usuario_id=request.session.get('usuario'))

            livros_emprestar = Livros.objects.filter(usuario=usuario).filter(emprestado=False)
            livros_emprestados = Livros.objects.filter(usuario=usuario).filter(emprestado = True)

            return render(request, 'ver_livro.html',{'livro': livro,
                                                    'categoria_livro':categoria_livro,
                                                    'emprestimos':emprestimos,
                                                    'usuario_logado': request.session.get('usuario'),
                                                    'form': form,
                                                    'id_livro': id,
                                                    'form_categoria': form_categoria,
                                                    'usuarios':usuarios,
                                                    'livros_emprestar':livros_emprestar,
                                                    'livros_emprestados':livros_emprestados,})
        else:
            return HttpResponseRedirect('Esse livro não é seu')
    return redirect('/auth/login/?status=2')
```

Figura 5.4: Código Visualizar dados Livro

- Cadastrar Livro (Figura 5.5) gera o form a partir da Classe Livro utilizando o django e valida se as informações são validas de acordo com as especificações da Classe.

```
def cadastrar_livro(request):
    if request.method == 'POST':
        form = CadastroLivro(request.POST, request.FILES)

        if form.is_valid():
            form.save()
            return redirect('/livro/home')
        else:
            return HttpResponseRedirect('DADOS INVALIDOS')
```

Figura 5.5: Código Cadastrar Livro

- Cadastrar Categoria (Figura 5.6) diferente do cadastro de livro que foi utilizado o django para gerar um form, aqui o form foi criado passo a passo e recebe o id do usuário que está logado, após receber as informações ele faz uma autenticação com o id do usuário para garantir que

uma nova categoria não seja criada em um usuário diferente da que está logado no sistema. Caso esteja tudo correto ele salva a nova categoria no Banco de dados.

```
def cadastrar_categoria(request):
    form = CategoriaLivro(request.POST)
    nome = form.data['nome']
    descricao = form.data['descricao']
    id_usuario = request.POST.get('usuario')
    if int(id_usuario) == int(request.session.get('usuario')):
        categoria = Categoria(nome=nome, descricao=descricao, usuario_id=id_usuario)
        categoria.save()
        return redirect ('/livro/home?cadastro_categoria=1')
    else:
        return HttpResponse('Erro')
```

Figura 5.6: Código Cadastrar Categoria

- Fazer Empréstimo (Figura 5.7) verifica se o usuário fará o empréstimo para outro usuário já cadastrado no sistema ou para um usuário 'anonimo' que seria uma pessoa que ainda não fez o cadastro, verifica o estatus do livro então caso um livro já esteja emprestado ele não aparecerá como opção para empréstimo, quando o empréstimo é feito ele atualiza a tabela de empréstimo e também o estatus do livro e redireciona a página para a home.

```
def cadastrar_emprestimo(request):
    if request.method == 'POST':
        nome_emprestado = request.POST.get('nome_emprestado')
        nome_emprestado_anonimo = request.POST.get('nome_emprestado_anonimo')
        livro_emprestado = request.POST.get('livro_emprestado')

        if nome_emprestado_anonimo:
            emprestimo = Emprestimos(nome_emprestado_anonimo = nome_emprestado_anonimo,
                                      livro_id = livro_emprestado)
        else:
            emprestimo = Emprestimos(nome_emprestado_id = nome_emprestado,
                                      livro_id = livro_emprestado)

        emprestimo.save()

        livro = Livros.objects.get(id=livro_emprestado)
        livro.emprestado = True
        livro.save()

    return redirect('/livro/home')
```

Figura 5.7: Código Empréstimo

- Fazer Devolução (Figura 5.8) verifica o id do livro que será devolvido e atualiza o estatus do livro, também altera a tabela de empréstimos para adicionar a data de devolução.

```
def devolver_livro(request):
    id = request.POST.get('id_livro_devolver')
    livro_devolver = Livros.objects.get(id = id)
    livro_devolver.emprestado = False
    livro_devolver.save()

    emprestimo_devolver = Emprestimos.objects.get(Q(livro = livro_devolver) & Q(data_devolucao = None))
    emprestimo_devolver.data_devolucao = datetime.now()
    emprestimo_devolver.save()

    return redirect('/livro/home')
```

Figura 5.8: Código Devolução

5.2 Documentação do Software

Passos para rodar o sistema:

1. Baixar todo código fonte disponibilizado;
2. Baixar e instalar o Python - <https://www.python.org/>
3. Abrir projeto em um editor de texto;
4. Baixar as dependencias do django e pillow no terminal com o comando **pip install django pillow**;
5. Iniciar servidor com o comando **python manage.py runserver**;
6. Abrir endereço gerado e completar a url com **/auth/cadastro**;
7. **exemplo http://127.0.0.1:8000/auth/cadastro**;
8. Fazer seu cadastro e utilizar o sistema.



6. Considerações Finais

Com o intuito de desenvolver um novo sistema para o gerenciamento de um biblioteca, sendo inspirado em sistemas já existentes para causar uma melhor transição e adaptação porém contendo todas as melhorias necessárias e atendendo seus requisitos, foram realizadas uma série de atividades, divididas em três etapas principais : planejamento, análise e projeto. Dentre as atividades realizadas estão o levantamento de requisitos, estudo de casos de uso para o sistema, análise dos custos e benefícios do desenvolvimento deste novo sistema, o desenvolvimento de Diagramas contendo os processos principais do sistema e representação do de forma "artística" a arquitetura do sistema. Tambem foi desenvolvido algumas funcionalidades do sistema possibilitando seu funcionamento, ainda é necessário adicionar novas funcionalidades para que o sistema possa ser considerado como completo, principalmente na parte estética da interface. No geral o sistema está funcional e atingiu requisitos impostos no inicio do semestre.





Referências Bibliográficas

- [DWR14] Alan Dennis, Barbara Haley Wixom, and Roberta M. Roth. *Análise e Projeto de Sistemas*. LTC, Rio de Janeiro, 5 edition, 2014. Citado na página 1.
- [Fur13] Sérgio Furgeri. *Modelagem de Sistemas Orientados a Objetos*. Érica Editora, São Paulo, SP, 1 edition, 2013. Citado na página 1.
- [Gue11] Gilleanes T.A. Guedes. *UML 2 : uma abordagem prática*. Novatec Editora, 2011. Citado na página 1.
- [Hel13] Helio Engholm Jr. *Análise e Design Orientados a Objetos*. Novatec, 2013. Citado na página 1.
- [SJB12] John W. Satzinger, Robert B. Jackson, and Stephen D. Burd. *Introduction to Systems Analysis and Design: An Agile, Iterative Approach*. Course Technology, CENGAGE Learning, Mason, Ohio, 6 edition, 2012. Citado 2 vezes nas páginas 1 e 29.
- [Som18] Ian Sommerville. *Engenharia de Software*. Pearson Education do Brasil, São Paulo, 10 edition, 2018. Citado na página 1.
- [SR12] Gary B. Shelly and Harry J. Rosenblat. *Analysis and Design for Systems*. Course Technology, CENGAGE Learning, 9 edition, 2012. Citado na página 1.
- [Waz11] Raul Sidnei Wazlawick. *Análise e Projeto de Sistemas de Informação Orientados a Objetos*. Editora Campus SBC. Elsevier, Rio de Janeiro, RJ, 2 edition, 2011. Citado 2 vezes nas páginas 1 e 5.