UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CURSO DE TECNOLOGIA EM SISTEMAS PARA INTERNET

PEDRO HENRIQUE SERPA IDA

SISTEMA PARA CASA DE ACOLHIMENTO

MONOGRAFIA DE TRABALHO DE CONCLUSÃO DE CURSO DE GRADUAÇÃO

GUARAPUAVA
2021

PEDRO HENRIQUE SERPA IDA

SISTEMA PARA CASA DE ACOLHIMENTO

Monografia de Trabalho de Conclusão de Curso de graduação, apresentado à disciplina de Trabalho de Conclusão de Curso 2, do Curso Superior de Tecnologia em Sistemas para Internet – TSI – da Universidade Tecnológica Federal do Paraná – UTFPR – Câmpus Guarapuava, como requisito parcial para obtenção do título de Tecnólogo em Sistemas para Internet.

Orientador: Prof. Me. Guilherme da Costa Silva
Universidade Tecnológica Federal do Paraná - Câmpus Guarapuava

Coorientador: Prof. Dr. Roni Fabio Banaszewski
Universidade Tecnológica Federal do Paraná - Câmpus Guarapuava

GUARAPUAVA
2021

Ministério da Educação
Universidade Tecnológica Federal do Paraná
Câmpus Guarapuava
Curso Superior de Tecnologia em Sistemas para Internet

ATA DE DEFESA DE MONOGRAFIA DE TRABALHO DE CONCLUSÃO DE CURSO

No dia **21 de Maio de 2021**, às 15:00 horas, em sessão pública nas dependências da Universidade Tecnológica Federal do Paraná Câmpus Guarapuava, ocorreu a banca de defesa da de Trabalho de Conclusão de Curso intitulada: **"Sistema para Casa de Acolhimento"** do acadêmico **Pedro Henrique Serpa Ida** sob orientação do professor **Prof. Me. Guilherme da Costa Silva** do curso de Tecnologia em Sistemas para Internet.

Banca Avaliadora	
Membro	Nome
Orientador	Prof. Me. Guilherme da Costa Silva
Coorientador	Prof. Dr. Roni Fabio Banaszewski
Avaliador 1	Prof. Dr. Andres Jessé Porfirio
Avaliador 2	Prof. Me. Dênis Lucas Silva

Situação do Trabalho	
Situação	(x) Aprovado ( ) Aprovado com ressalvas ( ) Reprovado ( ) Não compareceu
Encaminhamento do trabalho para biblioteca	(x) Autoriza o encaminhado para biblioteca ( ) Manter sigilo para publicação ou geração de patente

Guarapuava, 21 de Maio de 2021.

A folha de aprovação assinada encontra-se na coordenação do curso (ou programa).

Dedico este trabalho a Universidade Tecnológica Federal do Paraná - Campus Guarapuava e aos professores e servidores da instituição.

AGRADECIMENTOS

Agradeço em primeiro lugar à minha família por ter me dado forças e motivação em toda minha caminhada acadêmica.

Agradeço aos meus amigos e colegas por sempre estarem presentes ajudando e motivando das mais diversas formas para que eu chegasse até aqui.

Agradeço também ao Universo por ter me ajudado a ver soluções em problemas nos momentos difíceis e ter clareza nos meus objetivos.

Agradeço a todos os professores da Universidade Tecnológica Federal do Paraná, em especial ao meu orientador Guilherme da Costa Silva, que contribuiu na realização deste trabalho.

O mais competente não discute, domina a sua ciência e cala-se. (Voltaire)

RESUMO

IDA, Pedro. Sistema para Casa de Acolhimento. 2021. 31 f. Monografia de Trabalho de Conclusão de Curso de graduação – Curso de Tecnologia em Sistemas para Internet, Universidade Tecnológica Federal do Paraná. Guarapuava, 2021.

As casas de acolhimentos e locais que possuem objetivos similares, ou seja, acolher pessoas que necessitam de um local para permanecer durante um período, possuem grande importância para aqueles que não têm condições de pagar por um hotel ou algo do gênero. Esses locais são em grande parte próximos a hospitais ou lugares que possuem um fluxo maior de pessoas com necessidades médicas ou idade avançada. A gestão destes locais ainda é feita utilizando blocos de notas ou planilhas virtuais, uma prática já ultrapassada nos dias atuais. Desta forma, buscando suprir a necessidade de um método de gerenciamento centralizado e coeso, este trabalho realizou o desenvolvimento de um software para tal fim. Para isso foi desenvolvido um sistema que gerencia pessoas acolhidas, suas estadias, consumo de refeições que foram cedidas pelo local e controle de doações.
Palavras-chave: Casa de acolhimento. Programa. Software. Gerenciamento.

ABSTRACT

IDA, Pedro. System for Shelter Homes. 2021. 31 f. Monografia de Trabalho de Conclusão de Curso de graduação – Curso de Tecnologia em Sistemas para Internet, Universidade Tecnológica Federal do Paraná. Guarapuava, 2021.

Shelter homes and places with similar purposes, like receiving people who need a place to stay for a period, have a great importance for those who have no conditions of paying for a hotel or suchlike. These establishments are mostly near hospitals or places with a large flow of people with medical needs or advanced age. The management of such establishments is made using notebooks or virtual worksheets, a practice that is considered obsolete now-a-days. Thus, in order to meet the need for a cohesive and centralized method of management, this study was carried out to develop a software for this purpose. To this end, a system was developed, which manages the admission of people, their accommodation, consumption of meals given by the institution and the donation control.

Keywords: Shelter homes. System. Software. Management.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

ABNT	Associação Brasileira de Normas Técnicas
APOO	Análise E Projeto Orientado A Objetos
CSS	Folhas de Estilo em Cascata (do inglês Cascading Style Sheets)
HTML	Linguagem de Marcação de Hipertexto (do inglês Hypertext Markup Language)
PHP	Pré-processador de hipertexto (do inglês Hypertext Preprocessor)
SPA	Aplicativo de página única (do inglês Single Page Application)
SQL	Structured Query Language

SUMÁRIO

1 INTRODUÇÃO

Atualmente, não é mais viável que a gestão de pessoas e locais seja em meios físicos de armazenamento, seja em papel ou em planilhas. Estas formas de controle, além de serem demoradas e exigirem atenção extra, podem gerar inconsistências em dados sensíveis, por exemplo, por meio da inserção de um valor errado.

Na área social, essa necessidade de uma gestão bem feita e organizada não é diferente, e em muitos casos ainda é feita em cadernos ou meios físicos de armazenamento, pois não há oferta de sistemas no mercado que atendam a essa demanda mais específica. Os sistemas disponíveis são geralmente mais robustos e atendem, primordialmente, empresas e indústrias. Nas casas de acolhimento, existem necessidades diferentes das oferecidas pelos sistemas de gestão existentes, por exemplo, a gestão de pessoas não é feita para gerir funcionários, suas atribuições, salários, etc, ao contrário, é utilizada para fazer o controle de pessoas que foram acolhidas na casa e pessoas que são voluntárias.

As casas de acolhimentos, alvos deste projeto, são locais disponibilizados para pessoas que, por algum motivo, não possuem um lugar para ficar durante um período de tempo. Por exemplo, o acompanhante de um paciente hospitalar que não pode pernoitar no hospital ou, ainda, convidados de algum evento religioso que não possuem onde passar a noite. Nessas situações, as pessoas, vindo de origens diversas, são orientadas a procurar uma casa de acolhimento, que irá ceder um local para dormir, as refeições diárias e o que mais estiver disponível com base nas suas necessidades. Estes locais, que em sua maioria possuem como financiadores filantropos e instituições religiosas, têm o objetivo de ajudar o próximo. Ainda, não é necessária uma automação ou controle mais tecnológico destes locais, porém o acúmulo de registros de hospedagens, compras, etc, em livros ou outros meios é obsoleto, logo um sistema pode trazer ganhos, ajudando na prestação de contas ou na angariação de fundos.

A proposta deste trabalho é preencher essa lacuna, com o desenvolvimento de um sistema que ajude no controle dos processos gerenciais desses locais, a fim de trazer mais integridade, segurança aos dados e praticidade. Com a premissa que os usuários desse sistema são pessoas, em sua maioria, sem conhecimentos práticos de informática, o uso deverá ser intuitivo e simples. Este requisito é o principal desafio para o desenvolvimento do sistema, porque algumas funções ou interações que são claras para pessoas que têm mais contato com informática, certamente não são tão claras para quem pouco usa sistemas informatizados no seu dia a dia.

1.1 OBJETIVO GERAL

Desenvolver um sistema para auxiliar na administração de uma casa de acolhimento.

1.2 OBJETIVOS ESPECÍFICOS

- Realizar o levantamento e análise dos requisitos necessários ao desenvolvimento do sistema proposto;
- Analisar técnicas, padrões e ferramentas para serem utilizadas no desenvolvimento do sistema;
- Desenvolver modelos e diagramas, a fim de estruturar o sistema proposto;
- Gerenciar e executar o desenvolvimento do sistema utilizando princípios da metodologia ágil;
- Obter conhecimento na utilização de planilhas para importação de dados em sistemas.
- Implementar o sistema, privilegiando o simples e intuitivo para solução do problema;
- Implementar testes automatizados para garantir estabilidade no sistema;

1.3 METODOLOGIA

Esta seção apresenta os procedimentos metodológicos para a resolução do problema apresentado. A metodologia seguiu os seguintes passos:

1. Para a obtenção dos requisitos foi realizada uma consulta a Casa de Acolhida e retiros Nazaré da Igreja Santa Cruz em Guarapuava, para encontrar o maior número de informações sobre os dados relevantes e necessários que o sistema deveria contemplar para uma melhor gestão do local;
2. Com base nos estudos e coletas, foram levantados os requisitos funcionais, permitindo entender as necessidades que foram abordadas neste projeto;
3. Após os requisitos funcionais serem encontrados foram feitas a modelagem do banco de dados e a prototipação das principais telas do sistema;
4. Com base em todas essas informações, foram escritas todas as histórias e tarefas referentes ao projeto seguindo a metodologia do Scrum para subdividir e otimizar o desenvolvimento das funções;
5. Tendo os passos anteriores definidos, foi aperfeiçoado o conhecimento no *framework* VueJs para melhorar as interações do usuário com o sistema;
6. Para o desenvolvimento geral, foi utilizado como base o *framework* PHP Laravel, devido aos conhecimentos prévios e recursos ofertados por tal *framework*;
7. Desenvolvimento de testes para garantia de integridade do sistema;

2 REVISÃO BIBLIOGRÁFICA

Até o presente momento, não foram encontrados sistemas gestores ou similares que sejam diretamente ligados ao objetivo de administrar uma casa de acolhimento ou lugares que tenham mesmo propósito.

Os sistemas que se aproximam do contexto atual desta proposta são destinados, em sua maioria, a atender Casas Lar, que são abrigos para crianças, adolescentes, mulheres que não possuem um local para viver devido a agressões sofridas em seus antigos lares, abandonados pelos pais ou outros cujas circunstâncias inviabilizam que voltem para seu lar. A diferença entre os sistemas desses locais e o desta proposta é que aqueles são focados em mostrar o local do abrigo, mostrar seus sistemas e receber doações para o financiamento do local, diferentemente deste que precisa ser focado na gestão e controle interno do local. A seguir, na próxima seção, são apresentados alguns sistemas que possuem funcionalidades que se assemelham às necessidades que o sistema proposto deva atender.

2.1 SISTEMAS SIMILARES

Um sistema que pode ser relacionado à proposta neste trabalho é o da Irmandade do Divino Espírito Santo (IDES, 2020), uma instituição religiosa que tem como foco o auxílio de crianças, adolescentes, jovens e suas famílias. O sistema possui um fórum para o conhecimento do que é feito pela instituição, sistemas presentes e um espaço de doações financeiras para o desenvolvimento e mantenimento dos sistemas. Ele possui uma página inicial que dá acesso a todos os recursos conforme a Figura 1.

Figura 1 – Tela principal site IDES.
Fonte: IDES (2020)

Outro, o HQBeds (2020), é um sistema gestor de pousadas que tem o foco no gerenciamento de reservas de quartos, vendas e controle de hóspedes. Esse sistema possui diversas funções que suprem a necessidade de todo tipo de hospedagem, e sua principal característica e função é o controle de clientes e hospedagens seguindo a Figura 2, mantendo histórico de hospedagens, clientes ativos, inativos e proibidos de novas estadias.

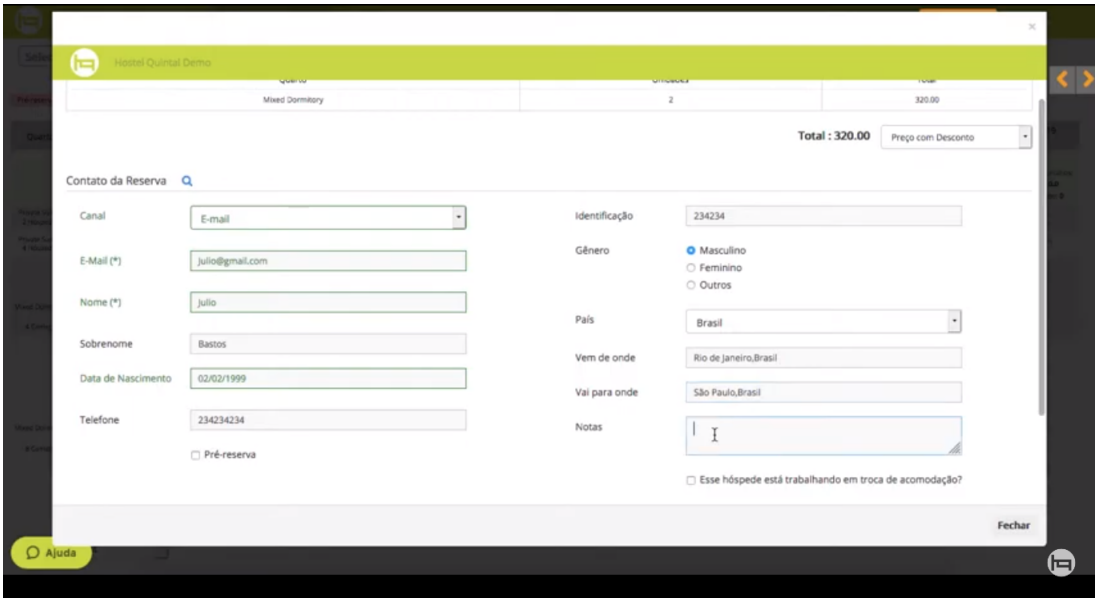


Figura 2 – Nova reserva na plataforma Hqbeds.
Fonte: HQBeds (2020)

Na mesma ideia de casas de acolhimento, há um sistema voltado para casas de repouso de idosos, o SisHOSP (2020). Esse sistema possui muitas funções voltadas para a área médica de cuidados com os idosos com prontuários, exames, contratos e afins. A similaridade com o projeto é o tratamento dos clientes como acolhidos no sistema e o controle destes, tendo o mesmo intuito de atender pessoas que necessitam de algum tipo de local ou cuidado específico, assim como pode ser verificado na Figura 3.

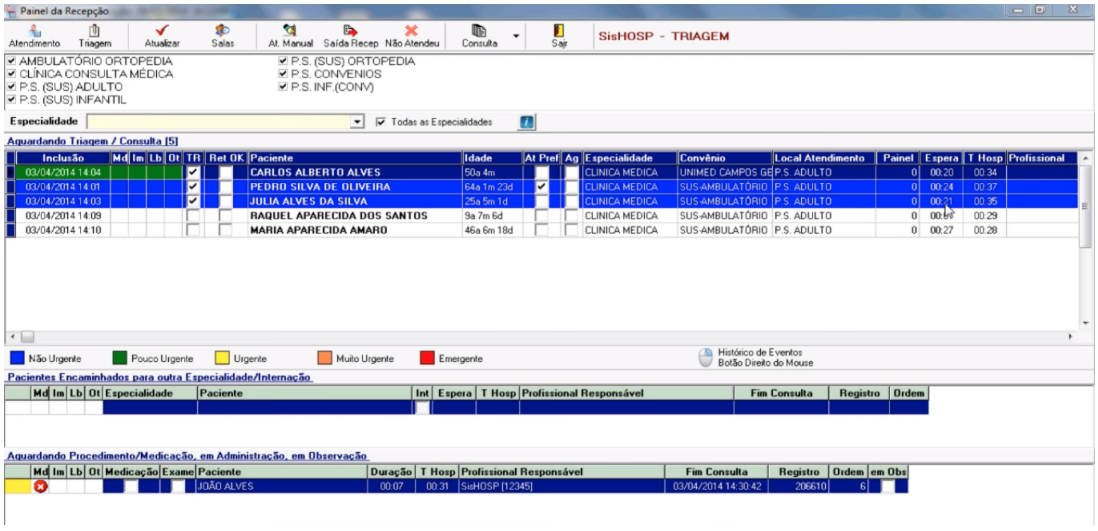


Figura 3 – Listagem de pacientes SisHOSP.
Fonte: SisHOSP (2020)

Cada um dos sistemas citados acima possuem suas próprias funções e características para auxiliar nos processos referentes ao seu contexto. O sistema deste trabalho busca inspiração

no site da organização (IDES, 2020) para trazer o lado acolhedor e fraterno do sistema, buscando a partir dele usar uma interface simples. O sistema da HQBeds (2020), provê a base para a criação de estadias, a área de hospedagens, abstraindo somente os atributos relevantes para o sistema. Do sistema da SisHOSP (2020) é utilizada a ideia do tratamento dos pacientes, que no caso são os acolhidos, para trabalhar com o controle dos mesmos de forma a trabalhar somente com dados estritamente necessários para realizar uma acolhida. Não é utilizada inspiração em interface do último sistema para essa área, por ser muito poluída e demandar de conhecimentos aprimorados do sistema para que seja utilizado.

2.2 REFERENCIAL TEÓRICO

No desenvolvimento de sistemas, necessita-se certa criatividade e proatividade, já que o objetivo é sempre replicar coisas do mundo real para o virtual, visando facilitar e auxiliar processos que podem ser burocráticos e demorados. Para cada tipo de problema atacado, há diferentes formas de ser resolvido com diferentes tecnologias, linguagens de sistemação e ferramentas. Como todo processo, para que haja um resultado concreto, é preciso utilizar um padrão e uma metodologia.

No desenvolvimento de sistema, são utilizadas metodologias para gerenciar sua execução. Uma das metodologias mais utilizadas é o Scrum (2020), que foi criada por Jeff Sutherland e Ken Schwaber. Segundo o artigo da DesenvolvimentoAgil (2020), o Scrum consiste em otimizar o processo de desenvolvimento de um projeto dividindo-o em pequenas entregas com prazos delimitados, seja quinzenal, semanal e assim por diante, intercalando com reuniões da equipe para alinhamento das tarefas. Essas entregas são chamadas de Sprints, que são subdivididas em entregas menores a fim de tornar os processos mais independentes e fracionados para que seja possível realizar entregas mais frequentes para o cliente. Todas as funcionalidades são mantidas em uma lista chamada de Product Backlog antes de serem inseridas em uma Sprint para serem desenvolvidas. Todos os passos do fluxo de trabalho do Scrum podem ser vistos conforme a Figura 4.

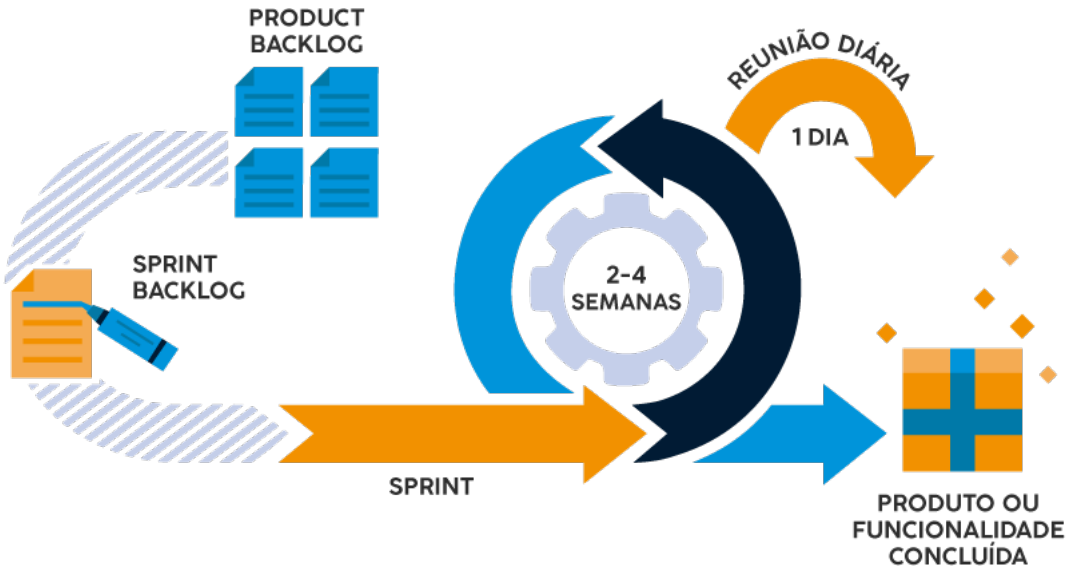


Figura 4 – Metodologia ágil Scrum.
Fonte: Tecnicon (2020)

Mas por que utilizar uma metodologia ágil para o desenvolvimento de um projeto mesmo não tendo proporções tão grandes? Quando se utiliza um modelo como este para o processo de desenvolvimento, é possível otimizar o tempo e a qualidade da entrega final do produto em questão. A ideia das metodologias ágeis é trazer um modelo de negócio que disponibilize entregas menores do produto, porém com uma constância maior, ou seja, consegue-se mostrar ao responsável do sistema a evolução do desenvolvimento e realizar alterações e correções de modo a não afetar de forma tão abrupta o tempo final de entrega do produto.

No desenvolvimento de aplicações monolíticas, em que camadas de interface e lógica estão no mesmo lugar, geralmente, é utilizada uma linguagem de sistemação para implementar a lógica das regras de negócio. Já para criar as interfaces de interação com o usuário há algumas possibilidades para se trabalhar, podendo realizar a junção de várias ferramentas ou utilizar somente uma para as regras de negócios. No momento existem algumas ferramentas *front-end* disponíveis no mercado, como por exemplo ReactJS, VueJs, JQuery, que são completas e cada uma possui sua especialidades buscando o melhor resultado para o sistema.

Na maior parte dos sites ou sistemas *web* presentes na internet, é utilizado o *framework front-end* Bootstrap, que utiliza HTML, CSS e Javascript. O objetivo dele é otimizar o desenvolvimento através da adoção de uma estrutura única de codificação de folhas de estilo pra diminuir inconsistências entre as forma de codificar que variam entre desenvolvedores. A tentativa deu tão certo que seus desenvolvedores perceberam o grande potencial da ferramenta, lançando-a no GitHub como um *software open source* (HOMEHOST, 2020). Essa ferramenta possibilita a criação de estruturas e modelos de sites e sistemas, que podem ir desde páginas

estáticas e com poucas informações até interfaces complexas e com grande quantidade de elementos presentes na tela, contando com um modelo de estruturação baseado em colunas sendo possível subdividir e ajustar os componentes da tela em até 12 colunas e ainda oferece um ótimo suporte a responsividade para utilização em dispositivos com telas menores. A estrutura de classes de estilo dessa ferramenta possibilita sua customização para que possa ser ajustada de forma simples de intuitiva para atender aos padrões de cores que são requisitados ao projeto trabalhado. Além de tudo isso, as animações e ações dinâmicas que a ferramenta oferece permite que a utilização do sistema se torne muito mais orgânica e fluída para o usuário final.

O gerenciamento de requisições a um site web deve ser realizado por um servidor web, o qual mantém as páginas online para serem acessadas pelos usuários 100% do tempo sem a dependência de um serviço ser executado e mantido manualmente por alguém. O servidor web nada mais é que um computador que possui hardware e software preparados para realizar o controle de acessos a sites e sistemas online. Os servidores web podem possuir duas distinções que definem seu tipo de dado trabalhado segundo a documentação de desenvolvedores da mozilla (2020). O primeiro tipo chamado estático não possui softwares adicionais para trabalhar com dados extras, ou seja, tudo que pode ser requisitado pelo usuário já está presente nos arquivos que foram armazenados no servidor, como mostra a Figura 5.

Figura 5 – Requisição estática.
Fonte: vrsys (2020)

O segundo tipo é chamado de dinâmico, pois os dados que são requisitados podem ser trabalhados e buscados em aplicações e em bancos de dados. Sendo assim os dados podem ser atualizados pelo servidor antes de serem entregues ao usuário, segundo a Figura 6.

Figura 6 – Requisição dinâmica.
Fonte: vrsys (2020)

Um dos servidores mais utilizados há tempos é o Apache, um servidor *web open source* mantido pela Apache Software Foundation e utilizado por mais de 45% dos sites da internet segundo a hostinger (2020). Sua função é servir dados para os sites na internet. Para fazer isso, ele age como um intermediador entre o servidor físico e os computadores dos clientes.

No desenvolvimento web, umas das linguagens que mais se destaca por trazer bons resultados é o PHP. Ele foi criado em 1994 por Rasmus Lerdorf e se tornou muito utilizado no desenvolvimento pela pequena curva de aprendizagem e grande número de recursos que oferece. Ele é *open source*, geralmente utilizado junto com o HTML. Desta forma o PHP executado em um servidor possibilita a geração de conteúdo dinâmico que as páginas de um sistema necessitam (PHP, 2020). Essa linguagem possui recursos muito completos, que suprem as necessidades tanto de sistemas mais robustos, quanto de sites básicos com menos lógica, e por isso está entre as linguagens mais utilizadas conforme a Figura 7.

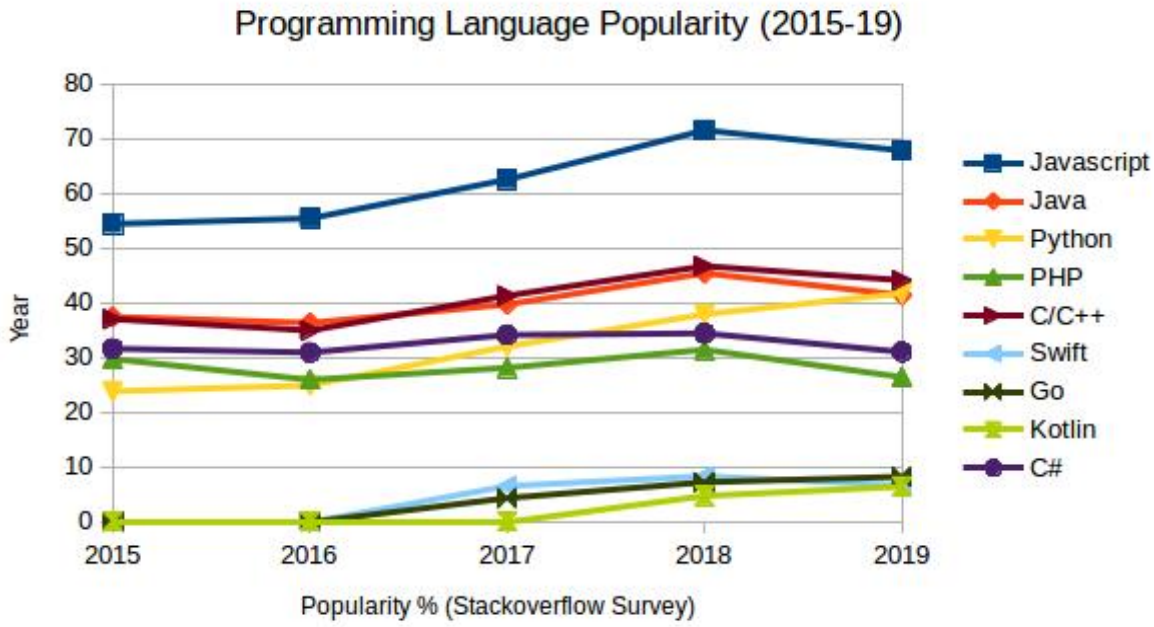


Figura 7 – Linguagens mais populares 2015-2019.
Fonte: CodingInfinite (2020)

A linguagem deixou de ser utilizada no desenvolvimento a partir do zero e passou a ser utilizada com *frameworks* que tratam da parte verbosa e básica do funcionamento web, deixando a cargo dos desenvolvedores somente o que é realmente específico do sistema em desenvolvimento, otimizando todo o processo e dando mais segurança e estabilidade ao sistema. Dentre os frameworks PHP, podem ser citados os três mais utilizados: Laravel, CodeIgniter e Symfony, conforme a Figura 8.

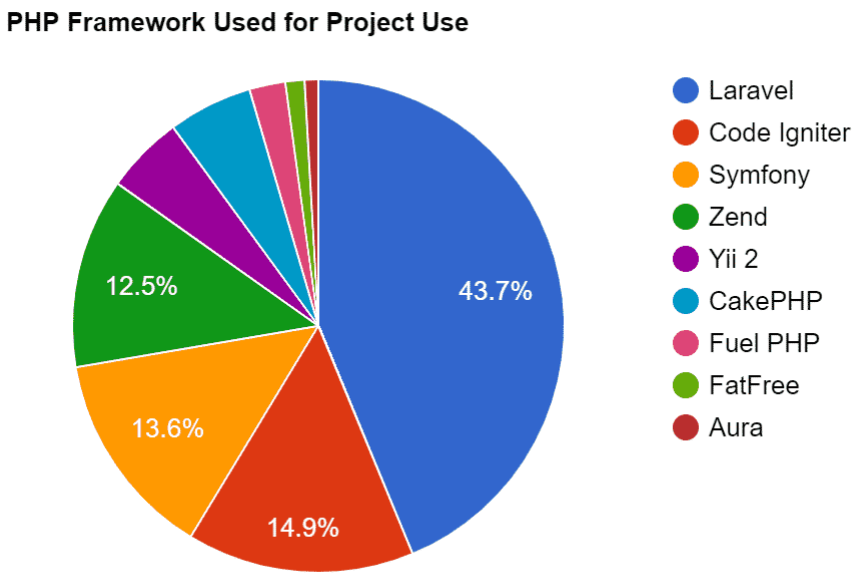


Figura 8 – Frameworks PHP.
Fonte: easy2becoder (2020)

O *framework* Laravel possui uma comunidade grande e disponibiliza constantemente atualizações, proporcionando diversas funcionalidades e otimizações para sistemas de pequeno a grande porte.

O VueJs (2020) é um *framework* javascript componentizado, isto é, trabalha com componentes que podem ser reutilizados e divididos conforme suas responsabilidades. Ele é chamado de progressivo porque pode ser incrementado aos poucos mesmo que haja outro *framework* sendo utilizado, tornando-o assim de rápida adoção em projetos que já estão sendo desenvolvidos. O VueJS não é somente uma ferramenta para construção de interface, ele possui um ecossistema completo que permite realizar toda a parte de *front-end* sem necessitar da utilização de outras ferramentas, portanto, sendo possível utilizá-lo para desenvolver uma aplicação Single Page Application(SPA) ou como ferramenta de construção de interface. Ademais a comunidade dele é muito extensa, assim como sua documentação.

A utilização de *frameworks front-end* proporciona aos usuários uma interface extremamente fluída e dinâmica, elevando a experiência de usuário para outro nível, tornando funcionalidades que por vezes podem ser complexas, em simples e pequenos componentes na tela, que se montados da forma correta, são intuitivos até mesmo aos usuários que não possuem tanta familiaridade com tecnologia.

A grande maioria dos sistemas tem como principal função a manipulação de dados, seja através da interação com usuários ou, ainda, através de comunicação direta com outros sistemas, em geral, solicitando e apresentando informações. Estes dados, enquanto necessários e úteis, precisam ser armazenados para posterior uso. A forma comumente utilizada para resolver isso é a adoção de um bancos de dados.

Os bancos de dados podem ser divididos em dois tipos de arquiteturas: relacionais e não-relacionais. Os relacionais possuem estrutura divida em tabelas, possuindo relacionamentos com outras tabelas para mostrar a conexão existente entre os dados de tabelas distintas. Os não-relacionais trabalham com todos os dados interligados por chaves e valores, não possuindo separação.

Os bancos relacionais são utilizados na grande maioria dos sistemas existentes com banco de dados, devido ao seu desempenho para trabalhar com muitos dados e seus relacionamentos. Para trabalhar com esses bancos deve ser utilizado um Sistema Gerenciador de Banco de Dados (SGBD) como é explanado no livro (DATE, 2004). São bancos relacionais, entre outros, o MySQL, MariaDB, PostgreSql e Oracle.

A manipulação dos dados, nestes bancos, ocorre através do uso de Standard Query Language (SQL), que é a linguagem padrão dos bancos de dados relacionais e com ela pode-se realizar todas as ações necessárias para manipulação desses dados armazenados. Apesar de ser um padrão, a forma de fazer algo em um determinado banco pode ser diferente em outro. Por exemplo, a declaração de funcionamento de gatilhos em tabelas e até mesmo a escrita de uma consulta pode variar conforme o banco de dados escolhido.

Em um sistema *web*, as ações tomadas pelo usuário devem ser realizadas através do navegador, que é o meio de comunicação entre o usuário e o sistema. O usuário realiza o acesso a uma página *web*, que pode ser desenvolvida utilizando as ferramentas citadas acima, e envia uma requisição ao servidor. Este então processa os dados enviados e encaminha-os ao interpretador da linguagem, neste caso o PHP sendo utilizado pelo Laravel, que executa as instruções referentes à requisição juntamente dos dados enviados pelo usuário, seja realizando uma consulta ou alteração no banco de dados, a solicitação de um arquivo ou até mesmo o disparo de um e-mail. Após a ação ser realizada, o servidor deve retornar uma resposta ao usuário referente ao que foi requisitado, podendo ser um redirecionamento, uma listagem de dados ou uma confirmação se a ação foi bem sucedida. Essa resposta é interpretada no servidor e devolvida para o usuário em seu navegador. Esse fluxo está ilustrado na Figura 9.

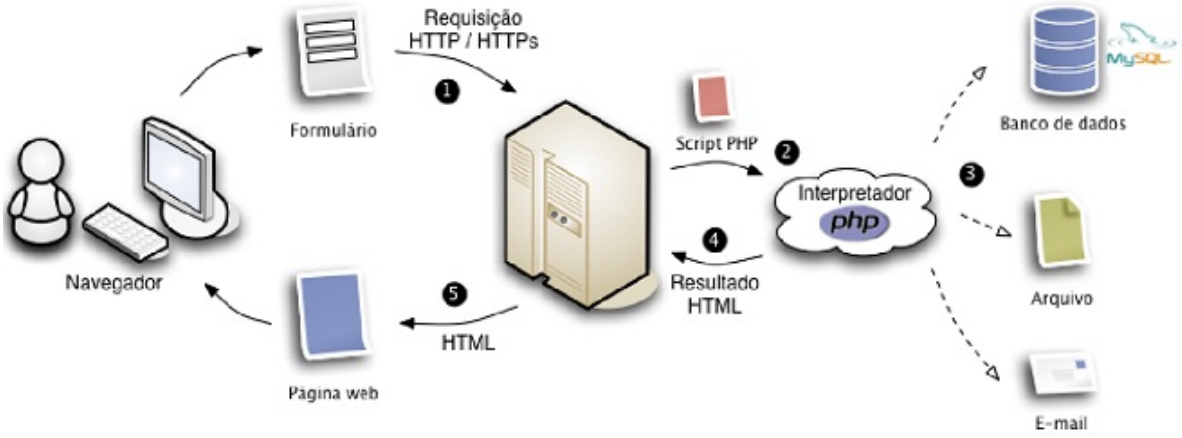
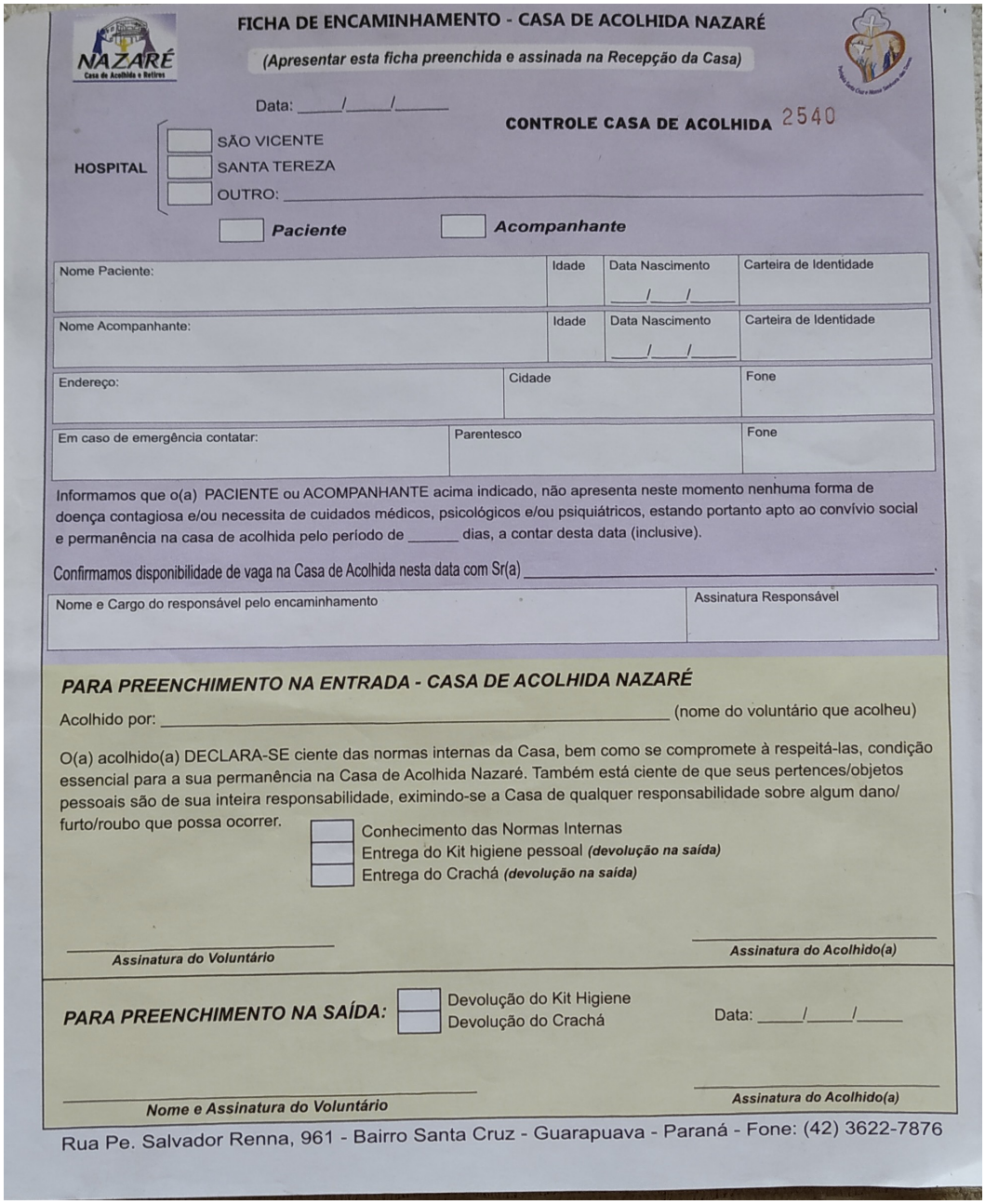


Figura 9 – Fluxo web.
Fonte: MARINHO (2020)

3 DESENVOLVIMENTO

Neste capítulo serão descritas as fases de desenvolvimento das soluções das questões que foram levantadas os capítulos anteriores.

A casa de acolhimento utilizada como base para este projeto não possui atributos ou necessidades que demandem um desenvolvimento ou um estudo muito específico, tendo ela um funcionamento simples como qualquer outra no modelo de negócio para trabalhar com um novo acolhimento. Para toda nova entrada nesta casa de acolhimento, deve-se possuir uma ficha com os dados do acolhido e do local de onde o mesmo foi enviado, como pode ser visto na Figura 10. As informações inerentes a instruções e assinaturas de responsáveis voluntários não é algo que deve estar no domínio do Software, portanto não foi levado em consideração no momento da implementação.



FICHA DE ENCAMINHAMENTO - CASA DE ACOLHIDA NAZARÉ

NAZARÉ Casa de Acolhida e Retiros

(Apresentar esta ficha preenchida e assinada na Recepção da Casa)

Data: ____/____/____

CONTROLE CASA DE ACOLHIDA 2540

HOSPITAL ☐ SÃO VICENTE ☐ SANTA TEREZA ☐ OUTRO: _____

☐ *Paciente* ☐ *Acompanhante*

Nome Paciente:	Idade	Data Nascimento	Carteira de Identidade
		____/____/____	

Nome Acompanhante:	Idade	Data Nascimento	Carteira de Identidade
		____/____/____	

Endereço:	Cidade	Fone

Em caso de emergência contatar:	Parentesco	Fone

Informamos que o(a) PACIENTE ou ACOMPANHANTE acima indicado, não apresenta neste momento nenhuma forma de doença contagiosa e/ou necessita de cuidados médicos, psicológicos e/ou psiquiátricos, estando portanto apto ao convívio social e permanência na casa de acolhida pelo período de _____ dias, a contar desta data (inclusive).

Confirmamos disponibilidade de vaga na Casa de Acolhida nesta data com Sr(a) _____.

Nome e Cargo do responsável pelo encaminhamento	Assinatura Responsável

PARA PREENCHIMENTO NA ENTRADA - CASA DE ACOLHIDA NAZARÉ

Acolhido por: _____ (nome do voluntário que acolheu)

O(a) acolhido(a) DECLARA-SE ciente das normas internas da Casa, bem como se compromete à respeitá-las, condição essencial para a sua permanência na Casa de Acolhida Nazaré. Também está ciente de que seus pertences/objetos pessoais são de sua inteira responsabilidade, eximindo-se a Casa de qualquer responsabilidade sobre algum dano/ furto/roubo que possa ocorrer.

☐ Conhecimento das Normas Internas
☐ Entrega do Kit higiene pessoal ***(devolução na saída)***
☐ Entrega do Crachá ***(devolução na saída)***

_____ ***Assinatura do Voluntário***

_____ ***Assinatura do Acolhido(a)***

PARA PREENCHIMENTO NA SAÍDA: ☐ Devolução do Kit Higiene ☐ Devolução do Crachá Data: ____/____/____

_____ ***Nome e Assinatura do Voluntário***

_____ ***Assinatura do Acolhido(a)***

Rua Pe. Salvador Renna, 961 - Bairro Santa Cruz - Guarapuava - Paraná - Fone: (42) 3622-7876

Figura 10 – Ficha de entrada.

3.1 LEVANTAMENTO DE REQUISITOS

O levantamento de requisitos do sistema foi realizado através de entrevistas com os voluntários da casa de acolhimento juntamente com visitas feitas ao local para o melhor entendimento do ambiente no qual o sistema deveria ser implantado. Com base nas informações obtidas, foi possível definir os requisitos funcionais e não funcionais necessários conforme os Quadros 1 e 2.

Quadro 1 – Requisitos funcionais.

Requisitos funcionais
1. O sistema deve dispor de login somente para administradores e voluntários
2. O administrador deverá ter permissão para cadastrar novas origens de acolhidos
3. O administrador deverá ter permissão para cadastrar novos acolhidos
4. O administrador deverá ter permissão para proibir um acolhido de ser recebido
5. O administrador deverá ter permissão para cadastrar novos administradores e voluntários
6. O administrador deverá ter permissão para cadastrar uma nova estadia
7. O voluntário deverá ter permissão para cadastrar uma nova estadia
8. O administrador deverá ter permissão para editar uma estadia que está em aberto
9. O voluntário deverá ter permissão para editar uma estadia que está em aberto
10. O administrador deverá ter permissão para editar a quantidade de refeições cedidas em um dia
11. O administrador deverá ter permissão para visualizar gráficos referentes às refeições cedidas
12. O voluntário deverá ter permissão para visualizar gráficos referentes às refeições cedidas
13. O administrador deverá ter permissão para importar planilhas de estadias realizadas antes da implantação do sistema
14. O administrador deverá ter permissão para realizar o controle de doações recebidas
15. O sistema deve possuir dois perfis de usuários: administrador e voluntário
16. O sistema não deve permitir duas estadias simultâneas do mesmo acolhido

Quadro 2 – Requisitos não funcionais.

Requisitos não funcionais
1. O sistema deve possuir uma interface simples e intuitiva
2. O sistema deve possuir interface leve para carregamento
3. O sistema deve ser acessível de qualquer lugar e de qualquer dispositivo

3.2 MODELAGEM DO BANCO

Tendo todos os requisitos do sistema, sendo eles funcionais ou não, bem definidos e descritos pelos usuários, foi possível elaborar o esquema completo do banco de dados para atender as necessidades de todas as funcionalidades requisitadas. O banco foi desenhado de forma a atender os requisitos do momento da implementação do sistema. O módulo da aplicação que gerencia as doações recebidas foi isolado, por conta de sua independência com qualquer outro tipo de dado inerente ao seu domínio.

Todo o esquema do banco pode ser visualizado na Figura 11. Conforme foi elaborado o banco de dados e priorizados os recursos do sistema, foi montado o modelo Scrum para a realização das tarefas na cronologia em que foi realizado o desenvolvimento das telas.

O banco de dados escolhido para o sistema foi o PostgreSQL (POSTGRESQL, 2020) devido ao seu bom desempenho para trabalhar com dados relacionais, sua otimização em consultas e a possibilidade de utilização de *plugins* que melhoram as interações que são requisitadas pelos usuários.

As tabelas do sistema e suas funções são as seguintes:

- **permissions**: Responsável por armazenar as permissões que o sistema possui;
- **roles**: Responsável por armazenar os perfis de usuário (administrador e voluntário);
- **model_has_roles**: Tabela intermediária que controla os tipos de perfis que uma entidade pode possuir no sistema;
- **hole_has_permissions**: Tabela intermediária que controla quais permissões um determinado perfil possui;
- **model_has_permissions**: Tabela intermediária que controla quais permissões uma determinada entidade possui;
- **password_resets**: Armazena as requisições de recuperação de senha;
- **media**: Controla os arquivos armazenados no sistema, no caso as planilhas de importação de estadias;
- **meals**: Controle das refeições que são cedidas no local por dia;
- **sources**: Armazena as origens que as estadias podem possuir;
- **stays**: Tabela principal do sistema que controla as estadias de acolhidos;
- **users**: Tabela que contém os administradores e voluntários do sistema;

- **clients**: Armazena os acolhidos e seus dados;
- **cities**: Possui as cidades brasileiras as quais um acolhido pode pertencer;
- **states**: Possui os estados brasileiros que uma cidade pode pertencer;
- **donations**: Controle das doações que a casa recebe;
- **donation_units**: Armazena as unidades que as doações possuem. Ex:kg, unidade, peça;
- **donation_categories**: Armazena as categorias que as doações possuem. Ex:roupa, alimento;

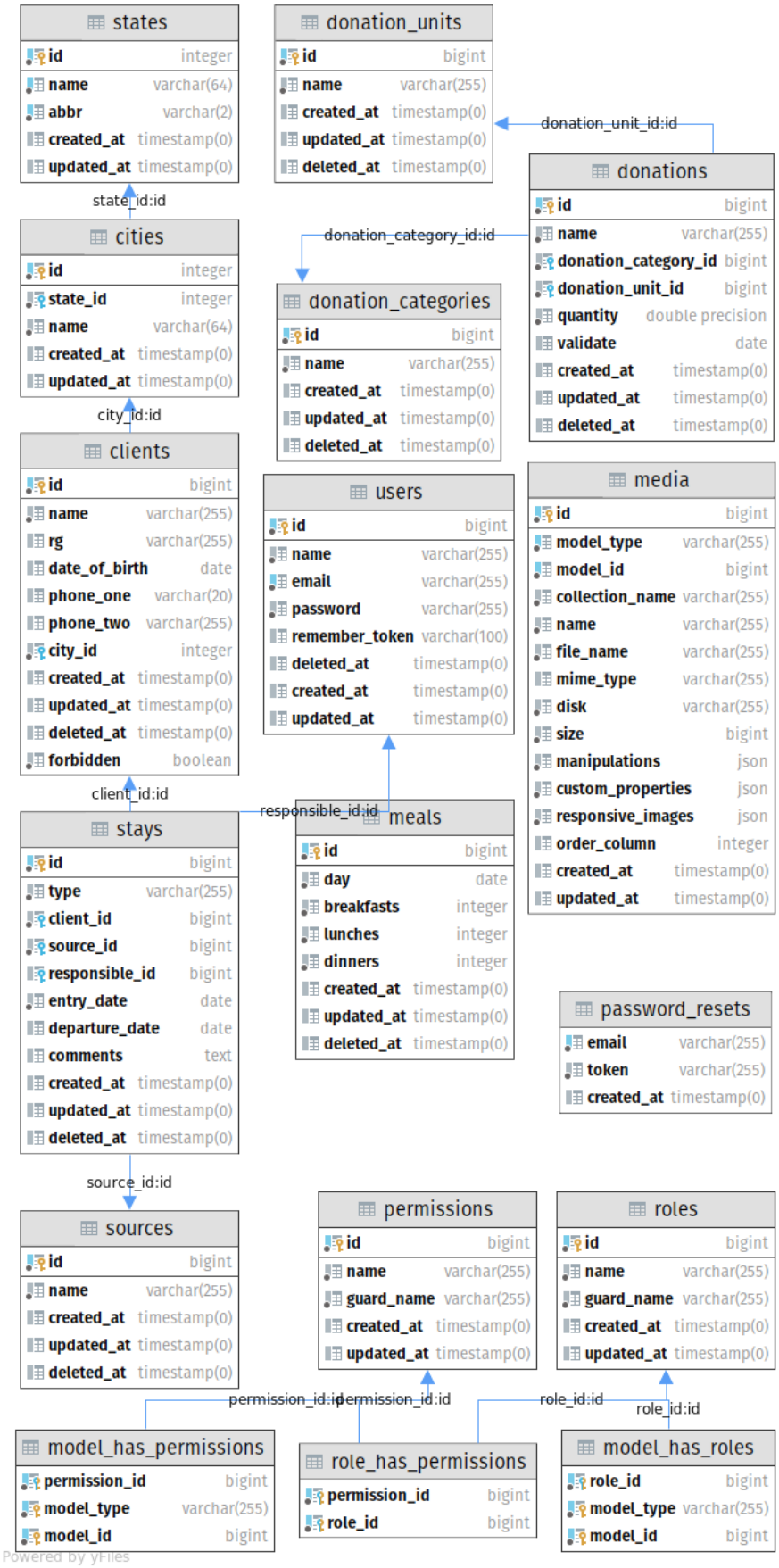


Figura 11 – Diagrama de Entidade Relacionamento.

3.3 DESENVOLVIMENTO DAS TELAS

Para o desenvolvimento do layout do sistema, foi utilizado o *template web* Stisla (STISLA, 2020) que provê uma interface muito simples e bonita com várias ferramentas visuais que suprem as necessidades do projeto para ser acessível.

Após a definição do banco de dados foi dado início ao processo de desenvolvimento das telas do sistema. Como os usuários do sistemas, em sua maior parte, não possuem muito contato com tecnologia, todas as telas devem possuir o mínimo de campos possíveis.

Como primeiro controle necessário, foram feitas as telas de controle de usuários administradores e voluntários do sistema, diferenciando-os por suas permissões no sistema. Ao realização a primeira carga de dados do sistema, já acontece a inserção de um usuário de cada tipo para interação. A tela de cadastro de usuário possui poucos atributos, sendo tanto para administradores quanto para voluntários como pode ser visto nas Figuras 12 e 13.

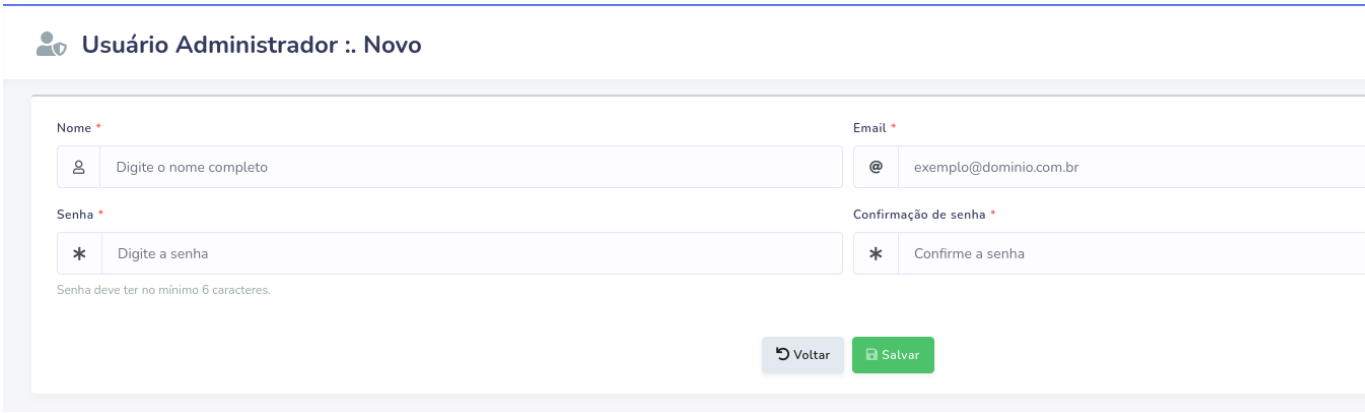


Figura 12 – Criação de novo usuário administrador.

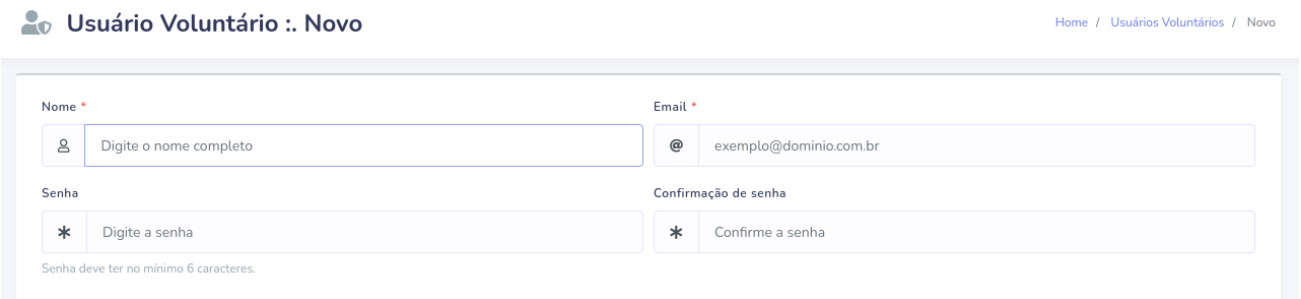


Figura 13 – Criação de novo usuário voluntário.

Após isso foi desenvolvida a área de origens de acolhidos, por ser básica, que devia possuir apenas o nome do local, como pode ser visto nas Figuras 14 e 15.

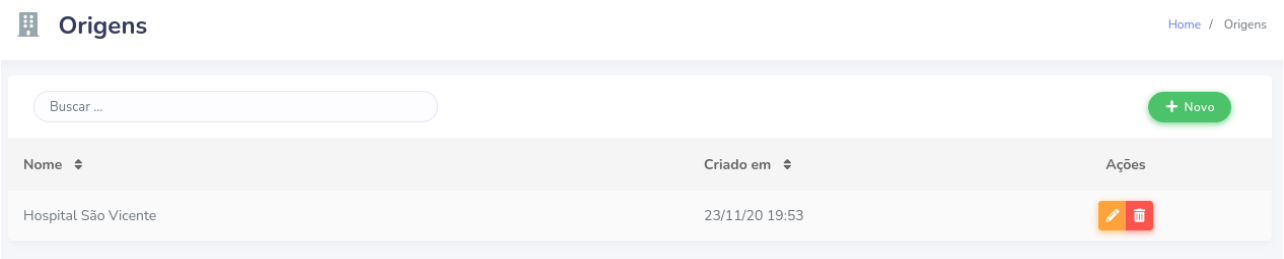


Figura 14 – Lista de origens.

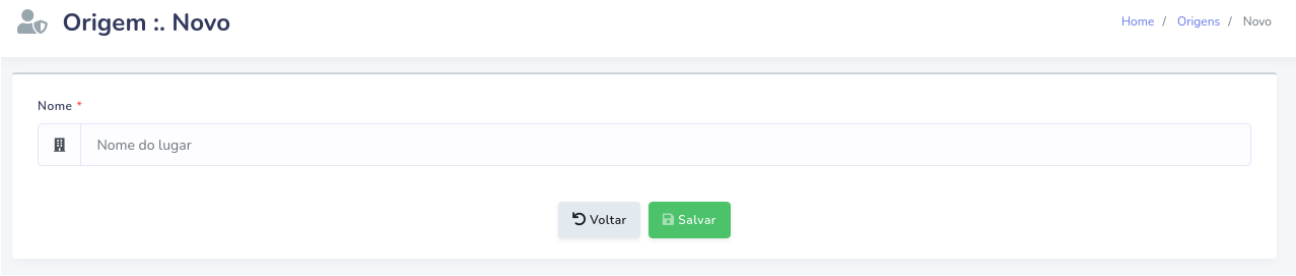


Figura 15 – Criação de nova origem.

Depois das origens foi desenvolvida a área de acolhidos que demandou um certo cuidado para ser feita devido aos dados mais trabalhados, como a data de nascimento, que foi feita com um calendário da biblioteca Vuejs-Datepicker (VUEJSDATEPICKER, 2020) para deixar mais amigável a interface, documento de identificação e dados de cidade e estado do acolhido. Os campos de cidade e estado, que eram dependentes entre si, foram desenvolvidos utilizando a tecnologia ajax. A tela de cadastro de acolhido pode ser vista na Figura 16.

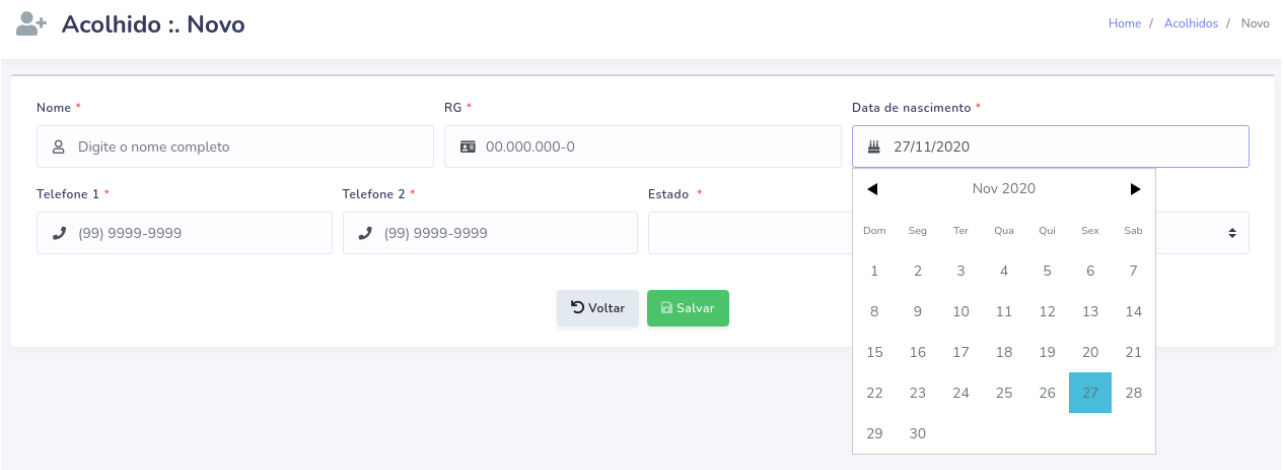


Figura 16 – Criação de novo acolhido.

Tendo as áreas de acolhidos, origens e usuário finalizadas, o principal recurso do sistema, que é a área de estadias, já podia ser implementada por conta da conclusão das áreas supracitadas. Esta demandou um certo tempo para ser elaborada e desenvolvida por ser a área do sistema que é responsável por popular o principal recurso. Para selecionar o acolhido e o

responsável por ele foi utilizada a biblioteca Select2 (SELECT2, 2020) para facilitar as buscas por nomes. Para as datas de entrada e saída foi utilizada a mesma biblioteca Vuejs-Datepicker. O resultado da tela de criação de uma nova acolhida pode ser visto na Figura 21.

As estadias possuem um recurso de importação de planilha, que foi desenvolvido para auxiliar na sincronização dos dados antigos da casa de acolhimento, que foram passados para uma planilha, para que não fosse necessário realizar o cadastro manual de anos de acolhidas e acolhidos. A importação da planilha foi colocada para executar em segundo plano devido ao tamanho da planilha, que poder ser extensa, evitando que os usuários necessitem esperar na tela até que ela seja finalizada. Nos últimos meses, essas informações começaram a ser passadas para uma planilha em Excel para o melhor controle dos dados, e isso facilitou para que fosse planejada uma funcionalidade neste sistema para importar essa planilha, a fim de agilizar o processo de sincronização dos dados antigos com o novo sistema.

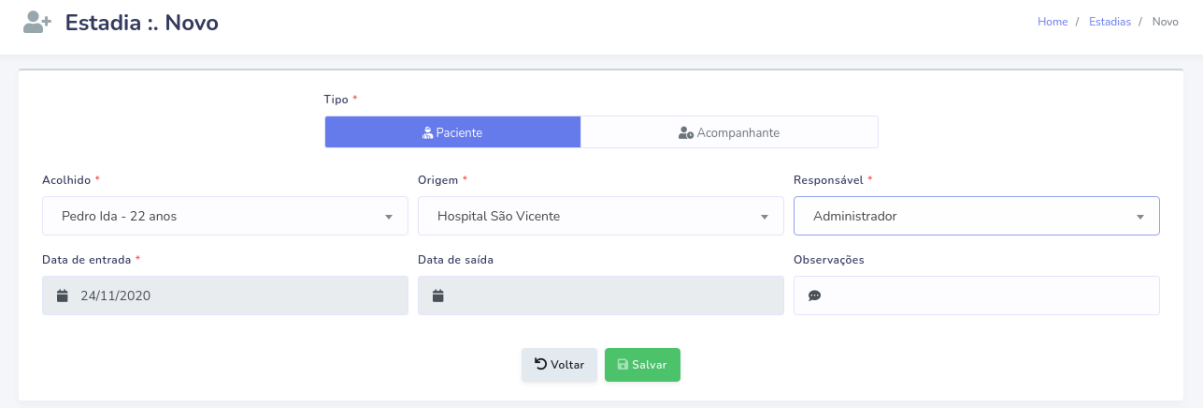


Figura 17 – Criação de nova estadia.

A área de refeições é um pouco mais simples, não dependendo de uma lista convencional assim como nos outros recursos do sistema. Portanto foi utilizado o modelo de cartões na listagem para trazer uma interface mais amigável nesta tela, como pode ser visto na Figura 18, já que não possuem muitos dados relacionados ao recurso de refeição.

Figura 18 – Lista de refeições diárias.

A criação de um novo dia de refeições também foi elaborada de modo diferente aos outros recursos do sistema. Para deixar mais dinâmico e rápido o sistema, foi utilizado um modal que contém um calendário para selecionar a data da refeição e os campos referentes aos tipos de refeições disponíveis (Figura 19). A criação ou edição de refeições foram feitas via ajax para evitar recarregamento de página desnecessário e garantir fluidez na utilização do recurso.

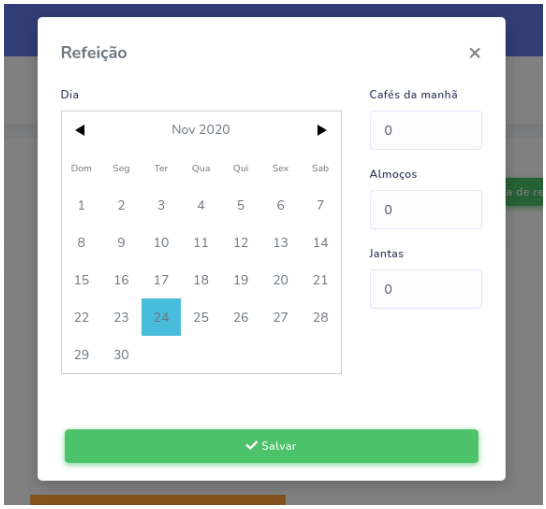


Figura 19 – Criação de nova refeição diária.

A *dashboard* foi o último recurso a ser desenvolvido por conta da necessidade de implementação de todos os outros recursos para que pudesse ser populado e utilizado. Como base de filtro, foi inserido uma escolha por mês ou ano para a busca dos dados. As informações mostradas nesta tela possuem planejamento para que sejam agregadas novas informações e novos modelos de filtros conforme a necessidade do sistema.

Os dados apresentados nesta versão possuem:

- Um contador de refeições cedidas no período com um total agregado;
- O total de novas estadias recebidas no período selecionado;

- Um gráfico para melhor ilustração do contador de estadias;
- Uma lista das últimas doações a serem inseridas ou atualizadas;

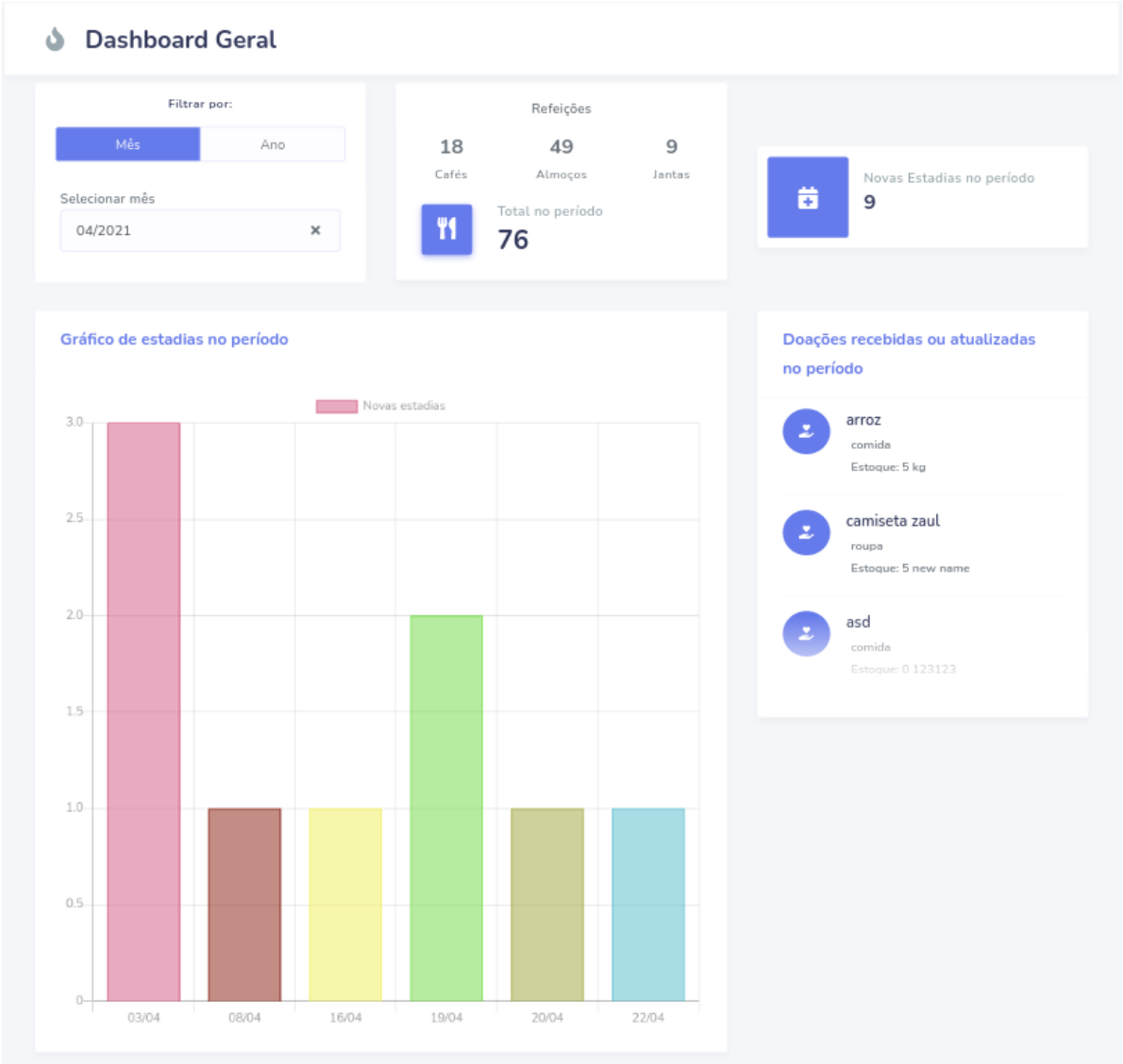


Figura 20 – Dashboard.

4 ANÁLISE E DISCUSSÃO DOS RESULTADOS

Neste capítulo serão apresentados os resultados, dificuldades e problemas que foram encontrados no desenvolvimento do projeto.

4.1 REALIZAÇÃO DA COLETA DE REQUISITOS

Na realização da visita ao local, foi possível visualizar o meio no qual o sistema deve funcionar e qual é o padrão de usuário que terá acesso. A experiência que foi ganha tendo um contato dessa maneira para obter os requisitos do sistema foi um desafio, tendo em vista que essa tarefa requer um olhar clínico para que seja possível abstrair o máximo de informações sobre o processo como um todo e não é uma tarefa que é feita corriqueiramente.

4.2 PRINCIPAIS DADOS APÓS COLETA

Após a coleta dos requisitos, foi possível desenhar o Diagrama de Entidade Relacionamento que atende as necessidades de forma simples e direta, sem levar em consideração as mudanças futuras que podem ocorrer no projeto, utilizando assim alguns conhecimentos que foram adquiridos nas aulas de Análise e Projeto Orientados a Objetos no curso.

Com base nas informações coletadas e a estruturação do banco de dados, foram desenhadas. Pensando na melhor experiência para os usuários, as principais telas que seriam utilizadas, a listagem de estadias, o cadastro de uma nova estadia de um acolhido e o controle de refeições diárias. Foi percebida a necessidade de complementar as caixas de textos com ícones para a melhor associação e assimilação dos campos por parte dos usuários do sistema para acelerar o processo de aprendizagem. Foi aplicado o conceito da matéria de Interação Humano-Computador de *affordance* nos menus e formulários, que consiste em utilizar de meios visuais para sugerir e indicar uma função por meio de conhecimentos comuns da população. Isso foi aplicado para a trazer a melhor forma de interação de usuário com o sistema, como pode ser visto nas Figuras 21, 22 e 23.

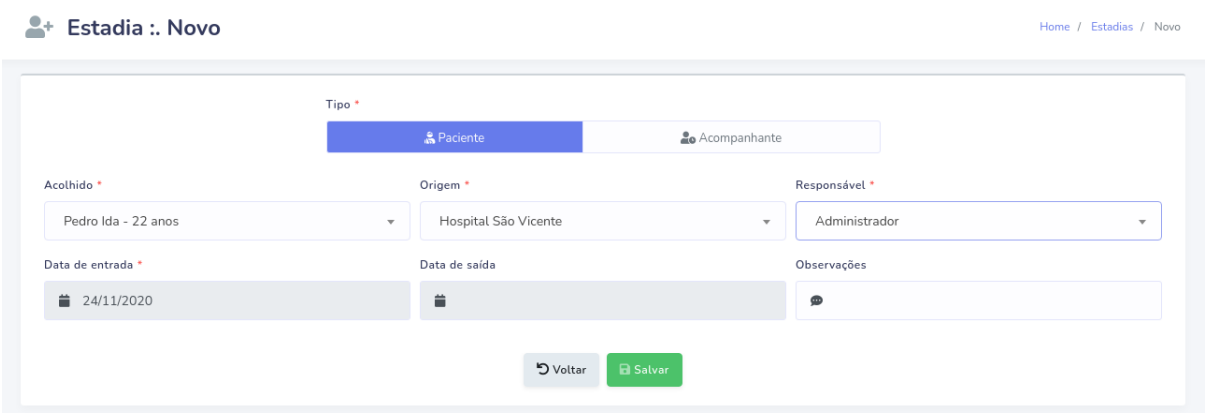


Figura 21 – Criação de nova estadia.

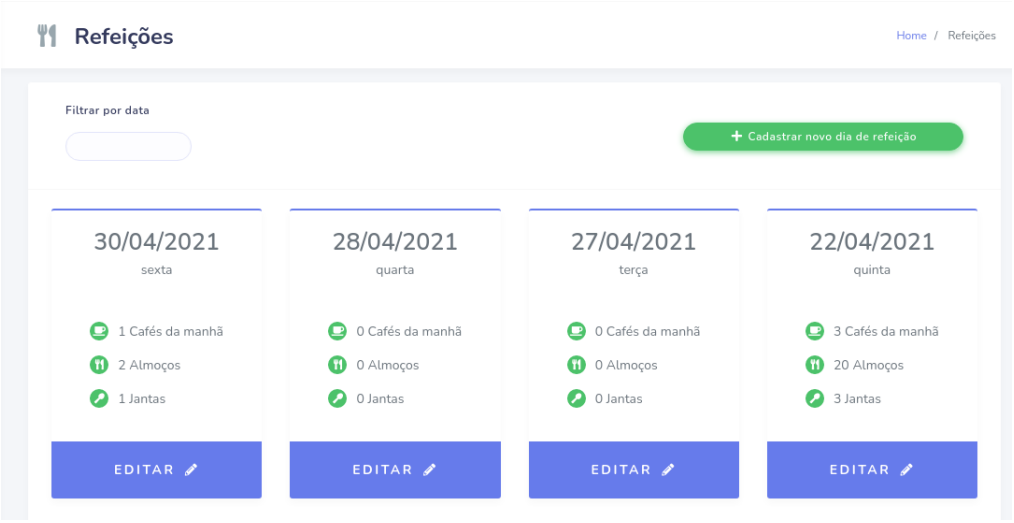


Figura 22 – Lista de refeições diárias.

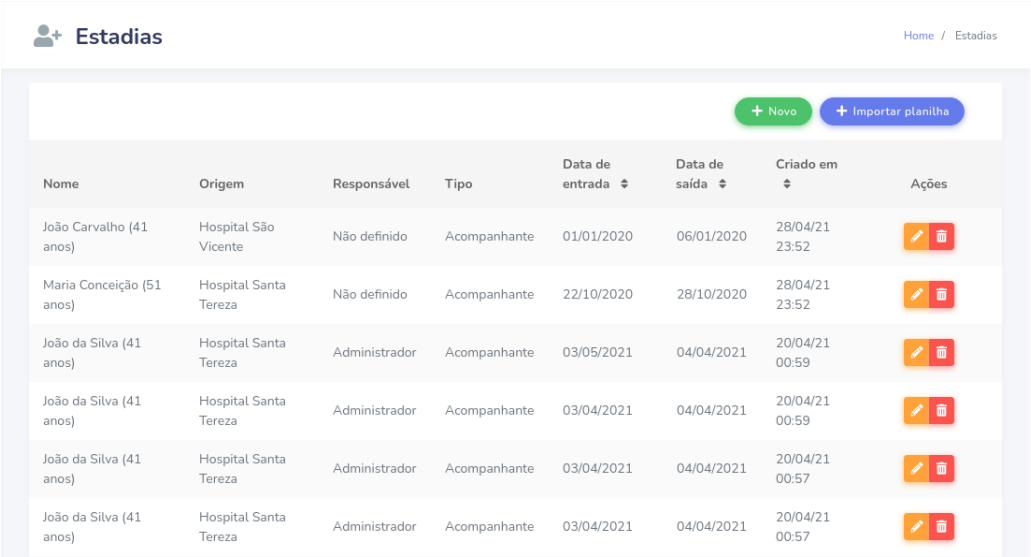


Figura 23 – Lista de estadias.

4.3 DESENVOLVIMENTO DE TESTES

Após o desenvolvimento das funcionalidades do sistema, foram desenvolvidos os testes automatizados para garantia de estabilidade. O foco dos testes foi centralizado em garantir a cobertura das entidades e controladores que gerenciavam as requisições. No total foram escritos 145 testes para as funções do sistema como pode ser visto na Figura 24.

```
PHPUnit 8.5.15 by Sebastian Bergmann and contributors.

...............................................................  63 / 145 ( 43%)
...............................................................  126 / 145 ( 86%)
...................                                              145 / 145 (100%)
```

Figura 24 – Total de testes.

Com esse número de testes foi possível chegar a 100% de cobertura de testes nas entidades. Cerca de 93% de testes nos recursos referentes a requisições do sistema, devido a linhas no código que não são executadas diretamente nas requisições, fez com que não fosse possível atingir um número maior. Os *middlewares* tiveram uma cobertura baixa por alguns serem fixos do *framework* e não não propriamente inseridos nas requisições. Os resultados são visíveis nas Figuras 25 e 26.

	Code Coverage								
	Lines			Functions and Methods			Classes and Traits		
Total		100.00%	31 / 31		100.00%	23 / 23		100.00%	10 / 10
Category.php		100.00%	2 / 2		100.00%	2 / 2		100.00%	1 / 1
City.php		100.00%	2 / 2		100.00%	2 / 2		100.00%	1 / 1
Client.php		100.00%	6 / 6		100.00%	4 / 4		100.00%	1 / 1
Donation.php		100.00%	6 / 6		100.00%	4 / 4		100.00%	1 / 1
Meal.php		100.00%	4 / 4		100.00%	1 / 1		100.00%	1 / 1
Source.php		100.00%	1 / 1		100.00%	1 / 1		100.00%	1 / 1
State.php		100.00%	3 / 3		100.00%	3 / 3		100.00%	1 / 1
Stay.php		100.00%	3 / 3		100.00%	3 / 3		100.00%	1 / 1
Unit.php		100.00%	2 / 2		100.00%	2 / 2		100.00%	1 / 1
User.php		100.00%	2 / 2		100.00%	1 / 1		100.00%	1 / 1

Figura 25 – Cobertura de entidades.

	Code Coverage								
	Lines			Functions and Methods			Classes and Traits		
Total		92.70%	673 / 726		84.94%	141 / 166		62.00%	31 / 50
Controllers		91.28%	429 / 470		83.87%	104 / 124		31.82%	7 / 22
Middleware		47.37%	9 / 19		25.00%	1 / 4		0.00%	0 / 3
Requests		97.10%	67 / 69		92.31%	24 / 26		92.31%	12 / 13
Resources		100.00%	168 / 168		100.00%	12 / 12		100.00%	12 / 12
Kernel.php		n/a	0 / 0		n/a	0 / 0		n/a	0 / 0

Figura 26 – Cobertura de requisições.

4.4 IMPLANTAÇÃO

Devido a pandemia do novo coronavírus OPAS (2021), não foi possível realizar a implantação do sistema na casa de acolhimento, tampouco realizar a validação com os usuários. Portanto a implantação ficará para um momento posterior à pandemia.

Todo o processo de implantação e coleta de *feedbacks* será feito em um momento posterior a atual pandemia para garantir a continuidade do projeto.

5 CONCLUSÃO

O objetivo deste trabalho foi desenvolver uma solução simples e de fácil utilização para o gerenciamento da Casa de Acolhimento Nazaré, um sistema que pode ser utilizado por pessoas que não possuem tanto conhecimento com tecnologia. Nesta solução foi implementado um sistema *web*, ou seja, que é acessível tanto através de um computador quanto de um *smartphone*. O sistema possui três funções primordiais: realizar o controle de acolhimentos e acolhidos, controle de refeições cedidas diariamente e controle de doações recebidas. A casa de acolhimento em questão possuía muitos registros de acolhimentos, realizados desde sua abertura, sendo mantidos apenas em cadernos de anotações, tornando praticamente inviável uma busca por alguma informação e até um risco para segurança dos dados. Com esse projeto, foi possível entender melhor como funciona a coleta de requisitos no momento de iniciar o desenvolvimento de um software, a entender melhor o lado do usuário que vai utilizá-lo, podendo assim assimilar as ideias e propor soluções que facilitem os processos.

5.1 TRABALHOS FUTUROS

O sistema possui algumas melhorias que foram colocadas como opcionais e com baixa prioridade de desenvolvimento pelos integrantes da instituição favorecida. Uma nova função do sistema seria o controle de quartos disponíveis no recinto, já que no momento não há a necessidade de tal função, porém a longo prazo poderá ser necessária caso o número de acolhidos aumente. Outra função que poderá ser necessária, com o passar do tempo é o controle de horário de disponibilidade de um quarto, já que um acolhido pode sair, por exemplo, antes do almoço e já liberar uma nova vaga a partir deste horário.

Referências

CODINGINFINITE. **CodingInfinite**. 2020. Disponível em: <https://codinginfinite.com/top-programming-languages-2020-stats-surveys/>. Acesso em: 16 de novembro de 2020. Citado na página 10.

DATE, C. **Introdução a Sistemas de Bancos de Dados**. 1a edição. ed. [S.l.]: LTC, 2004. Citado na página 12.

DESENVOLVIMENTOAGIL. **DesenvolvimentoAgil**. 2020. Disponível em: <https://www.desenvolvimentoagil.com.br/scrum/>. Acesso em: 17 de novembro de 2020. Citado na página 6.

EASY2BECODER. **easy2becoder**. 2020. Disponível em: <http://easy2becoder.blogspot.com/2017/04/top-5-php-frameworks-for-developers.html>. Acesso em: 16 de novembro de 2020. Citado na página 11.

HOMEHOST. **homehost**. 2020. Disponível em: <https://www.homehost.com.br/blog/tutoriais/o-que-e-bootstrap/>. Acesso em: 21 de novembro de 2020. Citado na página 7.

HOSTINGER. **O que é Apache? Uma Visão Aprofundada do Servidor Apache**. 2020. Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-e-apache>. Acesso em: 21 de novembro de 2020. Citado na página 9.

HQBEDS. **HQBeds**. 2020. Disponível em: <https://hqbeds.com/>. Acesso em: 10 de setembro de 2020. Citado 3 vezes nas páginas 4, 5 e 6.

IDES. **IDES**. 2020. Disponível em: <https://www.ides-sc.org.br/>. Acesso em: 10 de setembro de 2020. Citado 3 vezes nas páginas 3, 4 e 6.

MARINHO, F. **FREDERICO MARINHO**. 2020. Disponível em: <https://www.fredericomarinho.com/introducao-ao-desenvolvimento-web-com-php-aula-1-preparando-o-ambiente-para-in cliente-servidor-php/>. Acesso em: 16 de novembro de 2020. Citado na página 13.

MOZILLA. **O que é um servidor web (web server)?** 2020. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Learn/Common_questions/What_is_a_web_server>. Acesso em: 8 de abril de 2021. Citado na página 8.

OPAS. **Covid-19**. 2021. Disponível em: <https://www.paho.org/pt/covid19>. Acesso em: 24 de maio de 2021. Citado na página 28.

PHP. **Manual PHP**. 2020. Disponível em: <https://www.php.net/manual/pt_BR/intro-whatis.php>. Acesso em: 21 de novembro de 2020. Citado na página 9.

POSTGRESQL. **PostgreSQL**. 2020. Disponível em: <https://www.postgresql.org/>. Acesso em: 27 de novembro de 2020. Citado na página 17.

SCRUM. **Scrum**. 2020. Disponível em: <https://www.holmesdoc.com.br/scrum-o-que-como-funciona-e-como-aplicar/>. Acesso em: 22 de maio de 2021. Citado na página 6.

SELECT2. **Select2**. 2020. Disponível em: <https://select2.org/>. Acesso em: 27 de novembro de 2020. Citado na página 22.

SISHOSP. **SisHOSP**. 2020. Disponível em: <https://sishosp.com.br/software-de-gestao-para-casa-de-repouso/>. Acesso em: 24 de novembro de 2020. Citado 2 vezes nas páginas 5 e 6.

STISLA. **Stisla**. 2020. Disponível em: <https://getstisla.com/blog/open-source>. Acesso em: 27 de novembro de 2020. Citado na página 20.

TECNICON. **Tecnicon**. 2020. Disponível em: <https://www.tecnicon.com.br/blog/411-Metodologia_Scrum_para_a_gestao_de_processos_ageis_na_industria>. Acesso em: 16 de novembro de 2020. Citado na página 7.

VRSYS. **vrsys**. 2020. Disponível em: <https://www.vrsys.com.br/blog/site-estatico-ou-site-dinamico>. Acesso em: 16 de novembro de 2020. Citado 2 vezes nas páginas 8 e 9.

VUEJS. **VueJs**. 2020. Disponível em: <https://br.vuejs.org/index.html>. Acesso em: 22 de maio de 2021. Citado na página 11.

VUEJSDATEPICKER. **VuejsDatepicker**. 2020. Disponível em: <https://github.com/charliekassel/vuejs-datepicker>. Acesso em: 27 de novembro de 2020. Citado na página 21.