

Desenvolvimento de um Sistema Web para a Informatização da ONG Amor-Exigente na Região de Santa Maria

Valentim Bandeira Verardo¹, Gustavo Stangherlin Cantarelli¹

¹Curso de Sistemas de Informação – Universidade Franciscana
CEP 97010-491 – Santa Maria – RS – Brasil

{vb.verardo, gus.cant}@gmail.com

Abstract. *This work presents a solution, through the development of a web system, to facilitate and assist the creation of the structure of the meetings given by the non-governmental organization Amor-Exigente and initially focusing on the children's meetings. In order to achieve the development of a robust and simple system, the Feature Driven Development methodology was chosen to determine the software construction processes, using the PHP programming language, Laravel framework, on the fullstack it will be the Livewire framework and as a management system for database was chosen MySQL.*

Resumo. *Este trabalho apresenta uma solução, por meio do desenvolvimento de um sistema web, para facilitar e auxiliar na criação da estrutura dos encontros ministrados pela organização não governamental Amor-Exigente e tem, inicialmente, foco nos encontros das crianças. Para que se consiga o desenvolvimento de um sistema robusto e simples, foi escolhida a metodologia Feature Driven Development para determinação dos processos de construção do software, fazendo uso da linguagem de programação PHP, framework para a estruturação do back ending Laravel, no fullstack será o framework Livewire e como sistema gerenciador de banco de dados foi escolhido o MySQL.*

1. Introdução

Todo e qualquer processo de informatização traz profundas transformações nos locais onde são aplicados e na vida de todas as pessoas envolvidas, resultando em benefícios para eles. Dentre os benefícios reais, podem ser destacados principalmente: a economia de tempo, o aumento da produtividade e agilidade nas tarefas do usuário.

Atualmente, uma Organização Não Governamental (ONG) se mantém por meio de auxílios, tanto dos próprios voluntários que nela já atuam quanto de diversas pessoas que apenas querem ajudar.

Na ONG Amor-Exigente tem-se o foco na prevenção, orientação e no apoio às famílias, dependentes químicos ou usuários de drogas. É um programa que faz refletir sobre o comportamento pessoal, da família e da sociedade, bem como auxilia na retomada dos valores para uma melhor qualidade de vida.

Além de ser um forte aliado na prevenção ao uso abusivo de drogas (lícitas e ilícitas) e no tratamento da codependência (por parte dos familiares dos dependentes químicos), é um programa destinado à comunidade em geral. Inclui, também, aqueles que

não convivem com o problema, visto que ensina maneiras de estabelecer um melhor convívio familiar e social através da sua filosofia e práticas de reflexão (autoconhecimento), orientação nas tomadas de atitude, caridade, percepção da realidade e enfrentamento dos problemas [Amor-Exigente 2020].

1.1. Justificativa

Após iniciar a participação em trabalho voluntário com crianças, dentro da ONG Amor-Exigente, foi identificada a necessidade da atualização da maneira pela qual o processo dos encontros é feito entre as equipes voluntárias e as famílias ou crianças.

Na atualidade, o roteiro seguido dentro de um encontro é feito via planejamentos, onde ele é escrito no programa Microsoft Word e armazenado na nuvem; consequentemente, é necessária uma série de repetições para que os novos planejamentos sejam escritos.

Juntamente com isso, o armazenamento de informações, a geração das atividades e o simples controle das crianças e voluntários é feito por papel ou de forma repetitiva e desgastante em planilhas do Microsoft Excel.

Tendo esses fatos em mente, o projeto aqui apresentado foi a informatização do processo como um todo, desenvolvendo um sistema web, inicialmente com foco nos encontros e com a prioridade de facilitar e agilizar a criação das atividades que dão suporte aos encontros.

1.2. Objetivo geral

Este trabalho teve por objetivo o desenvolvimento de um sistema de gerenciamento em plataforma web, inicialmente com porte regional, para controle e criação dos planejamentos, acompanhamento dos encontros, cadastros das pessoas (voluntárias e/ou famílias) e extração de dados para prevenções e até mesmo identificação de padrões de altos e baixos no comportamento identificado ao decorrer dos encontros semanais.

Por fim, o sistema conta com *dashboards*, que são interfaces de apresentação de informações, juntamente de interfaces de gerenciamento que vão focar nos seguintes tópicos: atividades, planejamentos, encontros e crianças, juntamente com as famílias e voluntários, dentre outras informações necessárias para a realização dos encontros com todo o ecossistema do Amor-Exigente.

1.3. Objetivos específicos

Para este projeto, os objetivos específicos encontrados são:

- Aplicar de forma prática as diretrizes de IHC (Interface Humano Computador) para que se tenha um sistema intuitivo;
- Ter um contínuo feedback após início do desenvolvimento, para que o quanto antes seja implantado e testado;
- Aprofundar o aprendizado em UX (*User Experience*) para desenvolver um software intuitivo, simples e prático;
- Proporcionar armazenamento e fácil acesso das informações pertinentes aos encontros, sendo as principais: equipes de voluntários, planejamentos e atividades já realizadas.

2. Referencial Teórico

Nesta seção, serão abordados todos os conceitos e tópicos necessários para o desenvolvimento deste software. Na sequência, serão apresentados conceitos sobre ONGS e linguagem de programação PHP, juntamente com o *framework* Laravel, a linguagem de marcação HTML em conjunto com *framework* Livewire e estilização CSS com seu *framework* Bootstrap e o gerenciador de banco de dados MySQL.

2.1. ONG - Organização não Governamental

ONG, ou Organização Não-Governamental, são todas as organizações de terceiro setor, ou seja, sem fins lucrativos, criadas por pessoas que trabalham voluntariamente em defesa de uma causa, seja ela proteção do meio ambiente, defesa dos direitos humanos, erradicação do trabalho infantil, entre outras [Estraviz 2018].

A sigla foi instituída em 1950 pela Organização das Nações Unidas (ONU) para fazer referência às organizações civis que não tinham nenhum vínculo com o governo.

A ONG é mantida financeiramente por pessoas físicas, empresas privadas, fundações e, em alguns casos, com a colaboração do próprio Estado. Os trabalhos são desenvolvidos por funcionários contratados e, principalmente, por voluntários [Estraviz 2018].

2.2. Amor-Exigente

Desde 1984, a ONG Amor-Exigente (AE) atua como apoio e orientação aos familiares de dependentes químicos e as pessoas com comportamentos inadequados. Através de um eficiente programa de auto e mútua ajuda, o Amor-Exigente desenvolve preceitos para a reorganização familiar, sensibilizando as pessoas e levando-as a perceber a necessidade de mudar o rumo de suas vidas a partir de si mesmas, proporcionando equilíbrio e melhor qualidade de vida [Amor-Exigente 2020].

Este programa, que há mais de 3 décadas funciona e dá certo, é praticado por meio de 12 Princípios Básicos, 12 Princípios Éticos, Espiritualidade Pluralista e Responsabilidade Social, através de reuniões semanais, cursos e palestras, sempre com a dedicação e comprometimento dos milhares de voluntários espalhados por todo o Brasil, Argentina e Uruguai [Amor-Exigente 2020].

Visando maior abrangência do seu movimento de proteção social, o Amor-Exigente expandiu seu programa, dando origem aos projetos especiais: Prevenção - visa desestimular a experimentação e uso de tabaco, álcool e outras drogas através de uma abordagem educativa junto a pais, avós e professores para ajudar na formação de seus filhos, netos e alunos; Sobriedade - esse trabalho, com abordagem distinta, destina-se especialmente às pessoas em processo de recuperação pelo uso e abuso de álcool e outras drogas; Amor-Exigentinho - proposta voltada para o público infanto-juvenil que utiliza os Princípios Básicos do Programa de forma lúdica, clara e objetiva, visando auxiliá-lo a adotar atitudes responsáveis e saudáveis; Sempre É Tempo - focado nos adultos da Melhor Idade que muitas vezes passam pela “síndrome do ninho vazio” ou com o desafio de cuidar dos netos enquanto os pais trabalham e/ou estudam [Amor-Exigente 2020].

2.3. PHP

PHP (*Hypertext Preprocessor*) é uma linguagem de script *open source* muito utilizada e bem adequada e estável para o desenvolvimento web, pois possui um fácil entrosamento com o HTML [PHP 2020].

É uma linguagem muito beneficiada por ser de simples e fácil aprendizado e disponibiliza para o desenvolvedor, iniciante ou profissional uma série de recursos muito úteis, não esquecendo também de sua vasta documentação e comunidade muito ativa.

2.4. Laravel

O Laravel foi criado em meados de 2011 inicialmente para ser uma alternativa ao framework do momento, CodeIgniter, que não dava suporte para autenticação nem autorização de usuários.

Sua primeira versão impactante foi a 4.0, onde, no momento, implementaram a utilização de *composer*, um gerenciador de pacotes do próprio PHP e reestruturaram todo o *framework* para que o mesmo fosse dividido dentro do *composer*. Consequentemente, com isso, diminuiria de “tamanho” inicial, visto que, a partir daquele momento, era possível utilizar e importar apenas o que se fazia necessário para o projeto.

O *framework* conta com o Eloquent ORM (*Object Relational Mapping*), o qual é uma estrutura que faz a relação com o banco de dados. Ele utiliza uma model para representação de uma dada tabela, sendo assim, como toda e qualquer classe do PHP, ele conta com diversos métodos para facilitar qualquer *query* executada no banco de dados.

É um *framework* PHP com sintaxe elegante e expressiva, como os desenvolvedores definem. Ele usa arquitetura MVC, deixando o código limpo, e ajuda a desenvolver aplicações seguras e robustas de forma ágil e eficiente. O Laravel conta com uma documentação vasta, o que torna o uso fácil e o conteúdo acessível [Alencar 2016].

É um ótimo *framework* e será de grande valia no projeto, visto que o mesmo proporciona um rápido desenvolvimento e ótimos retornos, tanto visuais quanto em desempenhos.

2.5 Padrão MVC

O padrão MVC foi desenvolvido em 1979 por Trygve Reenskaug com a finalidade de ser utilizado como arquitetura para aplicativos *desktop*. Entretanto, o padrão se popularizou para uso em sistemas web, a partir da adesão de milhares de Frameworks de mercado.

É um padrão de arquitetura de aplicações que divide a aplicação em três camadas: a visão (*view*), que é responsável apresentação das informações ao usuário e é o ponto de entrada de informações para serem inseridas na aplicação, o modelo (*model*), onde fica a lógica de negócios da aplicação e o controlador (*controller*), que trabalha a interação entre a camada de negócios e processamento das informações para a camada de apresentação [Massari 2017].

2.6. Livewire

A ferramenta Livewire é um *framework full-stack* utilizado tanto para auxiliar a construção de interfaces no *front-end* quanto para auxílio na construção do *back-end*.

Por ser uma ferramenta baseada em componentes o mesmo facilita bastante a estrutura e a troca de informação existente dentre as camadas view e controller.

O Livewire tem seu foco na camada view e controller do modelo MVC (Model-View-Controller), onde view é o padrão de arquitetura que separa a representação da informação do usuário [Livewire 2020].

Uma das grandes vantagens desse *framework* é a velocidade, juntamente com a escalabilidade e a simplicidade.

2.7. Bootstrap

O Bootstrap é uma ferramenta gratuita para desenvolvimento HTML (*Hyper Text Markup Language*), CSS (*Cascading Style Sheets*) e JavaScript desenvolvida pelo e para o Twitter [Logen 2020a].

Antes de ser *open source*, ou seja, uma estrutura de código aberto, o Bootstrap era conhecido como “TwitterBlueprint”, mas após alguns meses de desenvolvimento do framework, o projeto desandou e cresceu exponencialmente em um evento realizado pela empresa desenvolvedora à medida em que os desenvolvedores de diversos níveis de conhecimento evoluíram sem nenhuma orientação externa [Logen 2020b].

O HTML não é uma linguagem de programação, isso significa que não pode ser usado para criar funcionalidades dinâmicas. Entretanto, o HTML possibilita a organização e formatação de documentos, similar ao Microsoft Word. Já o CSS é um mecanismo para adicionar estilo (cores, fontes, espaçamentos, entre outros) a um documento web [Okubo 2018].

Para que não seja necessária a inclusão desses estilos dentro do documento, o CSS cria um link para uma página onde só contenham os códigos para estilização desse documento [W3C 2020].

No projeto proposto, a utilização do *framework* Bootstrap é necessária devido ao tempo ganho no desenvolvimento de interfaces, visto que ele já possui componentes prontos disponíveis para uso no site oficial.

O Bootstrap será utilizado dentro do Paper Dashboard, um *template* grátis de *Dashboards* Administrativas desenvolvido com VueJS que consiste em design e funcionalidades básicas já implementadas na questão do *front-end*.

2.8. MySQL

O MySQL é o sistema gerenciador de banco de dados (SGBD) de código aberto mais popular. Apesar de contar com diversos recursos, possui uma configuração facilitada e simples fazendo uso da Linguagem de Consulta Estruturada (SQL) [MySQL 2020].

Este SGBD foi selecionado para a consistência dos dados deste projeto visto sua facilidade de aprendizado, manutenção e atualizações constantes.

3. Trabalhos Relacionados

Os três trabalhos apresentados na sequência tratam o aspecto de desenvolvimento de um sistema de gerenciamento ou administração dentro da área de ONGS ou que tenham como base tecnologias como a linguagem PHP, HTML e JavaScript.

3.1. Desenvolvimento de uma ferramenta de gestão de relacionamento com o cliente para o pós-venda

O projeto realizado por Ambrosi (2018), pela Universidade Federal de Santa Catarina, teve como base o problema das empresas atualmente em segurar e conseguir acompanhar seus clientes dentro das suas plataformas e tem como objetivo o desenvolvimento de uma MPV - um Mínimo Produto Viável - para a solução do problema de pós-venda, ou seja, manter o cliente e acompanhar o mesmo.

O trabalho foi desenvolvido em PHP com o *framework* Laravel na parte do *back-ending*, já no *front* foi utilizado JavaScript com o *framework* VueJS e, como sistema de gerenciamento de banco de dados, foi escolhido o MySQL.

Por fim, o sistema desenvolvido obteve o resultado esperado, após utilização do MPV para controle e acompanhamento do cliente após efetuar a venda.

3.2. Desenvolvimento de um sistema web para controle de faltas da Universidade Federal de Ouro Preto

O trabalho realizado pela estudante Almeida (2018), pela Universidade Federal de Ouro Preto, teve como objetivo o fácil acesso de qualquer estudante a suas faltas em toda e qualquer matéria que ele esteja matriculado ou vinculado.

O trabalho conta com o desenvolvimento na linguagem PHP juntamente com o *framework* Laravel e, como sistema de gerenciamento de banco de dados é utilizado o MySQL.

Finalmente, o sistema desenvolvido teve a apresentação, a validação com testes e a devida capacidade de desempenhar a função que deve e substituir por inteira as metodologias e tecnologias utilizadas atualmente na instituição.

3.3. Proposta de desenvolvimento de um software web para vendas de kits gastronômicos

O trabalho apresentado por Dutra e Cantarelli (2018) tem como foco principal o desenvolvimento de uma plataforma web que proporciona o fácil acesso a receitas e insumos para fazer pratos culinários.

A aplicação conta com a utilização de tecnologias como PHP, HTML, Bootstrap, JavaScript e, como sistema de gerenciamento de banco de dados, foi escolhido MySQL.

3.4. Conclusão sobre os trabalhos relacionados

Os trabalhos relacionados acima possuem as tecnologias ou a finalidade e objetivo muito semelhante ao projeto, ambos os três são focados no desenvolvimento de sistemas e/ou ferramentas on-line para auxiliar as pessoas.

O trabalho desenvolvido por Ambrosi (2018) tem ponto de conexão com as tecnologias utilizadas, visto que ambos vão ser uma ferramenta on-line, MPV e irão utilizar o *framework* Laravel para o desenvolvimento da aplicação.

Já para Almeida (2018), que propôs e desenvolveu um sistema para controle de faltas dentro da universidade, também se tem a relação do *framework* Laravel juntamente com o sistema de gerenciamento de banco de dados MySQL e a finalidade, visto que era auxiliar todos os alunos a ficarem cientes das suas situações dentro das matérias.

Dutra e Cantarelli (2018) apresentaram um projeto de desenvolvimento web de uma plataforma com a finalidade de facilitar e auxiliar as pessoas a terem acesso a kits culinários. O projeto relaciona-se com este devido à linguagem de programação, onde foi escolhido PHP, o sistema de gerenciamento de banco de dados MySQL e a facilidade de acesso.

Os três trabalhos são de grande valia para a ideia do atual projeto, que visa o auxílio de várias pessoas, juntamente com tecnologias de fácil desenvolvimento e aprendizado, resultando em um sistema simples, bem apresentado e de fácil utilização.

4. Metodologia

Entre as metodologias ágeis que existiam antes do manifesto ágil, o FDD “*Feature Driven Development*” é uma delas.

Desenvolvido por Peter Coad e Jeff De Luca, foi inicialmente publicado em 1999 no livro “Java Modeling in Color with UML”, sendo que uma das principais obras, que possui uma versão completa da metodologia, foi publicada em 2002 após sua utilização em uma agência do United Overseas Bank em Singapura [Reis 2016].

O FDD incorpora muitas das boas práticas de desenvolvimento já reconhecidas pela indústria em um conjunto coeso. Essas práticas são orientadas a funcionalidades, que é um conceito de valor do ponto de vista do cliente. O principal objetivo do FDD é entregar uma peça de software tangível e funcional para o cliente em espaços de tempo regulares [Reis 2016].

O FDD busca o desenvolvimento por funcionalidade, ou seja, por um requisito funcional do sistema. É prático para o trabalho com projetos iniciais ou projetos com codificações existentes e conta com cinco processos:

- Desenvolver um modelo abrangente: Consiste na definição dos requisitos do domínio do sistema para sua construção e documentação da especificação de suas funcionalidades;
- Construir uma lista de funcionalidades: Momento onde cada funcionalidade é revisada pelo cliente (Apêndice A);
- Planejar por funcionalidades: Consiste em criar uma ordem sequencial onde as funcionalidades são executadas de acordo com sua prioridade (Apêndice B);
- Projetar por funcionalidades: etapa que visa possibilitar o entendimento do fluxo de execução do sistema (Apêndice C), suas atividades e respectivos tratamentos;
- Construir por funcionalidades: Consiste na implementação das funcionalidades requeridas pelo sistema, obedecendo, também, uma ordem prioridade.

4.1. Desenvolver modelo abrangente

Este é basicamente o processo inicial que abrange todo o projeto, devendo ser realizado por membros do domínio do negócio juntamente com os desenvolvedores. Realiza-se, então, um estudo dirigido sobre o escopo do sistema e seu contexto. Após, são realizados estudos com melhor detalhamento sobre o domínio do negócio para cada área a ser modelada [Reis 2016].

Na Figura 1 é apresentado o diagrama de domínio inicial do projeto, podendo-se verificar suas entidades e dependências.

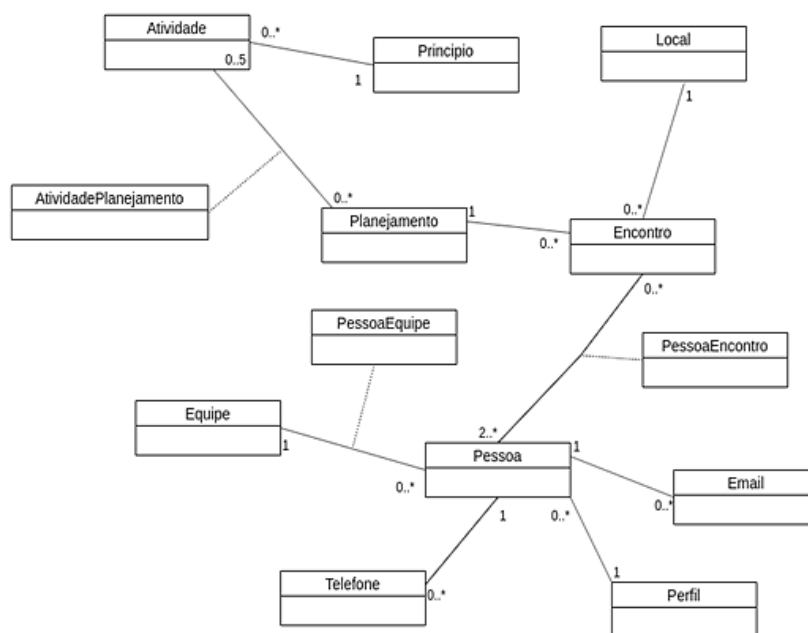


Figura 1. Diagrama de Domínio do Projeto

4.2. Projetar por funcionalidade

Esta etapa da metodologia tem como finalidade possibilitar o entendimento do fluxo de execução do sistema, quais as atividades que podem ser realizadas e como funciona seu tratamento dentro do mesmo [Reis 2016].

O Diagrama de Classes, apresentado na Figura 2, representa as classes a serem implementadas no sistema, ilustrando objetos juntamente com suas propriedades (atributos e métodos). Visto que todas as entidades servem de contexto de utilização para os encontros em si, ou seja, as atividades podem conter livros e estão vinculadas a princípios, planejamentos utilizam-se de atividades para montar o andamento dos encontros, e os encontros em si utilizam-se de pessoas, local e um planejamento, consequentemente tem-se o documento do encontro por completo contendo o local que será realizado, pessoas vinculadas (sendo elas crianças / participantes ou voluntários), data de realização e as atividades planejadas (com suas durações).

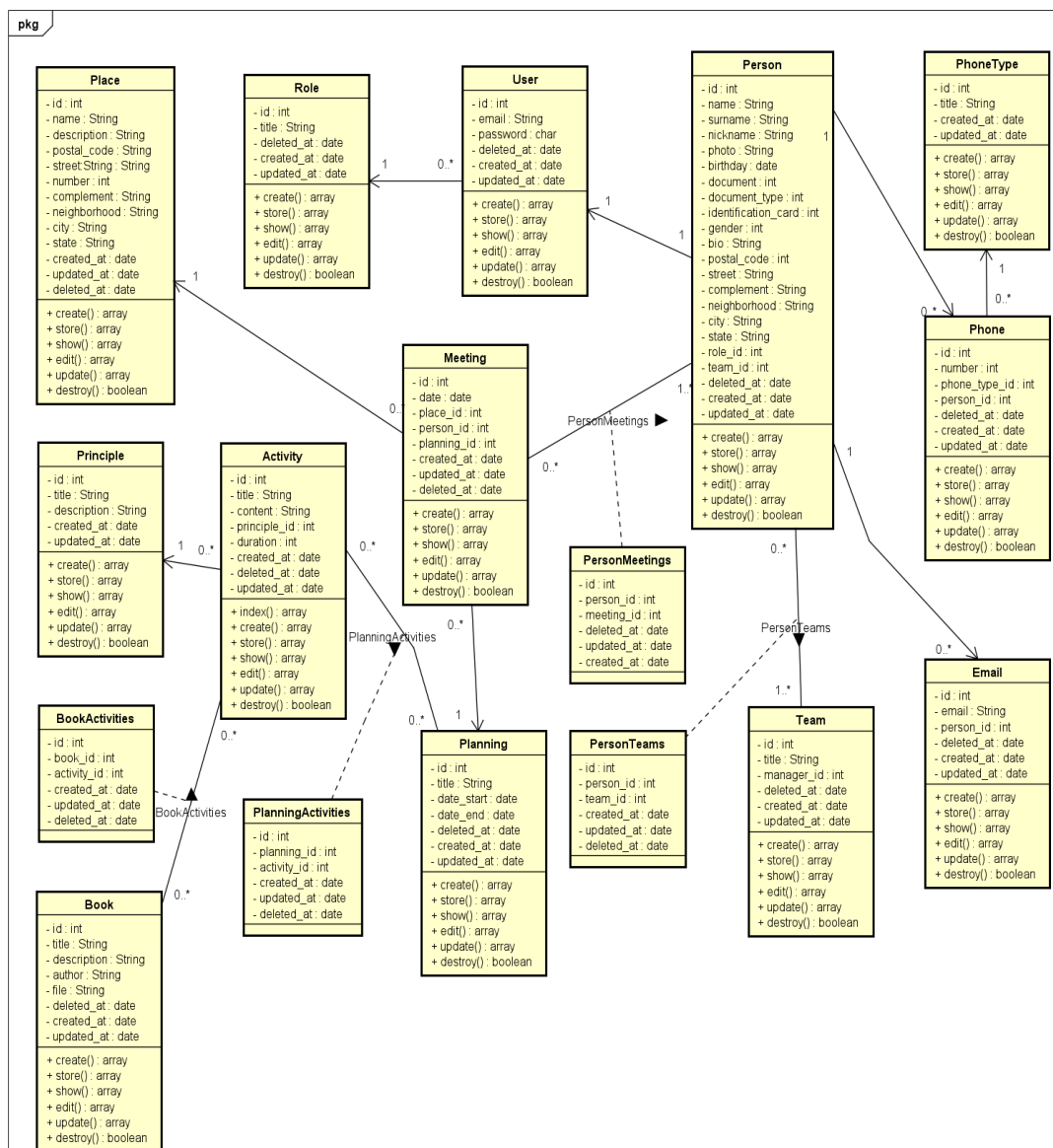


Figura 2. Diagrama de Classes

4.3. Construir por funcionalidade

Trata-se de uma fase da metodologia FDD que é executada uma única vez para cada funcionalidades, tendo como objetivo, produzir um Pacote de Arquitetura por funcionalidade. Abaixo vão ser apresentadas as três entidades principais do sistema, encontros, planejamentos e atividades.

No processo de desenvolvimento do sistema um dos pontos mais importantes é a estrutura de gerenciamento dos encontros onde todo e qualquer usuário, independentemente do perfil, poderá cadastrar, editar, excluir, visualizar e até mesmo gerar um pdf para impressão. A entidade conta com dois voluntários responsáveis, nenhum ou mais voluntários que vão auxiliar, nenhum ou mais participantes, sendo toda e qualquer pessoa cadastrada e que não possua um usuário ativo no sistema, um local para a realização e um planejamento previamente cadastrado. A Figura 3 apresenta a função

store que é responsável pelo cadastro do encontro e vínculo das dos participantes e voluntários ao mesmo.

```
public function store()
{
    $this->validate(
        [
            'date' => 'required',
            'place_id' => 'required',
            'first_responsible_id' => 'required|different:second_responsible_id',
            'second_responsible_id' => 'required|different:first_responsible_id',
            'planning_id' => 'required',
        ],
        [
            'date.required' => 'Necessário inserir a data',
            'place_id.required' => 'Necessário inserir o local',
            'first_responsible_id.required' => 'Necessário selecionar o primeiro responsável',
            'first_responsible_id.different' => 'Necessário que sejam diferentes',
            'second_responsible_id.required' => 'Necessário selecionar o segundo responsável',
            'second_responsible_id.different' => 'Necessário que sejam diferentes',
            'planning_id.required' => 'Necessário inserir o planejamento',
        ]
    );

    $data = request()->request->get('data');

    $meeting = new Meeting();

    if ($data['date']) {
        $data['date'] = Carbon::createFromFormat('d/m/Y', $data['date']);
    }

    $meeting->fill($data)->save();

    if ($this->participants) {
        $meeting->participants()->sync($this->participants);
    }

    if ($this->volunteers) {
        $meeting->volunteers()->sync($this->volunteers);
    }

    session()->flash('message', 'Cadastrado com sucesso!');
    session()->flash('type', 'success');

    $this->toIndex();
}
```

Figura 3. Função “store” implementada no componente de cadastro de encontros.

Também é importante a entidade de planejamentos, a estrutura de gerenciamento assim como encontros pode ser acessada por todo e qualquer usuário. Um planejamento conta com um título, possui vínculo a um princípio e relação com uma ou mais atividades de cada tipo. A Figura 4 apresenta a função *store* que é responsável pelo cadastro do planejamento e vínculo das atividades.

```

public function store()
{
    $data = $this->validate(
        [
            [
                'title' => 'required',
                'principle_id' => 'required',
            ],
            [
                'title.required' => 'Necessário inserir o título',
                'principle_id.required' => 'Necessário selecionar o princípio',
            ]
        ]
    );

    $planning = new Planning();
    $planning->fill($data)->save();

    if ($this->planningActivities) {
        $planning->activities()->sync($this->planningActivities);
    }

    session()->flash('message', 'Cadastrado com sucesso!');
    session()->flash('type', 'success');

    $this->toIndex();
}

```

Figura 4. Função “store” implementada no componente de cadastro de planejamentos.

Não menos importante o gerenciamento de atividades possui vínculo com um título, conteúdo onde é armazenada a atividade em si, vínculo a um princípio, vínculo com um tipo de atividade (dinâmica, espiritualidade, leitura, confraternização ou capacitação), uma duração e relação com nenhum ou vários livros. A Figura 5 apresenta a função *store* que é responsável pelo cadastro da atividade e vínculo dos seus livros.

```

public function store()
{
    $data = $this->validate(
        [
            [
                'title' => 'required',
                'content' => 'required',
                'principle_id' => 'required',
                'activity_type_id' => 'required',
                'duration' => 'required',
            ],
            [
                'title.required' => 'Necessário inserir o nome',
                'content.required' => 'Necessário inserir a atividade',
                'principle_id.required' => 'Necessário selecionar o princípio',
                'activity_type_id.required' => 'Necessário selecionar o tipo da atividade',
                'duration.required' => 'Necessário inserir a duração da atividade',
            ]
        ]
    );

    $activity = new Activity();
    $activity->fill($data)->save();

    if ($this->activityBooks) {
        $activity->books()->sync($this->activityBooks);
    }

    session()->flash('message', 'Cadastrado com sucesso!');
    session()->flash('type', 'success');

    $this->toIndex();
}

```

Figura 5. Função “store” implementada no componente de cadastro de atividades.

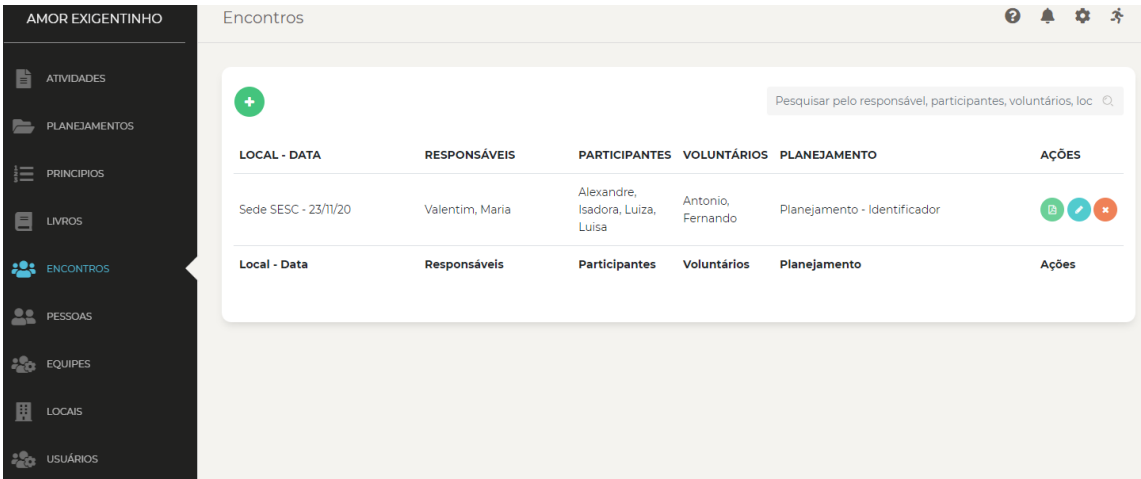
5. Resultados

Como resultado do presente trabalho tem-se a construção de um sistema web simples, intuitivo e eficaz. O mesmo conta com cerca de nove entidades onde cada uma possui seu próprio gerenciamento que conta com as ações de criação, edição, exclusão, juntamente da listagem reativa com campo de pesquisa dinâmico referente a informações pertinentes da entidade.

Todas as interfaces de listagem seguem o mesmo padrão, onde contam com uma tabela para apresentação dos registros da entidade, seja ela encontro, atividade, planejamentos, dentre os demais, botões de ações, tais como edição e exclusão, um campo para filtro rápido se baseando em todas as colunas apresentadas e um botão para cadastro.

Devido à grande quantidade de interfaces desenvolvidas, vão ser utilizadas como base para a apresentação das demais a listagem e o cadastro de encontros nas figuras 6 e 7 respectivamente.

Na Figura 6 a interface de listagem de encontros conta com uma tabela possuindo informações referentes a data, responsáveis, participantes, voluntários, planejamento e as ações que contam com a visualização e geração do pdf, edição e exclusão. Nesta figura temos a aplicação direta da questão de *user experience (UX)* onde mediante cores ligadas a intuição temos a inserção e visualização como sendo verde, sinalizando algo positivo, a exclusão como vermelho, identificando algo cuidadoso e a edição como neutro, a cor azul, e não menos importante, temos também a questão de uma busca reativa nas informações, onde conforme é digitada a pesquisa a tabela já é reajustada para apresentar os registros que tem relação com o que está sendo pesquisado.



The screenshot shows a web application interface for 'Encontros' (Meetings). On the left is a dark sidebar menu with options: ATIVIDADES, PLANEJAMENTOS, PRINCIPIOS, LIVROS, ENCONTROS (highlighted), PESSOAS, EQUIPES, LOCAIS, and USUÁRIOS. The main content area has a title 'Encontros' and a search bar with the placeholder text 'Pesquisar pelo responsável, participantes, voluntários, loc'. Below the search bar is a table with the following data:

LOCAL - DATA	RESPONSÁVEIS	PARTICIPANTES	VOLUNTÁRIOS	PLANEJAMENTO	AÇÕES
Sede SESC - 23/11/20	Valentim, Maria	Alexandre, Isadora, Luiza, Luisa	Antonio, Fernando	Planejamento - Identificador	[View] [Edit] [Delete]
Local - Data	Responsáveis	Participantes	Voluntários	Planejamento	Ações

Figura 6. Listagem de encontros.

Na Figura 7 é apresentada a interface de cadastro de encontros, ela conta com a seleção dos dois voluntários responsáveis pelo encontro, juntamente da data de realização, do local, do planejamento que já foi previamente cadastrado, dos participantes, sendo eles toda as pessoas que não possuem usuários no sistema e os voluntários os quais são todos os usuários do sistema.

Figura 7. Criação de um encontro.

Não menos importante, Figura 8, a interface de visualização e geração de pdf do encontro, onde conta com a disposição em ‘blocos’ de todas as informações pertinentes ao encontro. Informações como responsáveis, o local onde será realizado, tema e objetivos que são relacionados ao princípio são apresentados inicialmente, juntamente do momento de acolhimento.

Figura 8. Visualização e Geração do PDF da estrutura inicial.

Na sequência, na Figura 9 é apresentado o segundo momento do encontro, onde é utilizada as atividades do tipo ‘espiritualidade’ que estão vinculadas. Por ser uma atividade a mesma possui duração, tipo, e a dinâmica.

2º Momento: Espiritualidade

ATIVIDADE 1 - AO AMOR ANTIGO - DURAÇÃO 00:15:00

Efetuar a leitura e reflexão da poesia abaixo:

Ao Amor Antigo
O amor antigo vive de si mesmo,
não de cultivo alheio ou de presença.
Nada exige nem pede. Nada espera,
mas do destino vão nega a sentença.
O amor antigo tem raízes fundas,
feitas de sofrimento e de beleza.
Por aquelas mergulha no infinito,
e por estas suplanta a natureza.
Se em toda parte o tempo desmorona
aquilo que foi grande e deslumbrante,
a antigo amor, porém, nunca fenece
e a cada dia surge mais amante.
Mais ardente, mas pobre de esperança.
Mais triste? Não. Ele venceu a dor,
e resplandece no seu canto obscuro,
tanto mais velho quanto mais amor.

Figura 9. Segundo momento, atividades do tipo espiritualidade.

Na Figura 10 é apresentado o terceiro momento, onde começam todas as atividades de abordagem do princípio, essas atividades, juntamente com a espiritual preenchem a informação dos materiais, presente abaixo. Também na mesma interface é apresentada a avaliação, a oração e as observações que podem vir a ser importantes, o que marca o final do encontro.

3º Momento: Abordagem do princípio

Apresentação Das Atividades Do Planejamento Vinculado Ao Encontro

- **Introdução:** Perguntar aos participantes qual o princípio do mês e o que eles já guardaram até agora do que foi passado para eles, destacando também sobre o sigilo que se mantém sobre os encontros [Ético]

ATIVIDADE 1 - ATIVIDADE LEITURA DE LIVRO - DURAÇÃO 00:45:00

Juntamente com as crianças o voluntário fará a leitura de um dos livros adicionados nos materiais e irá instigar elas a relacionar o livro com o princípio apresentado.

Exemplos de perguntas para iniciar a conversa:

1. O que entenderam do livro?
2. Qual parte foi a mais marcante do livro?
3. Em que momento é relatado algo relacionado ao princípio?
4. Costaram da leitura?
5. Podemos relacionar o livro com a nossa vida?

Avaliação e Oração da Serenidade

Avaliação do trabalho realizado no dia, dúvidas, etc.

- O que mais gostou do encontro?
- O que menos gostou?
- Sugestões?

Materiais

Livros, Notebook, Folhas...

VAVÁ E POPO DESCUBREM AS FAMÍLIAS DAS EMOÇÕES, VAVÁ E POPO DESCUBREM AS FORÇAS PESSOAIS

Observação

Encontro que deverá ser iniciada a preparação para a apresentação do natal.

Figura 10. Terceiro momento, avaliação, materiais e observação.

Devido ao momento de pandemia o sistema foi testado diretamente pelos voluntários que vão utilizar. Os testes estão sendo feitos nos encontros virtuais que vem sendo realizados semanalmente e não foi aplicado na prática para com os encontros presenciais.

Durante os testes nas apresentações das funcionalidades semanalmente, ficou nítida a diferença do tempo de criação e escrita das atividades, planejamentos e

principalmente dos encontros ocorridos, visto que todas as informações ficam salvas no sistema e podem ser consultadas e reutilizadas a qualquer momento.

Optou-se juntamente aos voluntários por não contemplar inicialmente uma estrutura de relatórios e exportação, visto que no momento a necessidade não era esta, e sim focar nos pontos pertinentes e que já vinham sendo executados referentes as escritas e armazenamento dos documentos dos encontros. Conseqüentemente teve-se o foco nas entidades de pessoas e atividades, juntamente dos planejamentos e encontros, seguindo assim uma ordem de necessidade e pré-requisito.

Não menos importante este trabalho também contou com uma publicação no ano de 2020 no XXIV Simpósio de Ensino, Pesquisa e Extensão – SEPE.

6. Conclusão

Este trabalho possibilitou o desenvolvimento de um sistema web para organização e melhor gerenciamento dos encontros realizados entre os voluntários, famílias e crianças dentro da ONG Amor-Exigente, e tendo o foco inicial nos encontros com as crianças.

O sistema desenvolvido está apenas com a estrutura inicial, ou seja, o MPV (Mínimo Produto Viável) e possibilita expansão, conseqüentemente o desenvolvimento da plataforma terá continuidade para que assim chegue o mais próximo da realidade dos voluntários que vão utilizar e para que possa ser utilizado não só pela região de Santa Maria mas também pelas demais regionais e talvez servindo como base para a ONG.

Para isso destaca-se a importância de um *framework* robusto juntamente de um sistema de gerenciamento de banco de dados eficiente que comportem tamanha carga e quantidade de requisições e entregue ao usuário o que lhe é necessário em um tempo aceitavelmente rápido.

Identificou-se também a importância de começar os testes do sistema logo que o mesmo começou a ser desenvolvido, visto que apesar de não estar sendo testado diretamente em um encontro com as crianças e famílias já teve-se *feedbacks* necessários para alterações e otimizações do mesmo com a finalidade de ficar cada vez mais simples e próximo da real necessidade do usuário.

Os trabalhos correlatos foram de suma importância, visto que, com o estudo deles, foi possível identificar, principalmente, os requisitos e as tecnologias de fácil desenvolvimento e ótimo retorno, servindo, então, como base para o desenvolvimento do projeto aqui apresentado.

Referências

- Alencar, J. (2016) “API RESTful PHP com Laravel”, <http://clubedogeeks.com.br/programacao/php/api-restful-laravel>, Março.
- Almeida, B. C. (2018) “Desenvolvimento de um Sistema Web para Controle de Faltas da Universidade Federal de Ouro Preto”, https://www.monografias.ufop.br/bitstream/35400000/820/1/MONOGRAFIA_DedenvolvimentoSistemaWeb.pdf, Março.
- Ambrosi, M. A. (2018) “Desenvolvimento de uma ferramenta de gestão de relacionamento com o cliente para o pós-venda”, <https://repositorio.ufsc.br/handle/123456789/197750>, Março.

- Amor-Exigente. (2020) “Federação de Amor-Exigente – FEAE”, <https://amorexigente.org.br/>, Maio.
- Dev Media. (2020) “Introdução ao FDD - Feature Driven Development”, <https://www.devmedia.com.br/introducao-ao-fdd-feature-driven-development/27971>, Abril.
- Dutra, Á. e Cantarelli, G. S. (2018) “Proposta de desenvolvimento de um Software Web para vendas de Kits Gastronômicos”.
- Estraviz M. (2018) “100 melhores ONGS 2018”, https://www.institutodoar.org/wp-content/uploads/2019/01/Guia_100_melhores_ONGs_completo.pdf, Março.
- Livewire (2020) “What is Livewire”, <https://laravel-livewire.com>, Novembro.
- Logen, A. (2020a) “O Que é HTML? Guia Básico Para Iniciantes”, <https://www.hostinger.com.br/tutoriais/o-que-e-html-conceitos-basicos/>, Maio.
- Logen, A. (2020b) “O Que é Bootstrap? Guia para Iniciantes”, <https://www.hostinger.com.br/tutoriais/o-que-e-bootstrap/>, Maio.
- Massari, J (2017) “Padrão MVC | Arquitetura Model-View-Controller”, <https://www.portalgsti.com.br/2017/08/padrão-mvc-arquitetura-model-view-controller.html>, Março.
- MySQL. (2020) “MySQL Documentation”, <https://dev.mysql.com/doc/>, Março.
- Okubo, B. (2018) “O que é HTML e Para que Serve?”, <https://br.godaddy.com/blog/o-que-e-html-e-para-que-serve/>, Março.
- PHP. (2020) “Manual do PHP”, https://www.php.net/manual/pt_BR/intro-what-is.php, Março.
- Reis, L. (2016) “Metodologia ágil FDD”, <http://www.leandromtr.com/metodologia-agil-fdd/>, Março.
- W3C. (2020) “CSS Documentation”, <https://www.w3.org/Style/CSS/>, Março.

Apêndice A. Construir a lista de funcionalidades

Trata-se da segunda atividade da metodologia FDD que visa reconhecer as diversas funcionalidades que buscam satisfazer os requisitos do sistema. Esta fase possui uma abrangência sobre toda a concepção do projeto [Dev Media 2020].

A listagem abaixo é referente a todos os Requisitos Não Funcionais (RNF) do projeto:

- RNF1. Fazer uso da Linguagem de Programação PHP;
- RNF2. Fazer uso do *framework* Laravel;
- RNF3. Fazer uso do *framework* Livewire;
- RNF4. Fazer uso do *framework* Bootstrap;
- RNF5. Fazer uso do *template* Paper Dashboard;
- RNF6. Fazer uso do Sistema de Gerenciamento de Banco de Dados MySQL.

A Listagem abaixo apresenta todos os Requisitos Funcionais (RF) do projeto:

- RF1. Gerenciar Usuários:
 - RF1.1. Gerenciamento do Login;
 - RF1.2. Gerar relatório de Usuários;
 - RF1.3. Descrição: Módulo responsável pela inclusão, exclusão, edição e consulta de usuários (gerenciamento) e pela exportação da listagem juntamente com filtros;
- RF2. Gerenciar Pessoas:
 - RF2.1. Gerenciamento das Pessoas;
 - RF2.2. Exportar a listagem das pessoas juntamente com seus filtros;
 - RF2.3. Descrição: Gerenciamento das pessoas (usuários, crianças, adultos);
- RF3. Gerenciar Atividades:
 - RF3.1. Gerenciar Atividades;
 - RF3.2. Exportar a listagem das atividades juntamente com seus filtros;
 - RF3.3. Descrição: Gerenciamento das atividades que podem ser inseridas dentro dos encontros;
- RF4. Gerenciar Encontros:
 - RF4.1. Gerenciamento dos encontros;
 - RF4.2. Exportar a listagem dos encontros juntamente com seus filtros;
 - RF4.3. Análise e interpretação dos encontros pelas pessoas;
 - RF4.4. Descrição: Gerenciamento dos encontros, seleção das atividades atreladas ao princípio ao qual o encontro será vinculado;
- RF5. Gerenciar Livros:

- RF5.1. Gerenciamento dos Livros;
- RF5.2. Exportar a listagem dos livros juntamente com seus filtros;
- RF5.3. Descrição: Gerenciamento dos livros presentes para seleção dentro de cada atividade relacionada a seu princípio;
- RF6. Gerenciar Princípios:
 - RF6.1. Gerenciamento dos princípios;
 - RF6.2. Exportar a listagem dos princípios juntamente com seus filtros;
 - RF6.3. Descrição: Gerenciamento dos princípios (descrição e informações adicionais);
- RF7. Gerenciar Equipes:
 - RF7.1. Gerenciamento das equipes (voluntários ou responsáveis);
 - RF7.2. Exportar a listagem das equipes juntamente com seus filtros;
 - RF7.3. Descrição: Gerenciamento das equipes;
- RF8. Gerenciar Perfis:
 - RF8.1. Gerenciamento dos perfis de acesso;
 - RF8.2. Exportar a listagem dos perfis juntamente com seus filtros;
 - RF8.3. Descrição: Gerenciamento dos perfis de acessos ao sistema;
- RF9. Gerenciar Locais:
 - RF9.1. Gerenciamento dos locais;
 - RF9.2. Exportar a listagem dos locais juntamente com seus filtros;
 - RF9.3. Descrição: Gerenciamento dos locais onde serão realizados os encontros.

Apêndice B. Planejamento por funcionalidades

Na etapa de planejamento por funcionalidade, é definida a ordem da implementação dos requisitos funcionais, baseada na complexidade e dependências desses, estimando-se um tempo de desenvolvimento para cada funcionalidade do software [Reis 2016].

Mostra-se, através da listagem de requisitos funcionais apresentada anteriormente, a estimativa de tempo para desenvolvimento do projeto. Após, será seguida a ordem da listagem de requisitos funcionais para implementação das principais funcionalidades e, para o cálculo da estimativa de tempo, utilizou-se a métrica presente na Tabela 1.

Tabela 1. Planejamento por funcionalidade

Ordem	Funcionalidade	Tempo de desenvolvimento estimado
1	Gerenciar Usuários	12 horas
2	Gerenciar Pessoas	12 horas
3	Gerenciar Princípios	12 horas
4	Gerenciar Equipes	12 horas
5	Gerenciar Livros	12 horas
6	Gerenciar Locais	12 horas
7	Gerenciar Perfis	12 horas
8	Gerenciar Atividades	36 horas
9	Gerenciar Encontros	36 horas

Na Figura 11, representando o cenário de aplicação das funcionalidades do sistema, apresenta-se o desenvolvimento do Diagrama de Casos de Uso.

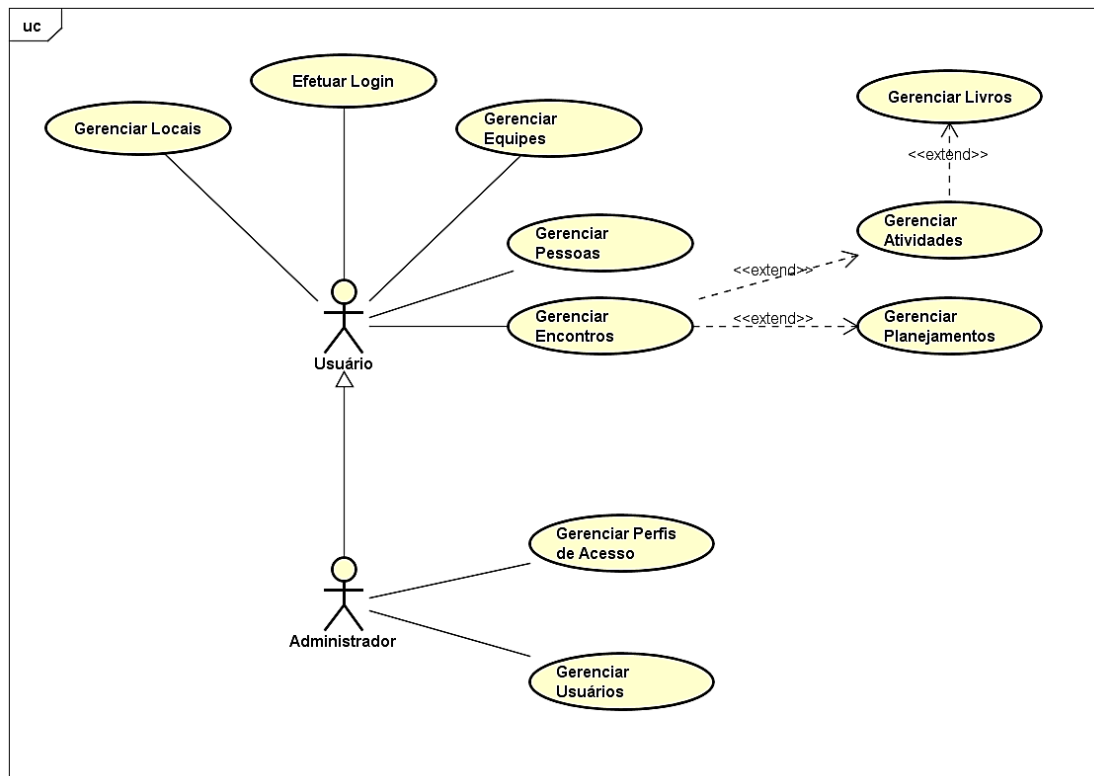


Figura 11. Diagrama de Casos de Uso

Na sequência, serão apresentados três Descritivos de Casos de Usos que são considerados os de maior importância para o desenvolvimento e utilização do software.

Na Tabela 2, é apresentado o Descritivo de Caso de Uso da funcionalidade de Gerenciar Atividades - criação, edição, exclusão e consulta - a qual pode ou não ser incluída dentro de um encontro.

Tabela 2. Descritivo de Caso de Uso - Gerenciar Atividades

Identificação	CDU01
Caso de Uso	Gerenciar Atividades
Atores Principais	Administrador, Usuário
Atores Secundários	N/A
Pré-Condição	O ator deve estar logado no sistema
Pós-Condição	N/A
Fluxo Principal	P01 - O Sistema apresenta a interface inicial do sistema; P02 - O Ator seleciona o menu Atividades; P03 - O Sistema apresenta a interface de listagem de atividades; P04 - O Ator clica no botão de cadastrar Atividade [A01], editar Atividade [A02], excluir Atividade [A03] ou filtrar e listar Atividades [A04]; P06 - O Ator preenche o campo Título e Conteúdo;

Identificação	CDU01
Fluxo Principal	P07 - O Ator seleciona o Princípio ao qual a Atividade ficará vinculada; P08 - O Ator clica no botão Cadastrar; P09 - O Sistema valida os dados [E01]; P10 - O Sistema grava os dados no Banco de Dados; P11 - Fim do caso de uso.
Fluxo Alternativo	[A01] - Filtrar e listar Atividades: A01.1 - O Sistema exibe a listagem das Atividades; A01.2 - O Ator preenche os dados para pesquisa; A01.3 - O Sistema exibe os dados pesquisados [E02]; A01.4 - O Sistema retorna para P03; [A02] - Editar Atividade: A02.1 - O Sistema exibe a listagem das Atividades; A02.2 - O Ator verifica e altera os dados necessários; A02.3 - O Ator clica no botão Salvar; A02.4 - O Sistema valida os dados [E01]; A02.5 - O Sistema grava os dados no Banco de Dados; A02.6 - O Sistema retorna para P03; [A03] - Excluir Atividade: A03.1 - O Sistema exibe a listagem das Atividades; A03.2 - O Ator seleciona a Atividade a ser excluída;

	<p>A03.3 - O Ator clica no botão Excluir;</p> <p>A03.4 - O Sistema exibe um modal para confirmação de exclusão:</p> <p style="padding-left: 20px;">A03.4.1 - O Ator clica em Sim:</p> <p style="padding-left: 40px;">A03.4.1.1 - O Sistema exclui a Atividade do Banco de Dados;</p> <p style="padding-left: 40px;">A03.4.1.2 - O Sistema retorna para P03;</p> <p style="padding-left: 20px;">A03.4.2 - Desistência de Exclusão:</p> <p style="padding-left: 40px;">A03.4.2.1- O Ator clica no botão Não;</p> <p style="padding-left: 40px;">A03.4.2.2 - O sistema retorna para P03.</p>
Fluxo de Exceção	<p>[E01] - Dados inválidos:</p> <p style="padding-left: 20px;">E01.1 - O Sistema exibe alerta mostrando os dados inválidos;</p> <p style="padding-left: 20px;">E01.2 - O Sistema retorna para o fluxo correspondente;</p> <p>[E02] - Registros não encontrados:</p> <p style="padding-left: 20px;">E02.1 - O Sistema exibe a interface de listagem sem registros.</p>
Caso de Uso ou Cenário Incluído	N/A
Observação	N/A

Na Tabela 3, é apresentado o Descritivo de Caso de Uso da funcionalidade de Gerenciar Planejamento - criação, edição, exclusão e consulta - ele possui vínculo com a atividade e pode ser utilizado no encontro.

Tabela 3. Descritivo de Caso de Uso - Gerenciar Atividades

Identificação	CDU02
Caso de Uso	Gerenciar Planejamento
Atores Principais	Administrador, Usuário
Atores Secundários	N/A
Pré-Condição	O ator deve estar logado no sistema
Pós-Condição	N/A
Fluxo Principal	<p>P01 - O Sistema apresenta a interface inicial do sistema;</p> <p>P02 - O Ator seleciona o menu Planejamntos.;</p> <p>P03 - O Sistema apresenta a interface de listagem de atividades;</p> <p>P04 - O Ator clica no botão de cadastrar Planejamento [A01], editar Planejamento [A02], excluir Planejamento [A03] ou filtrar Planejamntos [A04];</p> <p>P06 - O Ator preenche o campo Título;</p> <p>P07 - O Ator seleciona as Atividades que farão parte do Planejamento;</p> <p>P08 - O Sistema calcula a duração do planejamento em cima da quantidade de atividades e suas respectivas durações;</p> <p>P08 - O Ator clica no botão Cadastrar;</p> <p>P09 - O Sistema valida os dados [E01];</p> <p>P10 - O Sistema grava os dados no Banco de Dados;</p> <p>P11 - Fim do caso de uso.</p>

Fluxo Alternativo	<p>[A01] - Consultar Planejamento: A01.1 - O Sistema exibe a listagem dos Planejamentos; A01.2 - O Ator preenche os dados para pesquisa; A01.3 - O Sistema exibe os dados pesquisados [E02]; A01.4 - O Sistema retorna para P03;</p> <p>[A02] - Alterar Planejamento: A02.1 - O Sistema exibe a listagem dos Planejamentos; A02.2 - O Ator verifica e alterada os dados necessários; A02.3 - O Ator clica no botão Salvar; A02.4 - O Sistema valida os dados [E01]; A02.5 - O Sistema grava os dados no Banco de Dados; A02.6 - O Sistema retorna para P03;</p> <p>[A03] - Excluir Planejamento: A03.1 - O Sistema exibe a listagem dos planejamentos; A03.2 - O Ator seleciona o Planejamento a ser excluído; A03.3 - O Ator clica no botão Excluir; A03.4 - O Sistema exibe um modal para confirmação de exclusão: A03.4.1 - O Ator clica em Sim: A03.4.1.1 - O Sistema exclui o Planejamento do Banco de Dados; A03.4.1.2 - O Sistema retorna para P03;</p>
-------------------	---

Identificação	CDU02
Fluxo Alternativo	A03.4.2 - Desistência de Exclusão: A03.4.2.1 - O Ator clica no botão Não; A03.4.2.2 - O sistema retorna para P03.
Fluxo de Exceção	[E01] - Dados inválidos: E01.1 - O Sistema exibe alerta mostrando os dados inválidos; E01.2 - O Sistema retorna para o fluxo correspondente; [E02] - Pesquisa não encontrada: E02.1 - O Sistema exibe alerta mostrando que a busca não foi encontrada; E02.2 - O sistema retorna para o fluxo correspondente.
Caso de Uso ou Cenário Incluído	N/A
Observação	N/A

Na Tabela 4, é apresentado o Descritivo de Caso de Uso da funcionalidade de Gerenciar Encontros - criação, edição, exclusão e consulta - ele contém ao menos dois integrantes, como sendo os dois voluntários responsáveis pelo mesmo, as crianças e o planejamento.

Tabela 4. Descritivo de Caso de Uso - Gerenciar Encontros

Identificação	CDU02
---------------	-------

Caso de Uso	Gerenciar Encontros
Atores Principais	Administrador, Usuário
Atores Secundários	N/A
Pré-Condição	O ator deve estar logado no sistema
Pós-Condição	N/A
Fluxo Principal	<p>P01 - O Sistema apresenta a interface inicial do sistema;</p> <p>P02 - O Ator seleciona o menu Encontros;</p> <p>P03 - O Sistema apresenta a interface de listagem de Encontros;</p> <p>P04 - O Ator clica no botão de cadastrar Encontro [A01], editar Encontro [A02], excluir Encontro [A03] ou filtrar e listar Encontros [A04];</p> <p>P06 - O Ator preenche o campo Título;</p> <p>P07 - O Ator seleciona o local que vai ser apresentado o encontro;</p> <p>P07 - O Ator seleciona as atividades que serão apresentadas no encontro, podendo ser duas ou mais, mas respeitando o total de 2 horas de encontro;</p> <p>P08 - O Ator clica no botão Cadastrar;</p> <p>P09 - O Sistema valida os dados [E01];</p>

Identificação	CDU02
Fluxo Principal	<p>P10 - O Sistema grava os dados no Banco de Dados;</p> <p>P11 - Fim do caso de uso.</p>
Fluxo Alternativo	<p>[A01] - Filtrar e Listar Encontros:</p> <p>A01.1 - O Sistema exibe a listagem dos encontros;</p> <p>A01.2 - O Ator preenche os dados para pesquisa;</p> <p>A01.3 - O Sistema exibe os dados pesquisados [E02];</p> <p>A01.4 - O Sistema retorna para P03;</p> <p>[A02] - Editar Encontro:</p> <p>A02.1 - O Sistema exibe a listagem dos Encontros;</p> <p>A02.2 - O Ator verifica e alterada os dados necessários;</p> <p>A02.3 - O Ator clica no botão Salvar;</p> <p>A02.4 - O Sistema valida os dados [E01];</p> <p>A02.5 - O Sistema grava os dados no Banco de Dados;</p> <p>A02.6 - O Sistema retorna para P03;</p> <p>[A03] - Excluir Encontro:</p> <p>A03.1 - O Sistema exibe a listagem dos Encontros;</p> <p>A03.2 - O Ator seleciona o Encontro a ser excluído;</p> <p>A03.3 - O Ator clica no botão Excluir;</p> <p>A03.4 - O Sistema exibe um modal para confirmação de exclusão:</p> <p>A03.4.1 - O Ator clica em Sim:</p> <p>A03.4.1.1 - O Sistema exclui o Encontro do Banco de Dados;</p> <p>A03.4.1.2 - O Sistema retorna para P03;</p> <p>A03.4.2 - Desistência de Exclusão:</p> <p>A03.4.2.1 - O Ator clica no botão Não;</p>

	A03.4.2.2 - O sistema retorna para P03.
Fluxo de Exceção	[E01] - Dados inválidos: E01.1 - O Sistema exibe alerta mostrando os dados inválidos; E01.2 - O Sistema retorna para o fluxo correspondente; [E02] - Registros não encontrados: E02.1 - O Sistema exibe a interface de listagem sem registros.
Caso de Uso ou Cenário Incluído	N/A
Observação	N/A

Apêndice C. Projetar por funcionalidades

Esta etapa da metodologia tem como finalidade possibilitar o entendimento do fluxo de execução do sistema, quais as atividades que podem ser realizadas e como funciona seu tratamento dentro dele.

Na Figura 12 representa-se o Diagrama de Atividades, mostrando as ações que o usuário comum pode fazer dentro do sistema.

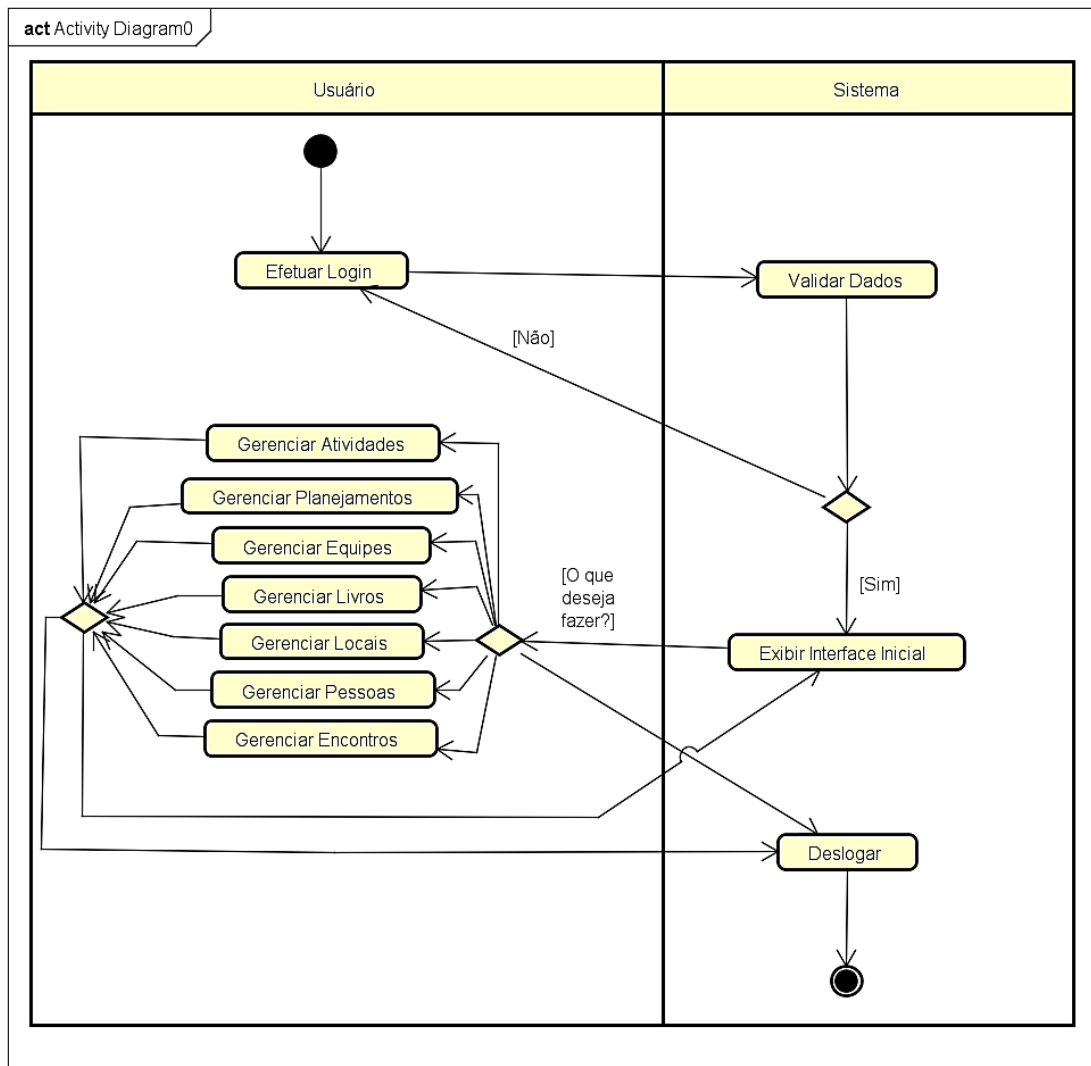


Figura 12. Diagrama de Atividades pelas funcionalidades do Usuário

Na Figura 13, apresenta-se o Diagrama de Atividades, mostrando as ações que o administrador pode fazer dentro do sistema, contendo todas as ações do usuário e mais as funcionalidades específicas.

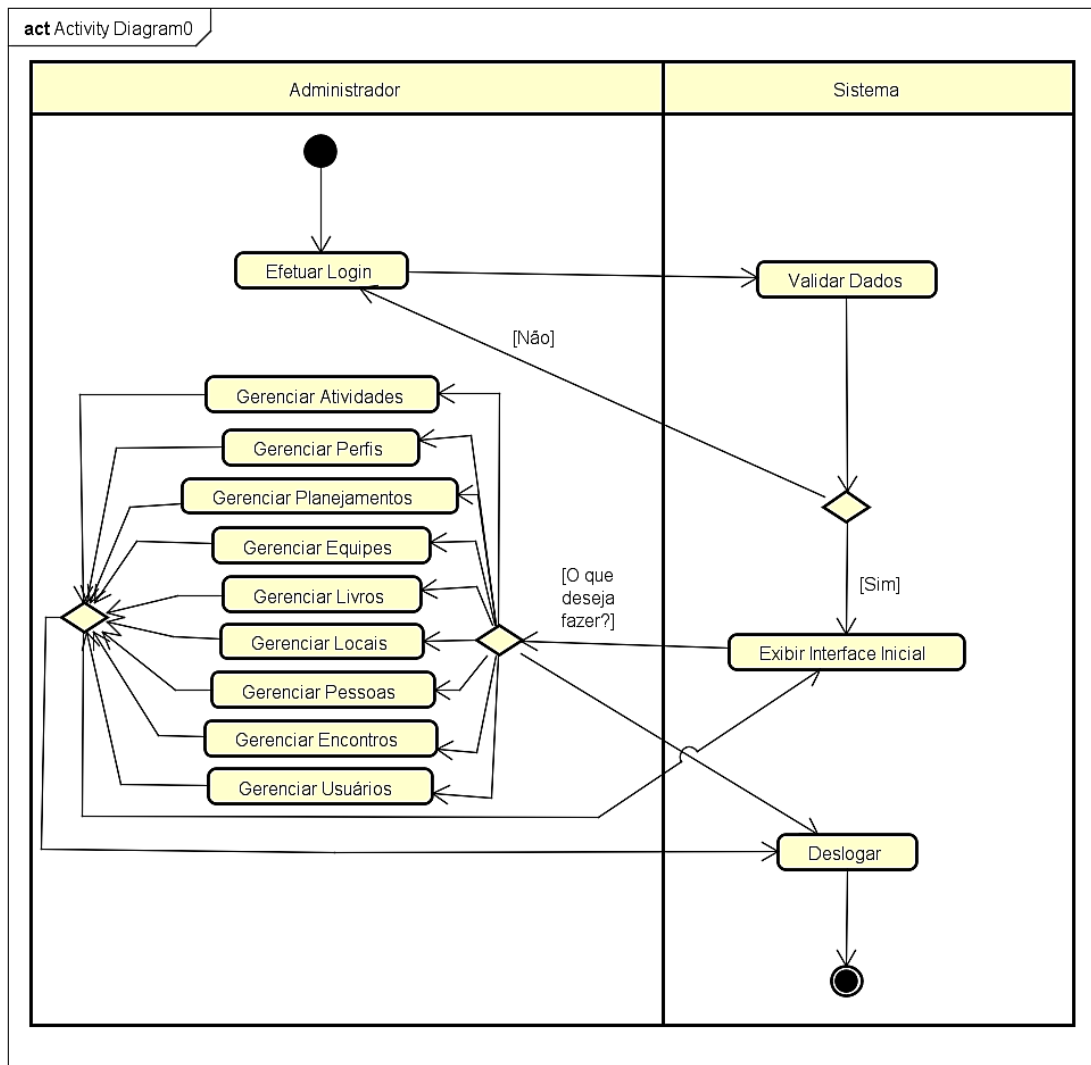


Figura 13. Diagrama de Atividades pelas funcionalidades do Administrador

Na Figura 14, apresenta-se o Diagrama de Atividades, mostrando o cadastro de planejamentos, visto que um planejamento é a somatória de uma série de atividades, juntamente com seus responsáveis e o local onde o mesmo será aplicado.

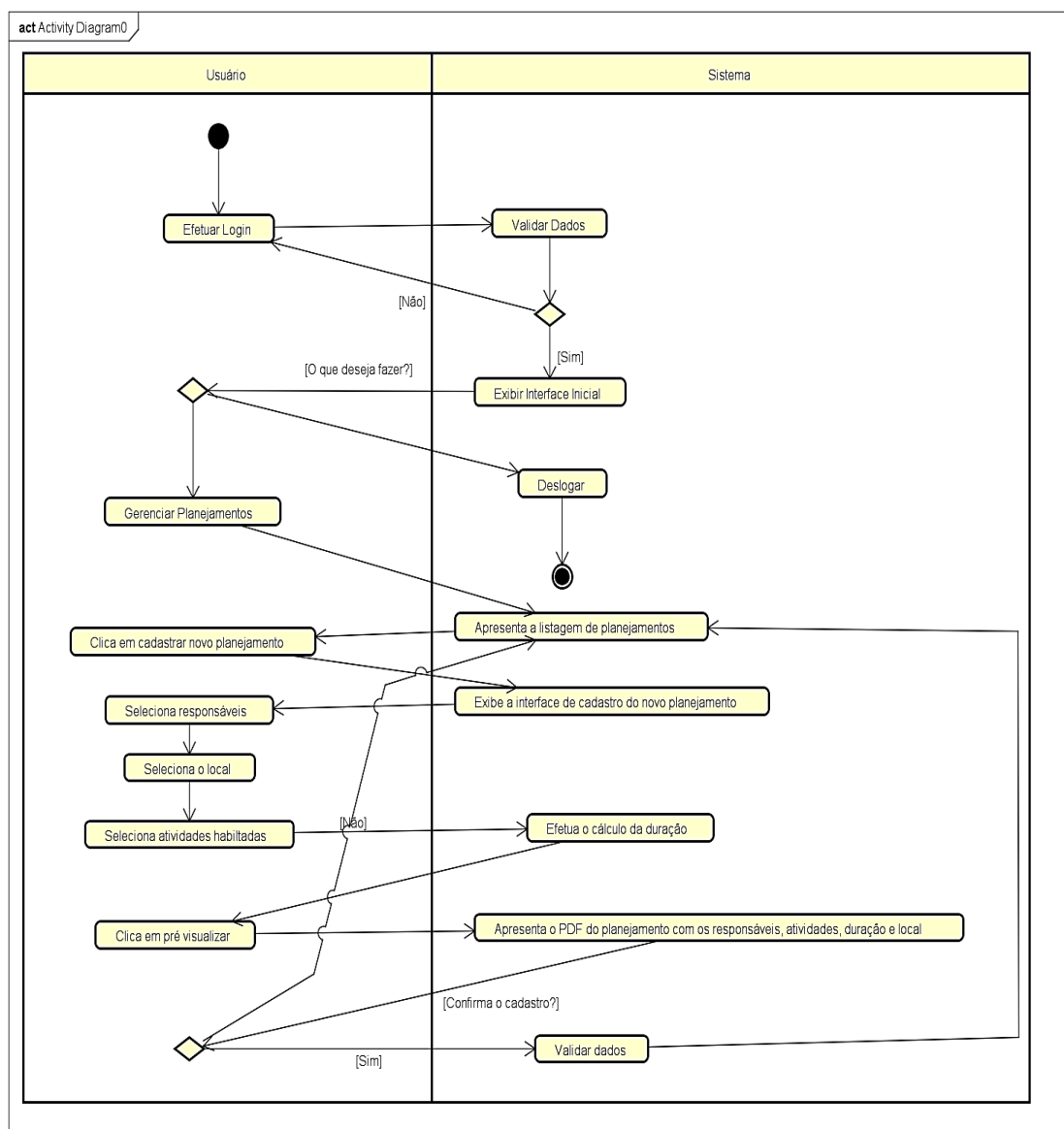


Figura 14. Diagrama de Atividades do cadastro de Planejamentos

Na Figura 15, representando a estrutura do Banco de Dados do projeto a seguir, tem-se o Diagrama Entidade Relacionamento (DER), apresentando, assim, as entidades existentes no software.

