
Curso de Ciência da Computação
Universidade Estadual de Mato Grosso do Sul

Desenvolvimento de Software para Gestão de Organizações Sem Fins Lucrativos

Alvaro Leonel Socorro Queiroz Mariano
Calebe Garcia Paes

Msc. Jéssica Bassani de Oliveira
(Orientadora)

Dourados– MS
2017

Desenvolvimento de Software para Gestão de Organizações Sem Fins Lucrativos

Alvaro Leonel Socorro Queiroz Mariano
Calebe Garcia Paes

Este exemplar corresponde à redação final da monografia da disciplina Projeto Final de Curso, como parte dos requisitos para a obtenção do título de Bacharel em Ciência da Computação.

Dourados, 9 de Novembro de 2017

Prof. Msc. Jéssica Bassani de Oliveira
(Orientadora)

Curso de Ciência da Computação
Universidade Estadual de Mato Grosso do Sul

Desenvolvimento de Software para Gestão de Organizações Sem Fins Lucrativos

Alvaro Leonel Socorro Queiroz Mariano
Calebe Garcia Paes

Setembro de 2017

Banca Examinadora:

Prof. MSc. Jéssica Bassani de Oliveira (Orientadora)

Área de Computação - UEMS

Prof. MSc. André Chastel Lima

Área de Computação - UEMS

Prof. Dr. Cleber Valgas Gomes Mira

Área de Computação - UEMS

Dedico este trabalho primeiramente aos meus pais, os quais sempre me apoiaram firmemente em todos os momentos. Dedico também a minha irmã, a responsável por não me deixar perder a oportunidade de ingressar nesta universidade e me apresentar novos horizontes. Com grande importância na reta final deste desafio, também agradeço a minha esposa, responsável por repor minhas esperanças quando elas estavam quase se esgotando. Também não posso deixar de agradecer a todos que passaram pela minha vida durante essa etapa, as pessoas que conheci, e principalmente os amigos que levarei para o resto da vida.

Alvaro Leonel Socorro Queiroz Mariano

Dedico este trabalho primeiramente a Deus, por ter me dado a vida. Aos meus pais, Lucia e Garão, que não mediram esforços para a realização de meus objetivos. Dedico também à minha avó Maria Ribeiro, mesmo não entendendo os meus sonhos, sempre me apoiou. A todos os meus amigos e colegas que conheci, sendo a minha família em Dourados. E por último, mas não menos importante, ao Ministério Agnus Dei e ao Ministério Universidades Renovadas, onde fiz muitos amigos e que nas minhas fraquezas, estavam sempre a postos a me ajudar a levantar.

Calebe Garcia Paes

AGRADECIMENTOS

As nossas famílias, que nos apoiaram de forma incondicional durante toda essa jornada.

A nossa orientadora Jéssica Bassani de Oliveira, pela paciência e dedicação ao dar as orientações necessárias para a conclusão deste trabalho.

A todos os professores que transmitiram seus conhecimentos e ensinamentos, durante a graduação.

A todos os amigos e colegas que vivenciaram essa jornada conosco, nas trocas de conhecimentos, e deixando momentos memoráveis que jamais serão esquecidos.

O nosso muito obrigado a todos.

Alvaro Leonel Socorro Queiroz Mariano e Calebe Garcia Paes

RESUMO

O objetivo deste trabalho é criar um protótipo para ajudar na gestão de organizações sem fins lucrativos. Estas instituições do Terceiro Setor, que tem como objetivo ajudar a sociedade e pessoas vulneráveis, sendo assim, não visam o lucro. Tendo como apoio, para as realizações de suas ações e projetos, os governos em todas as instâncias e voluntários, o dinheiro recebido é todo utilizado para as necessidades básicas da instituição, e dessa forma, as organizações não tem condições de adquirir *softwares* pagos. Com este intuito, o protótipo do projeto foi desenvolvido, possibilitando as organizações utilizarem na gestão de suas atividades. Foi desenvolvido na linguagem *Python*, utilizando o *framework*, *Django*, e o banco de dados *PostgreSQL*. Presume-se que o sistema venha solucionar a problemática apresentada e atendendo os objetivos. Ainda, como sugestão para futuros trabalhos, pode-se desenvolver novos módulos, atendendo as demais necessidades.

Palavras-chave: Protótipo. Terceiro Setor. Organização sem fins lucrativos. ERP. Django. Angular.

ABSTRACT

The purpose of this paper is to create a prototype to assist in the management of nonprofit organizations. These institutions of the Third Sector, which aims to help society and vulnerable people do not aim for profit. These institutions of the Third Sector, which aims to help society and vulnerable people, and thus, do not aim for profit. Having the support for the accomplishments of their actions and projects, governments in all instances and volunteers, the money received is all used for the basic needs of the institution, in this way, the organizations cannot afford paid software. With this purpose, the prototype of the project was developed, allowing the organizations to use in the management of their activities. It was developed in Python, using the framework Django and having PostgreSQL as a database. Presumed that the system solves the problem presented and meeting the objectives. Also, as a suggestion for future work, it is possible to develop new modules, meeting other needs.

Key words: Prototype. Third sector. Nonprofit organization. ERP. Django. Angular.

SUMÁRIO

	Lista de abreviaturas e siglas	xv
	Lista de ilustrações	xvii
	Lista de tabelas	xix
1	INTRODUÇÃO	1
1.1	Objetivo	2
1.1.1	Objetivos Específicos	2
1.2	Justificativa	2
1.3	Organização do Trabalho	2
2	FUNDAMENTAÇÃO TEÓRICA	3
2.1	Terceiro Setor	3
2.2	Gestão de Entidades sem Fins Lucrativos	4
2.3	Sistemas de Informação na Gestão	5
2.4	WebApp	6
2.5	Experiência Falicitada para o Usuário	6
2.6	Ferramentas e Técnicas Utilizadas	7
2.6.1	Python	7
2.6.2	Django	8
2.6.3	API Rest	8
2.6.4	ECMAScript	10
2.6.5	Docker	10
2.6.6	PostgreSQL	11
2.6.7	Angular	11
2.7	Considerações Finais do Capítulo	12
3	TRABALHOS CORRELATOS	13
3.1	Economato	13
3.2	ONGFácil	14
3.3	HYB	16
3.4	Considerações Finais do Capítulo	17
4	DOCUMENTAÇÃO	19
4.1	Organizações Visitadas	19
4.1.1	Casa da Esperança	19

4.1.2	Associação Pestalozzi de Dourados	19
4.1.3	APAE	20
4.1.4	Lar do Idoso de Dourados	20
4.2	Especificação dos Requisitos do Sistema	21
4.2.1	Requisitos Funcionais	21
4.2.2	Requisitos Não Funcionais	22
4.3	Visão dos casos de Uso	22
4.3.1	Módulos	23
4.3.2	Diagrama de Atividade	24
4.3.3	Diagrama Entidade Relacionamento	25
4.4	Considerações Finais do Capítulo	25
5	DESENVOLVIMENTO	27
5.1	Camada de Dados	27
5.2	Camada de Aplicação	28
5.3	Camada de Apresentação	30
5.3.1	Capturas de Tela	35
5.4	Considerações Finais do Capítulo	39
6	RESULTADOS	41
6.1	Resultados: Identificação Básica	41
6.2	Resultados: Conhecimento do Protótipo	42
6.3	Considerações Finais do Capítulo	43
7	CONCLUSÃO	45
7.1	Melhorias Futuras	45
	REFERÊNCIAS	47
	APÊNDICES	49
	APÊNDICE A – QUESTIONÁRIO	50
	APÊNDICE B – RESPOSTAS DO QUESTIONÁRIO	52

LISTA DE ABREVIATURAS E SIGLAS

APAE	Associação de Pais e Amigos dos Excepcionais
API	Application Programming Interface
CRUD	Create, Read, Update e Delete
CSV	Comma-separated values
CTES	Centro de Estudos do Terceiro Setor
DRF	Django Rest Framework
ERP	Enterprise Resource Planning
IBGE	Instituto Brasileiro de Geografia e Estatística
JHU	John Hopkins University
JSON	JavaScript Object Notation
HTTP	HyperText Transfer Protocol
MVC	Model-view-controller
ONG	Organização Não Governamental
ORM	Object-Relational Mapping
PIB	Produto Interno Bruto
REST	Representational State Transfer
RF	Requisito Funcional
RNF	Requisito Não Funcional
SPA	Single Page Applications
SQL	Structured Query Language
TCP	Transmission Control Protocol
UML	Unified Modeling Language

LISTA DE ILUSTRAÇÕES

Figura 1 – Métodos comuns do HTTP	9
Figura 2 – Arquitetura REST	9
Figura 3 – Diferenciação entre virtualização e containers	11
Figura 4 – Captura de tela do vídeo de apresentação do software	13
Figura 5 – Módulos desenvolvidos	15
Figura 6 – Módulos desenvolvidos	15
Figura 7 – Valores dos planos mensais	16
Figura 8 – Integração financeira do software	17
Figura 9 – Diagrama do modelo de caso de uso	23
Figura 10 – Módulos para o desenvolvimento do protótipo	23
Figura 11 – Diagrama de atividades gerenciamento de contas bancárias	24
Figura 12 – Diagrama entidade relacionamento	25
Figura 13 – Estrutura de camadas	27
Figura 14 – Trecho de código do modelo de dados das contas bancárias	28
Figura 15 – Trecho de código da classe de serialização do modelo de contas bancárias	29
Figura 16 – Trecho de código das classes de visão das contas bancárias	29
Figura 17 – Endpoints da API Django/DRF dentro do ApiService	31
Figura 18 – Métodos de autenticação do ApiService	32
Figura 19 – Módulo Wallet	32
Figura 20 – Componente wallet-editor	33
Figura 21 – Componente wallet-list	34
Figura 22 – Componente wallet-viewer	34
Figura 23 – Captura de tela do protótipo: tela de login	35
Figura 24 – Captura de tela do protótipo: tela inicial	35
Figura 25 – Captura de tela do protótipo: listagem de contas bancárias	36
Figura 26 – Captura de tela do protótipo: listagem de transações de uma conta bancária	36
Figura 27 – Captura de tela do protótipo: tela de edição de uma categoria de transação	37
Figura 28 – Captura de tela do protótipo: listagem de transações	38
Figura 29 – Captura de tela do protótipo: listagem de colaboradores	38
Figura 30 – Tela	39
Figura 31 – Resultados: Função dos colaboradores na organização	41
Figura 32 – Resultados: Avaliação final do protótipo	42

LISTA DE TABELAS

Tabela 1 – Respostas dos questionários	52
--	----

1 INTRODUÇÃO

Este trabalho propõe o desenvolvimento de um *software* de gestão para organizações sem fins lucrativos, com o objetivo de facilitar e melhorar o trabalho dentro destas entidades, englobando desde área financeira e institucional até a social. Com o avanço das tecnologias, a gestão de uma instituição pode ser facilitada em muitos aspectos. Mas por falta de conhecimento e investimentos, muitas destas instituições não têm acesso facilitado ao uso de tecnologias de gestão que contribuiria para sua administração e desempenho.

As organizações sem fins lucrativos tem como principal objetivo ajudar a sociedade onde ela apresenta vulnerabilidade e a atuação do Estado às vezes é insuficiente ou inexistente. O presente trabalho por razões de delimitação do tema, optou por organizações sem fins lucrativos que atendem pessoas com demandas socioeconômicas. Todas essas organizações realizam estes serviços sociais sem visar nenhum tipo de lucro. Para a realização desses projetos e ações, as organizações realizam convênios e projetos com os governos, instituições privadas e doações de particulares. Dessa forma, o dinheiro arrecadado é apenas para custear despesas e realizar investimentos que possibilitem o desenvolvimento dos trabalhos.

Com os os valores monetários adquiridos sendo suficiente apenas para as realizações de projetos e custos básicos, a parte organizacional dessas entidades não avançaram em meio a era digital, a maioria das entidades, ainda realizam as atividades através de métodos ultrapassados, utilizando-se de papel e caneta ou no máximo uma planilha eletrônica para a prática de sua gestão. Sem possibilidades de acesso às tecnologias de informação, por vezes vêm suas potencialidades reduzidas por entraves operacionais.

As organizações visitadas foram a Associação de Pais e Amigos dos Excepcionais (APAE), Pestalozzi, Lar dos Idosos de Dourados e Casa da Esperança, todas localizadas na cidade de Dourados. As duas primeiras instituições tem como objetivo atender crianças e adolescentes com necessidades especiais, oferecem serviços nas áreas de educação, social e saúde. O Lar dos Idosos abriga pessoas da terceira idade que estão em estado de vulnerabilidade, assim oferecendo atenção por período integral. E por fim a Casa da Esperança que atende dependentes químicos, através da internação e reabilitação assistida.

Diante desta necessidade, vê-se uma oportunidade de contribuir para o desenvolvimento dessas atividades, muitas vezes ligadas à coletividade, na busca do bem comunitário, ou no auxílio de necessidades urgentes.

1.1 Objetivo

O objetivo deste trabalho é desenvolver um protótipo para uma base de um sistema *Enterprise Resource Planning* (ERP) para organizações sem fins lucrativos da cidade de Dourados.

1.1.1 Objetivos Específicos

- a) Identificar necessidades básicas de gerenciamento de uma organização sem fins lucrativos.
- b) Definir prioridades de gestão de uma organização sem fins lucrativos.
- c) Desenvolver o protótipo de uma base *Open Source* para um sistema ERP.

1.2 Justificativa

Este trabalho justifica-se pela necessidade de um *software* para organizações sem fins lucrativos. Conhecendo melhor a realidade dessas organizações, e entendendo que geralmente elas não possuem orçamento para aquisição de *softwares*, e acabam por desenvolver suas atividades sem controle, o que afeta diretamente a gestão da organização.

1.3 Organização do Trabalho

A organização do trabalho é composto por 7 capítulos. O capítulo 1 apresenta a introdução e objetivos do trabalho. O capítulo 2 apresenta o referencial teórico utilizado neste projeto. No capítulo 3 apresentamos os trabalhos correlatos. Já o capítulo 4 apresenta a metodologia da ferramenta construída. No capítulo 5 é exposto a implementação da ferramenta bem como capturas das telas a fim de melhor demonstrar as interfaces. No capítulo 6 são apresentados os resultados. E, finalmente no capítulo 7 é discorrida a conclusão do projeto.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo serão introduzidos os conceitos teóricos utilizados para o desenvolvimento deste trabalho, além de expor quais ferramentas serão utilizadas para a implementação do protótipo.

2.1 Terceiro Setor

O termo *terceiro setor* é recente e, portanto, ainda é objeto de inúmeras discussões. Segundo a professora Ana Maria Domeneghetti da Pontifícia Universidade Católica de São Paulo (PUC-SP), o terceiro setor é definido como:

“Refere-se às organizações que cuidam de problemas ligados à educação, saúde, meio ambiente, assistência social, abuso de álcool e drogas, sindicatos, museus e etc., que cria capital social (um corpo de voluntários) e empregos.” (DOMENEGHETTI, 2001).

Os autores que se dispõem a estudar o assunto podem divergir sobre o conceito e a abrangência do termo, dando ênfase a um dos elementos ou características do setor, como abrangência, finalidade ou natureza jurídica. Como podemos observar o conceito elaborado por Maria da Glória Marcondes Gohn:

O terceiro setor é um tipo de *‘Frankenstein’*: grande, heterogêneo, construído de pedaços, desajeitado, com múltiplas facetas. É contraditório, pois inclui tanto entidades progressistas como conservadoras. Abrange programas e projetos sociais que objetivam tanto a emancipação dos setores populares e a construção de uma sociedade mais justa, igualitária, com justiça social, como programas meramente assistenciais, compensatórios, estruturados segundo ações estratégico-rationais, pautadas pela lógica do mercado. Um ponto em comum: todos falam em nome da cidadania. (...) O novo associativismo do terceiro setor tem estabelecido relações contraditórias com o ‘antigo’ associativismo advindo dos movimentos populares (na maioria urbanos) dos anos 70 e 80 (GOHN, 2001).

Como vemos, o Terceiro Setor está sendo associado a cidadania e é formado pela sociedade civil, diferindo-se do Estado e do mercado econômico, primeiro e segundo setor respectivamente, sendo um espaço de participação nas causas coletivas. Como a senhora Ruth Cardoso ex-primeira dama disse: “Terceiro Setor descreve um espaço de participação e experimentação de novos modos de pensar e agir.” (DOMENEGHETTI, 2001, p. 20).

Não tão evidente como os demais setores, o Terceiro Setor acaba por ser subestimado quanto a sua capacidade de resultados, uma amostra é que há poucas estatísticas sobre esse setor. “Portanto, por falta de informações consistentes, o Terceiro Setor como ideologia

não vem recebendo atenção à altura de sua importância, sendo utilizado como cortina de fumaça para encobrir preocupações políticas e ideológicas.” (DOMENEGHETTI, 2001).

Atualmente, apenas 12 países consideram o Terceiro Setor em suas estatísticas, incluindo Brasil, que desde 2004, passou a considerar separadamente o Terceiro Setor em suas estatísticas das contas nacionais.

O Centro de Estudos do Terceiro Setor (CETS) tornou-se um dos protagonistas do processo para convencer o Instituto Brasileiro de Geografia e Estatística (IBGE) a adotar a nova metodologia. Em setembro de 2004, o CETS, em parceria com o Centro de Estudos da Sociedade Civil da Universidade Johns Hopkins, organizou uma reunião em São Paulo com lideranças dos principais centros de estudos brasileiros, que resultou em uma manifestação coletiva, encaminhada ao IBGE, solicitando que fosse considerada a possibilidade de incluir nas contas nacionais a participação das organizações sociais (MEREGE, 2007).

O IBGE desenvolve desde 2006 uma pesquisa com as entidades de assistência social privadas sem fins lucrativos com objetivo de mapear estas entidades e suas áreas de atuação. Como também, identifica a participação do Terceiro Setor no Produto Interno Bruto (PIB), que representa cerca de 1,4%, ou seja, 32 bilhões de reais (IBGE, 2007).

O Terceiro Setor representa uma grande força na economia, pelo número de empregos que gera, os valores aplicados a ele, e suas despesas. A fonte de renda do Terceiro Setor é variável, e não necessariamente apenas fruto de caridade de pessoas físicas. E pode advir também de repasses das três esferas estatais, como Governo Federal, Estadual ou Municipal, ou ainda de empresas privadas (LUCA et al., 2008).

(DOMENEGHETTI, 2001) ressalta que no Brasil, as instituições religiosas, como os centros de umbanda, candomblé, centros espíritas e igrejas em geral, criam e mantêm o maior número de organizações do Terceiro Setor.

2.2 Gestão de Entidades sem Fins Lucrativos

O crescimento das atividades do Terceiro Setor pressupõe cada dia mais, uma atuação profissionalizada, considerando conceitos de eficácia, eficiência, e qualidade nos serviços. É importante ressaltar, que entidades sem fins lucrativos, nem sempre se adaptam bem a ferramentas tradicionais da administração, que foram criadas orientadas ao mercado e que fundamentalmente visam lucros.

A Organização Não Governamental (ONG) tem o intuito de realizar atividades que auxiliam a sociedade, por meio de seus trabalhos para o bem da comunidade sem fins lucrativos. As ONGs desempenham um papel social importante perante a sociedade (LIMA; SILVA, 2013).

De forma a suprir as necessidades da sociedade, as ONGs por meio de seu voluntariado, conseguem realizar atividades em que o governo não atua de uma forma satisfatória ou até suprir a falta desse serviço. A área de atuação é muito extensa: social, saúde, tratamentos de dependentes químicos, idosos, educação, entre outros (FRANCISCO, 2010).

Através das pesquisas que foram realizadas pelo *Instituto Studies* da *John Hopkins University* (JHU), o instituto criou cinco critérios para que uma organização seja considerada do terceiro setor: formalização, natureza privada, não distribuição de lucros, autogestão e participação voluntária (FERREIRA, 2005).

2.3 Sistemas de Informação na Gestão

Um sistema de gerenciamento para uma empresa serve como um ponto de ligação, onde todos os setores conseguem se comunicar, atendendo a todos os setores da empresa.

A importância de uma informação não são os dados coletados, mas sim de que forma esses dados serão utilizados para o bem de sua gestão. (MCGEE; PRUSAK, 1994) entendem que para que os dados tenham um resultado satisfatório, os mesmos devem ser coletados, estudados e com os resultados, a decisões são colocadas em prática.

(MCGEE; PRUSAK, 1994) escrevem que “a informação é capaz de criar valor significativo para as organizações, possibilitando a criação de novos produtos e serviços e aperfeiçoando a qualidade do processo decisório”. Tendo as informações necessárias, o aperfeiçoamento de projetos obtêm um maior índice de resultados eficientes.

Seguindo esse raciocínio, (MCGEE; PRUSAK, 1994) afirmam que para um desenvolvimento de um gerenciamento de informações satisfatórias são realizados em quatro etapas:

- Identificar as necessidades e os requisitos, conhecer a sua fonte de origem e coletar as informações necessárias e entender o ambiente de onde estão sendo colhido os dados. Esta é considerada a etapa mais importante, pois é baseada nesta etapa que as próximas etapas são embasadas.
- Classificar e armazenar as informações, decidindo quais usuários terão acesso às informações e qual a melhor forma de se armazenar as informações adquiridas.
- Tratar e adaptar as informações, e esta etapa ocorre simultaneamente com a etapa anterior, onde são apresentadas a todos as informações de uma forma clara a todos, assim facilitando o entendimento de todos os usuários envolvidos.

- Desenvolver o produto, quando se é colocado em prática tudo que foi estudado e realizado para que o usuário final tenha a oportunidade de conhecer o projeto e usá-lo.

(DOMINGUES, 2014) afirma que com a implementação do sistema de gerenciamento, uma empresa consegue melhorar trabalhos internos e assim produzir rendimento em muitos aspectos, dentre eles: disponibilizar de uma forma mais facilitada as informações, proporcionando a geração de relatórios; melhoria na produtividade; melhor adaptação para problemas não esperados; projeções para tomadas de decisões mais seguras; melhoria na estrutura organizacional e maior interação entre os setores.

2.4 WebApp

Nos primórdios da informática, quando os programas de computadores ainda vinham em disquetes ou CDs, todos eram classificados como *softwares*. Uma nomenclatura que surgiu para separar os componentes de um computador, já conhecidos como *hardware*, dos programas escritos para fazê-los funcionar. Segundo Jorge Fernandes, uma definição clássica de *software* é:

"Software é uma sentença escrita em uma linguagem computável, para a qual existe uma máquina (computável) capaz de interpretá-la. A sentença (o software) é composta por uma sequência de instruções (comandos) e declarações de dados, armazenável em meio digital. Ao interpretar o software, a máquina computável é direcionada à realização de tarefas especificamente planejadas, para as quais o software foi projetado."
(FERNANDES, 2002)

Com o advento da *internet* e da popularização dos dispositivos móveis, os *softwares* foram obrigados a se aperfeiçoarem para conseguir suprir as novas necessidades. Com a explosão do crescimento de *smartphones*, popularizados pela chegada do *Iphone* e em seguida do *Android*, o *software* evoluiu para os *Apps*, que nada mais são do que *softwares* simples e de fácil utilização, amplamente distribuídos pelas *Apps Stores*, lojas *onlines* que facilitam o processo de venda e instalação destas aplicações (FAZFÁCIL, 2017).

2.5 Experiência Falcitada para o Usuário

Com o desenvolvimento da humanidade, o avanço tecnológico está cada vez mais presente em nossa sociedade. Durante todo o dia, estamos rodeados de tecnologias. E, o constante aperfeiçoamento, facilita o manuseio dos mais diversos aparelhos eletrônicos. As empresas que desenvolvem produtos na área de tecnologia fazem estudos aprofundados para que a experiência do usuário seja a mais satisfatória possível (ARAUJO et al., 2014).

As necessidades dos usuários evoluíram, o que torna o estudo do usuário importante. Pois, quanto mais se conhece o usuário, maior a possibilidade de desenvolvimento da tecnologia desejada para atender às necessidades do público alvo. No entanto, não basta somente estudar o usuário, é necessário estudar também o ambiente em que será utilizado, qual a melhor forma de usabilidade para aquela tecnologia e prever quais melhorias serão realizadas futuramente (ARAUJO et al., 2014).

“Se a experiência do usuário trata da relação entre as pessoas com os serviços digitais, está aí sua importância, pois a interação implica na escolha do serviço que uma pessoa irá utilizar, qual lhe oferecerá o melhor benefício ou possibilidades.” (LOPES, 2012)

O usuário sempre irá utilizar a tecnologia que melhor atende suas necessidades, que seja ao mesmo tempo de fácil usabilidade e lhe traga uma confiança para continuar utilizando.

A compreensão por parte do usuário relacionado ao ambiente utilizado se dá diante de um esfera pensada justamente para o fácil entendimento. Por isso, as tecnologias são desenvolvidas por uma equipe multidisciplinar, envolvendo diversos profissionais que vão além dos desenvolvedores de *software* ou sistemas, podendo comportar, por exemplo, analista de mídia, *webdesign* e *UX designer* (BRAGA et al., 2004).

(MORVILLE, 2004) estabelece sete características de forma eficiente e facilitada que visam atender as necessidades dos usuários: útil, utilizável, desejável, encontrável, acessível, credível e valioso.

2.6 Ferramentas e Técnicas Utilizadas

2.6.1 Python

Python é uma linguagem de programação orientada a objetos. O maior diferencial desta linguagem é que, ao executar o código fonte, cada linha é lida separadamente, pois utiliza-se um interpretador e não um compilador. Dessa forma, não é feita toda a leitura do código-fonte para a geração do código-objeto (LUTZ; ASCHER, 2007).

Com a linguagem *Python* pode-se criar diversos servidores *Web*, onde existe a possibilidade de utilizar *sockets*, realizando-se a conexão entre cliente servidor. Dessa forma, é possível receber e enviar e-mails e/ou realizar pesquisas em páginas da *Web* (LUTZ; ASCHER, 2007).

A linguagem *Python* é considerada de fácil aprendizagem, além de ser rápida e poderosa. Pode ser utilizada para fazer muitas tarefas ou apenas ser um componente do sistema independente e completo. A documentação e o seu interpretador podem ser

encontrados no site oficial (PYTHON SOFTWARE, 2017).

2.6.2 Django

Django é uma *framework* feito na linguagem *Python* desenvolvido em código aberto e voltado para o desenvolvimento de sistemas *Web*. Utilizando o padrão de arquitetura de *software* chamado *Model-view-controller* (MVC), boa parte das tarefas repetitivas do desenvolvimento das aplicações para a *internet* são facilitadas (FRAMEWORK, 2015).

Adrian Holovaty e Simon Willison criaram a primeira versão do *Django* com o objetivo de facilitar novas implementações e fazer modificações rápidas no sistema (HOLOVATY; KAPLAN-MOSS et al., 2007).

2.6.3 API Rest

O objetivo da *Application Programming Interface* (API) é facilitar o desenvolvimento de aplicações, através de conjuntos de anotações, classes e interfaces. Esta simplificação permite uma programação mais rápida e é considerada uma programação de alto nível (ROBERTO, 2015).

O REST (*Representational State Transfer*) é uma arquitetura apresentada por (FIELDING; TAYLOR, 2000) utilizada para sistemas de hipermídia distribuídas. Ele afirma que este método conta com os seguintes elementos: elementos de dados, onde basicamente são os recursos, os identificadores e suas representações; conectores que são os clientes, *caches*, servidores e *tunnel*; componentes, são os *gateways*, *proxy* e *user agents*, basicamente classificados como servidores de origem.

Para algumas ações, foram definidos métodos com palavras-chave, e, dessa forma facilitando a programação. Na figura 1 podemos observar os principais métodos.

Essa arquitetura funciona de uma maneira onde nem o cliente e nem o servidor armazenam o estado da conexão, e em cada mensagem HTTP está contida toda a informação necessária para compreensão do pedido. Como podemos visualizar na figura ??, esta característica permite a divisão da responsabilidades em dois ambientes, um, o servidor, que se preocupa somente com comunicação com banco de dados, gerenciamento de cache, *logs* e etc. E o outro, cliente, que tem o foco na interface e na experiência do usuário.

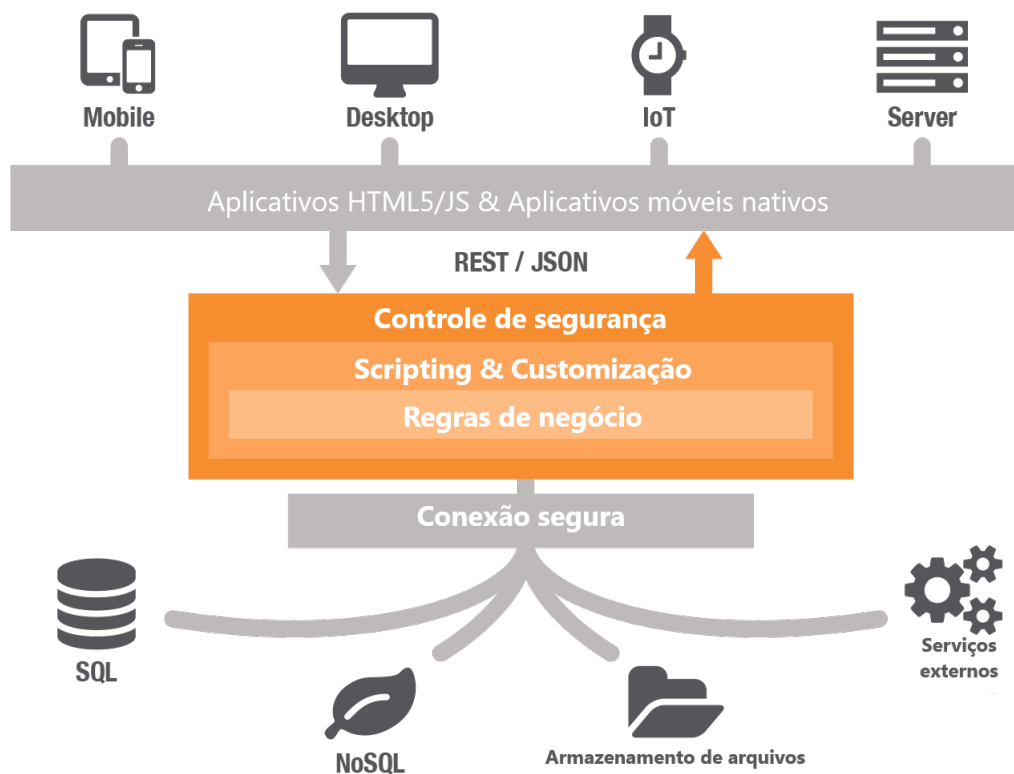
Figura 1 – Métodos comuns do HTTP

Método	Descrição
GET	Solicita que o servidor envie um recurso.
PUT	O inverso do GET. Realiza operações de escrita no servidor.
DELETE	Solicita a exclusão de um recurso no servidor.
POST	Foi projetado para envio de dados de input, como, por exemplo, os dados de um formulário para o servidor.
HEAD	Mesmo comportamento que o método GET, porém o servidor retorna apenas o cabeçalho da resposta.

Fonte: (FIELDING, 2015)

Com essa divisão de responsabilidades a evolução dos dois extremos da aplicação não ficam amarradas uma a outra, permitindo assim que varias plataformas possam ser atingidas em ritmos diferentes. Assim como pode ser vista na figura 2.

Figura 2 – Arquitetura REST



Fonte: (Elaborada pelos autores)

2.6.4 ECMAScript

A *ECMAScript* é uma linguagem padronizada pela empresa europeia *Ecma International* na especificação-262. É uma versão padrão do *Javascript*, onde qualquer empresa pode utilizar este recurso no desenvolvimento de seu *software* (NERY, 2017).

“*JavaScript* sempre incluirá funcionalidades que não são parte da especificação *ECMAScript*; *JavaScript* é compatível com *ECMAScript*, enquanto provém funcionalidades adicionais. ” (NERY, 2017)

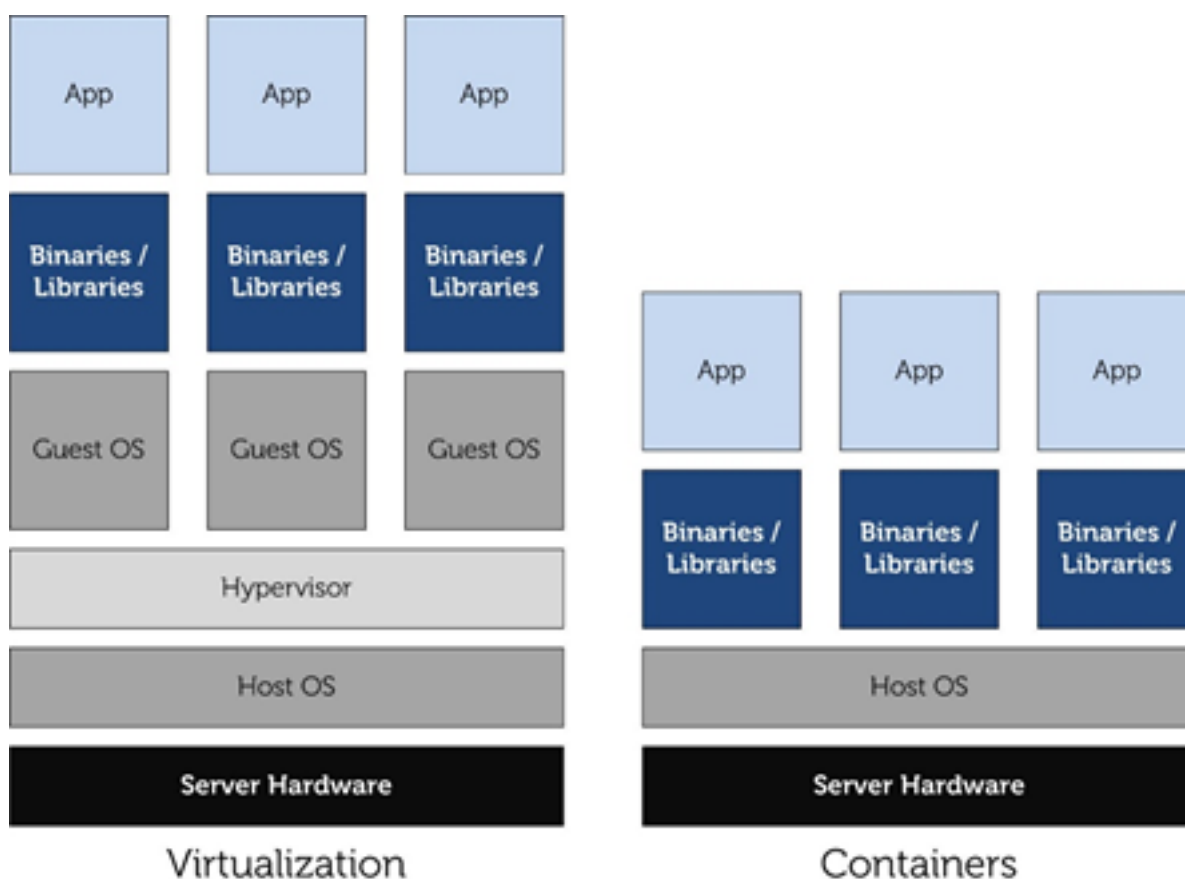
O *JavaScript* surgiu da necessidade de resolver as limitações de de interação com um usuário dentro de um site. No início ele apenas permitia a manipulação de códigos HTML agindo na melhoria da usabilidade. Hoje em dia com sua facilidade de aprendizado, grande tolerancia a falhas e aumento crescente da comunidade de usuários, ele é utilizado nas mais diversas plataformas, tais como dispositivos moveis, sistemas embarcados e até em servidores de aplicação (CAELUM.COM.BR, 2017a).

2.6.5 Docker

O *Docker* é uma plataforma que automatiza a implantação de aplicações em ambientes isolados. Tem como principal objetivo permitir a coexistência de vários ambientes isolados dentro de um mesmo servidor, mas que permita a comunicação entre os ambientes e entre os ambientes exteriores através de portas *Transmission Control Protocol* (TCP). É uma solução para desenvolvimento, execução e integração de uma forma rápida. O *Docker* está disponível em <https://www.docker.com/> (GOMES; SOUZA, 2015).

O *Docker* não é uma virtualização, consiste em um *container*, onde todo o seu ambiente está empacotado. Dessa forma, facilita a portabilidade para qualquer outro *Host* que também tenha o *Docker* instalado (DIEDRICH, 2015). Na figura 3 torna-se compreensível a diferença entre *Docker* e virtualização.

Figura 3 – Diferenciação entre virtualização e containers



Fonte: (DIEDRICH, 2017)

2.6.6 PostgreSQL

PostgreSQL é um banco de dados relacional criado com base no *postgres*, projeto conduzido pelo professor Michael Stonebraker, tendo sua primeira versão pública em 1988. O *PostgreSQL* vem sendo considerado a alguns anos como o banco de dados com código aberto evoluído (SOUCEFORGE, 2015).

O *postgres* teve como sua última versão 4.2, sendo um dos principais fatores para a finalização desse banco o grande aumento de usuários em 1993. Dessa forma, demandava-se muito tempo com o suporte, que deveria ser usado para a evolução do banco. Com isto, consideraram como a melhor solução, adicionar o interpretador de linguagem *Structured Query Language* (SQL). Já nesta primeira versão com o SQL, o banco apresentava um desempenho duas vezes melhor que a versão anterior (SOUCEFORGE, 2015).

2.6.7 Angular

O *Angular* é um *framework JavaScript* de código aberto. Criado pelo Google ele utiliza do padrão MVC para criar aplicações *Single Page Applications* (SPA). SPA's são

softwares da internet moderna, que se aproveitam de novos padrões e tecnologias para a criação de sistemas mais interativos, ágeis e fácil utilização. As SPA's permitem que paginas HTML se comporte como sistemas *desktop*, sem a necessidade recarregar a página inteira quando o usuário executa alguma ação.(CAELUM.COM.BR, 2017b)

Códigos *Angular* são escritos em *TypeScript*, que é um *superset* do *JavaScript*, e depois transpilados para o *JavaScript*. Combinando *templates* declarativos, injeção de dependência e ferramentas ponta a ponta, o *Angular* integra as melhores práticas para solucionar os desafios do desenvolvimento.(ANGULAR.IO, 2017)

2.7 Considerações Finais do Capítulo

Após conhecer o embasamento teórico para o desenvolvimento deste projeto e entender como se iniciaram as organizações sem fins lucrativos, possibilitou neste capítulo a definição das tecnologias utilizadas para o desenvolvimento do *software*, visando atender as demandas observadas.

3 TRABALHOS CORRELATOS

Neste capítulo são apresentados alguns *softwares* que são utilizados por organizações sem fins lucrativos. Existem outros *softwares* que são utilizados por essas instituições, porém, estes que são apresentados possuem semelhanças com o projeto que foi desenvolvido.

3.1 Economato

O *software* Economato produzido pela empresa bhbit, está presente no mercado há mais de 10 anos, oferecendo soluções em tecnologia para o terceiro setor e congregações religiosas de todo o Brasil.

Com o foco na parte financeira da organização, o Economato vem trazendo uma maior transparência e facilidade para as organizações, e dessa forma permitindo uma gestão facilitada. O *software* é de fácil manuseio, e com uma rápida aprendizagem. Disponível <https://www.economato.com.br>.

Figura 4 – Captura de tela do vídeo de apresentação do software

A captura de tela mostra a interface do sistema Economato. No topo, há uma barra de navegação com o nome 'ECONOMATO bhbit' e o perfil do usuário 'tatiane@bhbit.com.br'. O menu lateral à esquerda contém opções como 'VISÃO GERAL', 'FINANCEIRO', 'CADASTROS', 'PLANO DE CONTAS', 'RELATÓRIOS' e 'SUPORTE'. O conteúdo principal é dividido em seções:

- Visão geral**: O Painel do Economato oferece transparência e controle sobre os dados.
- Lançamentos em Atraso**: Tabela com as seguintes informações:

Data de Vencimento	Histórico	Valor
12/02/2016	Aluguel	R\$-3.000,00
12/02/2016	Material Didático	R\$-90,12
15/02/2016	IPTU	R\$-200,00
15/02/2016	Doação Rebeca Mendes (Apadrinhamento)	R\$50,00
15/02/2016	Doação Cláudia Ávila	R\$50,00
- Contas Bancárias - Saldo da Escrita**:

Banco Itaú BBA S.A. - Ag. 0001 - C. 12345 - CC	R\$14.735,13
Banco do Brasil S.A. - Ag. 0002 - C. 54321 - CDB	R\$620,00
Livro Caixa	R\$265,00
Saldo Total	R\$15.620,13
- Comparativo**: Opções para 'Mês Atual' e 'Mês Anterior'.

Um banner no canto inferior direito destaca: 'Melhoria na capacidade da equipe na identificação dos custos atuais e no monitoramento dos gastos'.

Fonte: (ECONOMATO, 2017)

Na figura 4, pode ser vista a tela inicial do programa. Ao lado esquerdo, encontra-se

quais funções o *software* realiza, que são os financeiros, cadastros, plano de contas, relatórios e suporte. Percebe-se que, o *software* mantém um melhor controle com o monitoramento de gastos.

O *software* conta com uma equipe técnica para o suporte, além de solucionar dúvidas dos usuários.

Como pode ser visto, o Economato trás uma melhor gestão para a parte financeira. Mas, infelizmente, não há um módulo para o gerenciamento de pessoas que usufruem dos projetos ali oferecidos. Dessa forma, apenas a parte financeira está sendo beneficiada com este *software*.

Outra grande desvantagem é que o *software* é pago, possui uma versão gratuita para experimentação por apenas 14 dias. Passado este período, assim como quase todos os *softwares* no meio empresarial e até mesmo no terceiro setor, há um custo para utilização do sistema. Este custo não condiz com a realidade das organizações de Dourados, pois as mesmas, não têm recursos para custear tais valores.

3.2 ONGFácil

O OngFácil é um *software* que atende tanto empresas como instituições do terceiro setor, desenvolvido pela Instituição Ekloos com parceria da empresa Easybus Software.

O *software* é desenvolvido em módulos, são divididos em 11 módulos, onde se encontram a parte de gerenciamento financeiro, controle de projetos, controle dos beneficiários, controle de usuários. É um software bem completo e organizado. Esta fórmula de ser separado por módulos facilita o gerenciamento. Nas figuras 5 e 6 é possível visualizar os módulos desenvolvidos no *software*.

Dos 11 módulos, apenas 4 são de livre acesso, ou seja, sem ônus. No entanto, apenas um usuário poderá utilizar tais módulos do sistema. Outra opção é o plano bronze, onde estão disponíveis 5 módulos e com registro de 1 usuário. O plano prata, estão disponíveis 9 módulos com o registro de até 6 usuários. E para o plano ouro estão disponíveis todos os 12 módulos e um registro de usuários ilimitados. Na figura 7 seguem os valores de cada plano.

Figura 5 – Módulos desenvolvidos



Fonte: (ONGFÁCIL, 2017)

Figura 6 – Módulos desenvolvidos



Fonte: (ONGFÁCIL, 2017)

Figura 7 – Valores dos planos mensais



	GRATUITO	BRONZE	PRATA	OURO
VALORES DE ASSINATURA MENSAL:	R\$ 00	R\$ 79	R\$ 210	R\$ 320
		/mês	/mês	/mês

Fonte: (ONGFÁCIL, 2017)

3.3 HYB

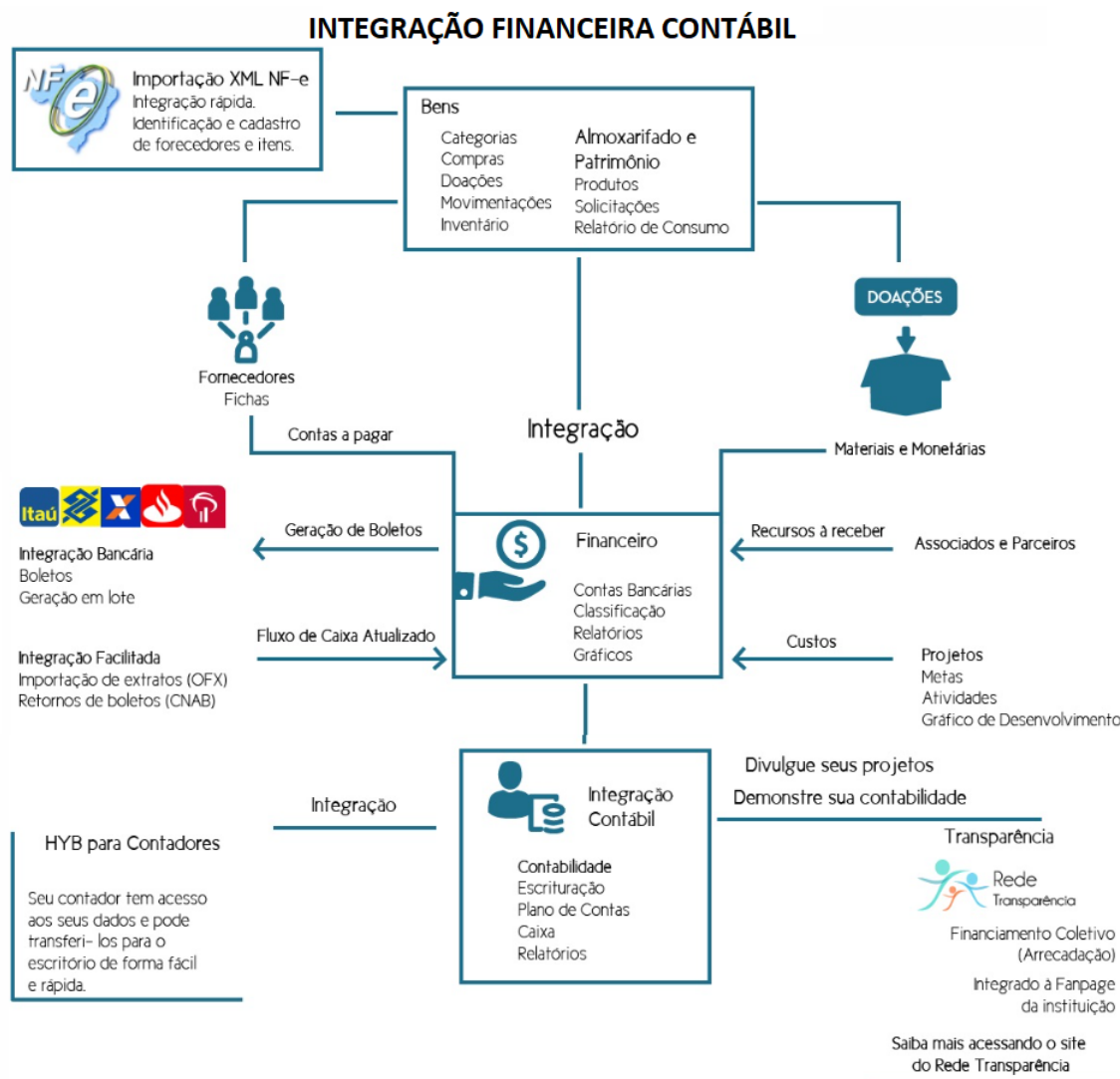
O HYB nasceu de um projeto aprovado entre a APAE da cidade de Giruá – RS, a Fundação Maurício Sirotsky Sobrinho, Souza Cruz, Rede Parceria Social e o Governo do Estado do Rio Grande do Sul. Foi desenvolvido pela empresa Conplan Sistemas de Informática *Ltda*.

O HYB é um software que tem como foco a transparência e uma melhor gestão. O sistema é dividido em 19 módulos, abrangendo a parte financeira, recursos humanos, beneficiários, voluntários, bens e patrimônios, educação, saúde, assistência social, entre outros. Como pode ser visto na figura 8.

O *software* é totalmente *online*, os dados ficam em servidores da própria empresa. O sistema ainda está em processo de melhorias e uma fácil mobilidade para acesso às informações.

O HYB tem um custo mensal considerável, mesmo sendo um ótimo *software*, muitas organizações não podem utilizá-lo por não haver recursos financeiros suficientes.

Figura 8 – Integração financeira do software



Fonte: (HYB, 2017)

3.4 Considerações Finais do Capítulo

Neste capítulo apresentou-se os *softwares* semelhantes a proposta deste projeto. Entre os sistemas pesquisados apresentamos os sistemas Economato, OngFácil e HYB. Assim possibilitou-se a contribuir com o projeto desenvolvido.

4 DOCUMENTAÇÃO

Neste capítulo, são apresentados os estudos e pesquisas para o desenvolvimento do projeto e a documentação deste trabalho baseados em padrões de projeto da *Unified Modeling Language* (UML). Nesta documentação, estão presentes as organizações visitadas, os requisitos funcionais e não funcionais, diagramas de casos de uso e suas descrições.

4.1 Organizações Visitadas

4.1.1 Casa da Esperança

A Casa da Esperança é uma comunidade de assistência e recuperação a dependentes químicos. É uma entidade assistencial terapêutica sem fins lucrativos, presente na cidade de Dourados a mais de 20 anos. São aceitos homens maiores de 18 anos que buscam o apoio para o tratamento de seu vício. Atualmente a casa tem a capacidade máxima de 20 dependentes.

A duração do tratamento é de nove meses, onde é neste período, através de terapia ocupacional, que consiste em: trabalho, disciplina, espiritualidade e autoconhecimento. A organização conta com o apoio financeiro dos próprios internados, além de convênio com o Governo Federal.

A Casa da Esperança tem como objetivo reabilitar esses dependentes para com o convívio da sociedade e a superação do vício de drogas muitas vezes ilícitas. A reintegração dos pacientes com a sociedade se inicia com fins de semana com seus familiares.

4.1.2 Associação Pestalozzi de Dourados

A Associação Pestalozzi de Dourados foi fundada em 1985 e atende de modo Assistencial, Educacional e Clínico. A entidade tem a capacidade de atender 130 usuários e realizar 18.000 procedimentos clínicos durante o ano.

Visando melhorar a qualidade de vida da pessoa com deficiência, a Pestalozzi é mantenedora da Escola Especializada Arco Íris, além de realizar atendimentos pedagógicos englobando todas as normas dos Conselhos Nacional, Estadual e Municipal.

A associação também disponibiliza serviço social e atendimento clínico aos alunos em contra turno da escola. São oferecidos procedimentos nas áreas de Psicologia, Fonoau-

diologia, Fisioterapia e Terapia Ocupacional, além de acompanhamento com neurologistas, oftalmologistas e dentistas.

Para atender esses 130 usuários, a entidade conta com mais de 30 funcionários, além dos estagiários e voluntários que se identificam com o projeto. A instituição conta com convênios municipais, estaduais e nacionais, além de doações, sócios contribuintes, e penas pecuniárias.

4.1.3 APAE

A APAE (Associação de Pais e Amigos dos Excepcionais) de Dourados atende crianças excepcionais há mais de 40 anos. São disponibilizados serviços no âmbito da saúde, da educação e da assistência social. Com este serviço, a organização busca a melhoria na qualidade de vida das crianças com deficiências, bem como conscientizar a população sobre a inclusão social.

Atualmente a instituição atende em média 250 crianças excepcionais, visando melhorar as capacidades mentais e motoras através de atividades realizadas dentro da instituição. Além dos projetos internos, há parcerias com as faculdades da cidade, por meio de projetos em diversas áreas, levando dessa forma a sociedade acadêmica mais perto dessa realidade.

A organização têm convênios com o Governo Federal, Estadual e Municipal, além de parcerias com empresas privadas. A APAE ainda recebe ajuda de doadores que se identificam com a missão da organização, os pais das crianças que frequentam o local e voluntários.

4.1.4 Lar do Idoso de Dourados

A organização acolhe idosos que estão na situação de vulnerabilidade há mais de 60 anos. Fundada pelo Rotary Club, a instituição tem como objetivo acolher idosos e prestar assistência biopsicossocioespiritual.

O Lar atende 45 idosos em regime de internato. Os mesmos contam com serviços especiais na área da saúde e da assistência social. São cuidados 24 horas por profissionais capacitados e o espaço é aberto para receber visitas e ajuda de voluntários.

Com base no Estatuto do Idoso, Lei nº 10.741, de 1º de Outubro de 2003, Capítulo VIII – Da Assistência Social, Art. 35, Parágrafos 1º e 2º, o idoso residente nesta Instituição contribui com 70% de qualquer benefício previdenciário ou de assistência social recebido. A instituição também tem parcerias com empresas e ajuda da sociedade com doações.

4.2 Especificação dos Requisitos do Sistema

Os requisitos definidos a seguir, delimitam o sistema baseado nas visitas realizadas nas organizações, onde foram realizadas entrevistas com os responsáveis e colaboradores das instituições.

4.2.1 Requisitos Funcionais

Os Requisitos funcionais são:

RF01 - O sistema deve cadastrar a organização com os seguintes dados: nome, endereço, telefone e CNPJ. (Requisitos Relacionados: —).

RF02 - O sistema deve permitir o gerenciamento do cadastro de grupos usuários com os seguintes dados: nome do grupo, permissões. (Requisitos Relacionados: RF1).

RF03 - O sistema deve cadastrar o responsável pela organização com os seguintes dados: nome completo, CPF, RG, data de nascimento, sexo, endereço, grupo de usuários, e-mail telefone e senha. (Requisitos Relacionados: RF2).

RF04 - O sistema deve permitir o gerenciamento do cadastro de colaboradores com os seguintes dados: nome completo, CPF, RG, data de nascimento, sexo, endereço, grupo de usuários, e-mail, telefone e senha. (Requisitos Relacionados: RF2).

RF05 - O sistema deve permitir o gerenciamento do cadastro de beneficiários com os seguintes dados: nome completo, CPF, RG, data de nascimento, sexo, endereço, grupo de usuários, e-mail, telefone e responsáveis legal. (Requisitos Relacionados: RF2).

RF06 - O sistema deve permitir o gerenciamento do cadastro de fornecedores com os seguintes dados: nome, CNPJ, endereço, inscrição estadual, telefone. (Requisitos Relacionados: —).

RF07 - O sistema deve permitir o gerenciamento de contas bancárias com os seguintes dados: nome, descrição, banco, número da agência, número da conta. (Requisitos Relacionados: —).

RF08 - O sistema deve permitir o gerenciamento de categorias de movimentações financeiras com os seguintes dados: nome, descrição, tipo de operação, necessidade de nota fiscal. (Requisitos Relacionados: —).

RF09 - O sistema deve permitir o gerenciamento de movimentações financeiras com os seguintes dados: conta bancária, valor, categoria de movimentação financeira, data de vencimento, CH/OB, descrição, status. (Requisitos Relacionados: RF5, RF6, RF7 e RF8).

RF10 - O sistema deve permitir o gerenciamento de grupos de atividade com os seguintes dados: nome do grupo de atividade, integrantes, tutor, turno, quantidade máxima de integrantes. (Requisitos Relacionados:RF4 e RF5).

RF11 - O sistema deve permitir o gerenciamento de consultas com os seguintes dados: paciente, médico, data, comparecimento, diagnóstico. (Requisitos Relacionados: RF4 e RF5).

4.2.2 Requisitos Não Funcionais

Os Requisitos não funcionais são:

RNF01 - O sistema deve solicitar a alteração de senha no primeiro acesso do usuário. (Requisitos Relacionados: RF4).

RNF02 - O sistema deve limitar o acesso do usuário conforme o cargo. (Requisitos Relacionados: RF3).

RNF03 - RNF3: O sistema não deve permitir o cadastro de um beneficiário sem o devido cadastro de um responsável. (Requisitos Relacionados: RF4).

RNF04 - O sistema pode permitir que um usuário pertença a mais de um grupo. (Requisitos Relacionados: RF3).

RNF05 - O sistema pode permitir a emissão de quaisquer tipo de relatório.

4.3 Visão dos casos de Uso

Esta seção apresenta o diagrama de caso de uso deste trabalho. Apresentamos uma breve introdução sobre os atores presentes nesta documentação, os módulos que compõem o protótipo e o diagrama de entidade e relacionamento.

Administrador: responsável em gerenciar a organização, além de ter acesso a parte financeira e de recursos humanos.

Administrativo: responsável em gerenciar recursos humanos e beneficiários.

Financeiro: responsável em gerenciar as contas da organização.

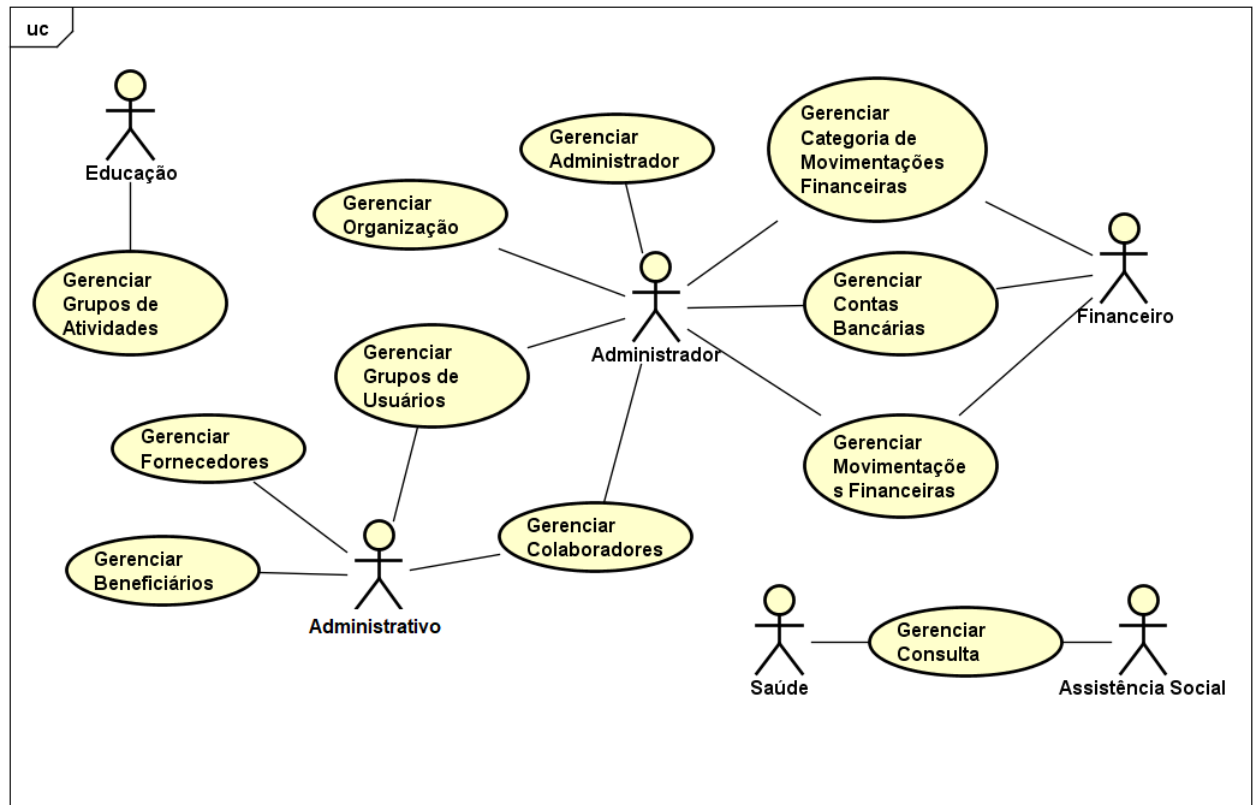
Assistência Social: responsável em gerenciar os atendimentos aos beneficiários.

Saúde: responsável em gerenciar as consultas aos beneficiários.

Educação: responsável em gerenciar os grupos de atividades, tendo o livre acesso de criar novos grupos de atividades.

Na figura 9 podemos observar os atores e suas funções:

Figura 9 – Diagrama do modelo de caso de uso



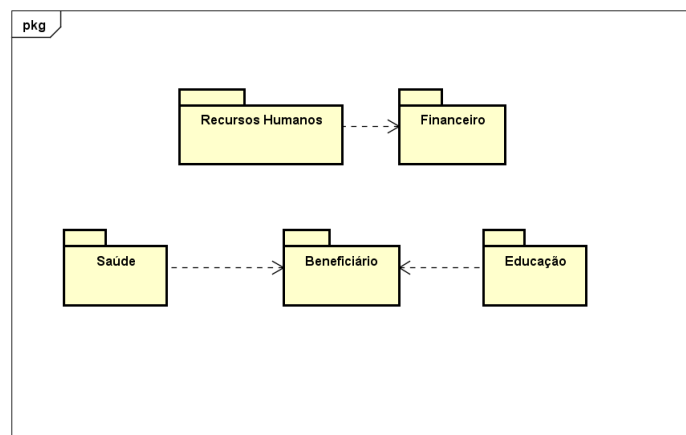
powered by Astah

Fonte: (Elaborada pelos autores)

4.3.1 Módulos

Para o desenvolvimento do protótipo houve uma divisão em módulos, onde cada módulo atende uma área da organização, conforme na figura 10

Figura 10 – Módulos para o desenvolvimento do protótipo



powered by Astah

Fonte: (Elaborada pelos autores)

Beneficiário: neste módulo se encontram todos os cadastros das pessoas que recebem de alguma forma atendimento dentro da organização.

Educação: neste módulo está a parte de controle dos projetos e cursos oferecidos pela organização.

Financeiro: neste módulo está a parte da gestão financeira da organização.

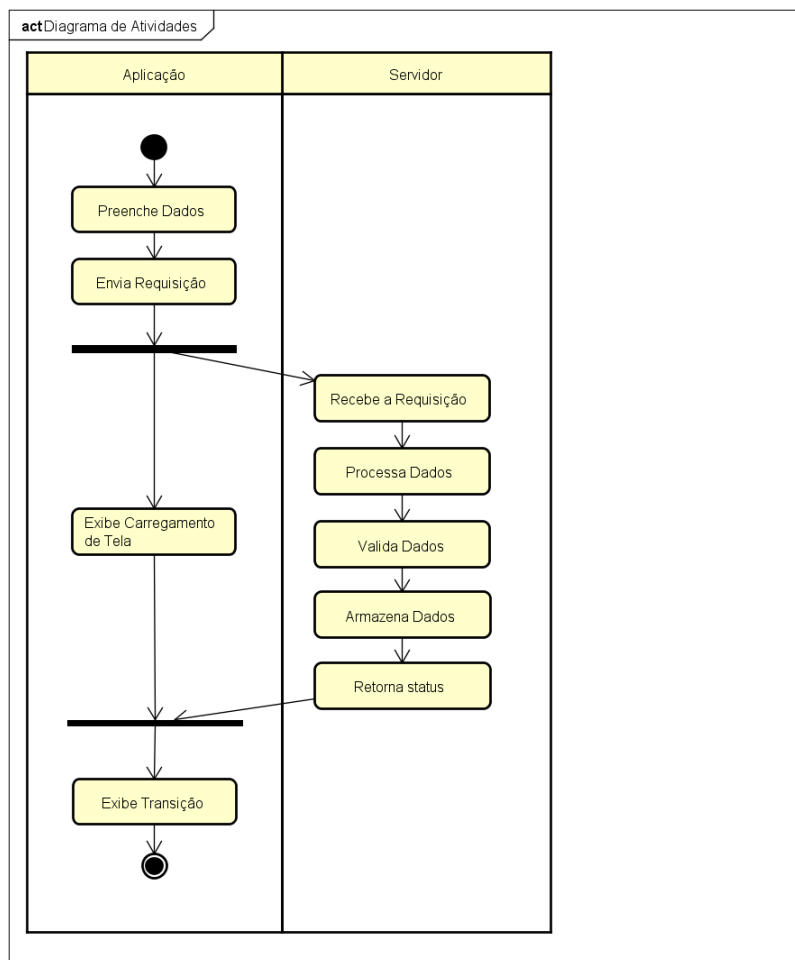
Recursos Humanos: neste módulo se encontra o cadastro dos colaboradores da organização. Tanto os colaboradores remunerados como os voluntários.

Saúde: neste módulo se encontra as informações dos serviços realizados na área da saúde pela organização.

4.3.2 Diagrama de Atividade

Conforme a figura 11 o diagrama de atividades mostra como funciona o principal caso de uso deste desenvolvimento: o gerenciamento de contas bancárias.

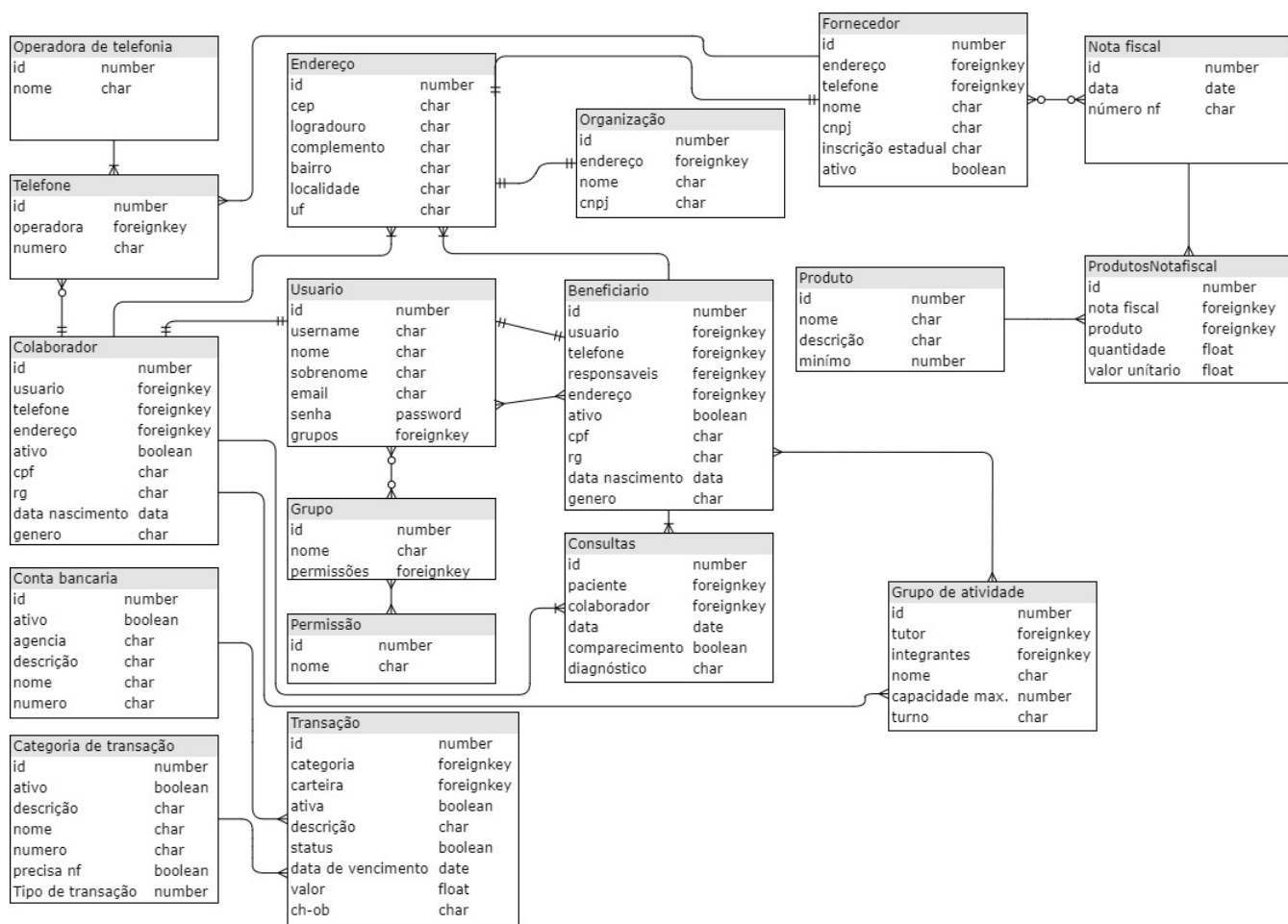
Figura 11 – Diagrama de atividades gerenciamento de contas bancárias



4.3.3 Diagrama Entidade Relacionamento

Através do diagrama apresentado na figura 12, pode-se observar as relações entre os objetos envolvidos. Dessa forma, entendemos melhor as entidades e as relações entre si, facilitando assim a comunicação entre os integrantes da equipe de desenvolvimento.

Figura 12 – Diagrama entidade relacionamento



Fonte: (Elaborada pelos autores)

4.4 Considerações Finais do Capítulo

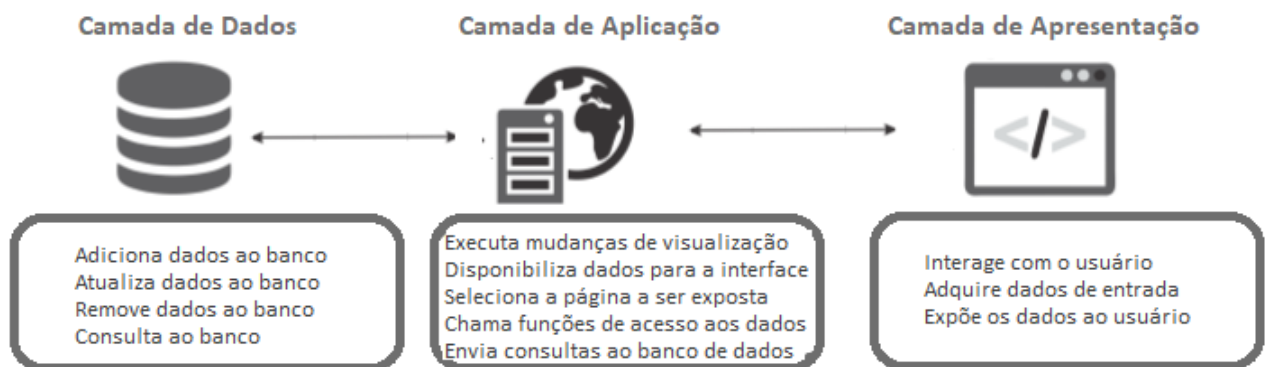
Neste capítulo apresentou-se a documentação para o desenvolvimento do *software*. Os documentos apresentados foram: organizações visitadas, requisitos funcionais, requisitos não funcionais, atores, módulos, diagrama de atividade e diagrama de entidade relacionamento. Dessa forma facilitando o desenvolvimento e manutenção do *software*.

5 DESENVOLVIMENTO

Neste capítulo são apresentadas as principais informações para a implementação do protótipo, a forma como foi desenvolvido, a abordagem adotada, as camadas utilizadas no sistema e problemas encontrados e como eles foram resolvidos.

Conforme a figura 13 o protótipo utiliza uma arquitetura em três camadas: camada de dados, camada de aplicação e camada de apresentação. Esta estrutura possibilita a divisão de responsabilidades, agilizando o desenvolvimento e facilitando a manutenção e a inserção de novas funcionalidades no futuro.

Figura 13 – Estrutura de camadas



Fonte: (Elaborada pelos autores)

5.1 Camada de Dados

Nesta camada, foi desenvolvido um banco de dados relacional responsável pelo armazenamento de informações. Para esta atividade foi utilizada a ferramenta *PostgreSQL* na sua versão 9.6.5 juntamente com o *Object-Relational Mapping* (ORM) do *Django* para a modelagem das tabelas e relacionamentos.

Figura 14 – Trecho de código do modelo de dados das contas bancárias

```
25 class Wallet(models.Model):
26     active = models.BooleanField(default=True)
27     agency = models.CharField(max_length=128, null=True, blank=True)
28     description = models.TextField(null=True, blank=True)
29     label = models.CharField(max_length=128, unique=True)
30     number = models.CharField(max_length=128, null=True, blank=True)
31
32     objects = WalletManager()
33
34     class Meta:
35         verbose_name = _('wallet')
36         verbose_name_plural = _('wallets')
37         ordering = ['label', 'agency', 'number']
38
39     def __str__(self):
40         return self.label
41
42     def balance(self):
43         return self.transactions.aggregate(
44             total=Sum(
45                 Case(
46                     When(active=True, category__transaction_type=TRANSACTION_INPUT, done=True, then=F('value')),
47                     When(active=True, category__transaction_type=TRANSACTION_OUTPUT, done=True, then=-1 * F('value')),
48                     output_field=DecimalField()
49                 )
50             )
51         )['total'] or Decimal('0.0')
52
53     def enable(self):
54         self.active = True
55         self.save()
56
57     def disable(self):
58         self.active = False
59         self.save()
```

Fonte: (Elaborada pelos autores)

Na figura 14 podemos ver, o trecho de código responsável por mapear os dados de uma conta bancária em um objeto *Python*, o qual permite manipular facilmente a informação e desta forma possibilita a aplicação das regras de negócio necessárias.

Este modelo é estruturado utilizando as classes fornecidas pelo ORM do *Django*. Como podemos observar, ele abstrai totalmente a linguagem SQL e ainda provê diversos recursos para as questões de segurança, possibilitando que o desenvolvedor tenha um maior foco para a resolução do problema principal do negócio.

Infelizmente essa abstração pode acarretar em algumas dificuldades em casos específicos. Nas consultas de alta complexidade, ele pode se tornar ineficiente ou até mesmo inviabilizar o seu uso, sendo assim necessário escrever consultas manualmente.

5.2 Camada de Aplicação

Esta é a camada mais extensa, responsável principalmente pelas regras de negócio e também pela comunicação com o banco de dados. Foi desenvolvida com a ferramenta

Django, um *Framework Web* para a linguagem *Python*, a qual é adotada de um sistema de mapeamento objeto-relacional que abstrai grande parte da complexidade de comunicação da aplicação com o banco de dados.

O *Django* é estruturado no padrão de *software MVC*, o qual ainda divide esta camada em três componentes. O modelo responsável pela comunicação com a base de dados, nada mais é do que um ORM. A visão é responsável por entregar dados obtidos através dos modelos à camada de apresentação. A camada de controle é o componente responsável pelas regras de negócio (MICROSOFT, 2017).

Juntamente com o *Django*, também foi utilizado o *plugin Django Rest Framework*, um poderoso e flexível conjunto de ferramentas que estende as funcionalidades básicas do *Django* e possibilita a implementação de uma API.

Figura 15 – Trecho de código da classe de serialização do modelo de contas bancárias

```
6 class WalletSerializer(serializers.ModelSerializer):
7     ... balance = serializers.SerializerMethodField('calculate_balance')
8
9     ... class Meta:
10         ... model = Wallet
11         ... fields = ('id', 'active', 'agency', 'description', 'label', 'number', 'balance')
12
13     ... def calculate_balance(self, wallet):
14         ... return wallet.balance()
```

Fonte: (Elaborada pelos autores)

Na figura 15 podemos observar a classe responsável por converter a estrutura de dados de uma conta bancária mapeada em um objeto *Python* para uma estrutura *JavaScript Object Notation* (JSON). O JSON é uma formatação leve para troca de dados que está sendo amplamente utilizada. A notação de fácil compreensão e de tamanho bem reduzido o torna ideal para troca de mensagens, como uma requisição *HyperText Transfer Protocol* (HTTP).

Figura 16 – Trecho de código das classes de visão das contas bancárias

```
11 class WalletList(generics.ListCreateAPIView):
12     ... queryset = Wallet.objects.all()
13     ... serializer_class = WalletSerializer
14
15
16 class WalletDetail(generics.RetrieveUpdateDestroyAPIView):
17     ... queryset = Wallet.objects.all()
18     ... serializer_class = WalletSerializer
19
20     ... def perform_destroy(self, instance):
21         ... instance.disable()
```

Fonte: (Elaborada pelos autores)

As duas classes apresentadas na figura 16 tem como objetivo responder as requisições feitas pela camada de apresentação. Elas são responsáveis pela listagem, criação, detalhamento, atualização e exclusão de contas bancárias. Como podemos observar, elas herdam duas classes, a *ListCreateAPIView* e a *RetrieveUpdateDestroyAPIView*, do *plugin Django Rest Framework*. E, dentro delas, apenas utilizamos o ORM do *Django* para buscar os elementos na base de dados e os serializadores, também herdados do *plugin Django Rest Framework* (DRF), para converter o modelo em notação JSON a serem enviados como resposta da requisição HTTP da camada de apresentação.

Vale ressaltar que as classes apresentadas na figura 16 são duas classes simples. Sendo apenas um CRUD (*Create, Read, Update e Delete*), sem necessidade de nenhuma regra de negócio muito específica. Desta forma, a combinação do *Django* com o DRF é permitido fazer todas essas operações com pouquíssimas linhas de código.

5.3 Camada de Apresentação

A camada de apresentação é responsável pela interação com usuário, a qual permite a inserção de dados e a visualização da informação armazenada na base de dados. Esta camada foi desenvolvida utilizando a ferramenta *Angular*, um *Framework front-end* desenvolvido pela *Google* focado na produtividade. Com esta ferramenta foi possível criar um aplicação leve e compatível com diversos dispositivos de computação.

Junto com *Angular* foi utilizado o *Materializecss*, um *framework front-end* que implementa o *Material Design*, “*O Material Design é um sistema unificado que combina teoria, recursos e ferramentas para criar experiências digitais.*” (MATERIAL.IO, 2017, tradução nossa). Esta combinação possibilitou a criação de um leiaute de fácil utilização e padronizado com outros *softwares* bem difundidos e consolidados, tais como o *Android*, *YouTube* e vários outros programas da própria *Google*.

O *Angular* é estruturado em dois módulos: componentes e serviços. Tem como princípio separar cada elemento da interface gráfica em um componente totalmente isolado dos demais. Isto facilita o *debug*, o teste e principalmente a manutenção do código. Os módulos são agrupamentos de componentes; e este componente pode ser qualquer elemento do leiaute de uma página, visíveis ou não. A comunicação entre componentes e agentes externos a aplicação é realizada por meio de um serviço. Um serviço deve ser isolado de outros serviços e componentes, e também deve seguir o princípio de responsabilidade única, ou seja, um serviço deve ser responsável pela solução de apenas um único problema.

Após conhecer a estrutura do *Framework*, o primeiro passo na construção da aplicação *Web* foi a elaboração do módulo API, o qual possui dois serviços: um serviço responsável pela comunicação entre a camada de apresentação e a API desenvolvida com

Django e DRF na camada de aplicação; e um serviço para comunicação com a API do site viacep.com.br para a consulta de endereços.

Figura 17 – Endpoints da API Django/DRF dentro do ApiService

```

25     ...this.isAuthenticated.next(localStorage.getItem('user') !== null);
26     ...this._endpoints = {
27     ...// ##### AUTH #####
28     |
29     ...login: () => '/api-token-auth/',
30     ...// ##### WALLETS #####
31     ...listWallets: () => '/wallets/',
32     ...createWallet: () => '/wallets/',
33     ...getWallet: (id: number) => `/wallets/${id}/`,
34     ...updateWallet: (id: number) => `/wallets/${id}/`,
35
36     ...// ##### TRANSACTIONS CATEGORY #####
37     ...listTransactionCategory: () => '/transaction-category/',
38     ...createTransactionCategory: () => '/transaction-category/',
39     ...getTransactionCategory: (id: number) => `/transaction-category/${id}/`,
40     ...updateTransactionCategory: (id: number) => `/transaction-category/${id}/`,
41
42     ...// ##### TRANSACTIONS #####
43     ...listTransactions: () => '/transactions/',
44     ...createTransaction: () => '/transactions/',
45     ...listWalletTransactions: (id: number) => `/transactions/wallet/${id}/`,
46     ...listCategoryTransactions: (id: number) => `/transactions/category/${id}/`,
47     ...getTransaction: (id: number) => `/transactions/${id}/`,
48     ...updateTransaction: (id: number) => `/transactions/${id}/`,
49     ...exportTransactions: () => '/transactions/csv/',
50     ...chartTransactions: () => '/transactions/chart/',
51
52     ...// ##### COLLABORATORS #####
53     ...listCollaborators: () => '/collaborators/',
54     ...createCollaborator: () => '/collaborators/',
55     ...getCollaborator: (id: number) => `/collaborators/${id}/`,
56     ...updateCollaborator: (id: number) => `/collaborators/${id}/`,
57
58     ...// ##### BENEFICIARY #####
59     ...listBeneficiaries: () => '/profiles/',
60     ...createBeneficiary: () => '/profiles/',
61     ...getBeneficiary: (id: number) => `/profiles/${id}/`,
62     ...updateBeneficiary: (id: number) => `/profiles/${id}/`,
63     ...};
64     ...}

```

Fonte: (Elaborada pelos autores)

Na figura 17 podemos visualizar um trecho do código, onde são armazenados os *endpoints* da API *Django*/DRF utilizados pelo serviço *ApiService* do *Angular*. Já na figura 18, podemos visualizar os métodos do serviço responsáveis pela autenticação de usuário junto a camada de aplicação.

Figura 18 – Métodos de autenticação do ApiService

```

65 authenticate(login: Login) {
66   this.http
67     .post(this.url('login'), login)
68     .toPromise()
69     .then(resp => {
70       localStorage.setItem('user', resp.text());
71       this.isAuthenticated.next(true);
72     })
73     .catch(resp => {
74       this.isAuthenticated.next(false);
75     });
76 }
77
78 getToken(): string {
79   const user = JSON.parse(localStorage.getItem('user') || '{}') as User;
80   return user ? user.auth_token : '';
81 }
82
83 getUser(): User {
84   return JSON.parse(localStorage.getItem('user') || '{}') as User;
85 }
86
87 logout(): void {
88   localStorage.removeItem('user');
89   this.isAuthenticated.next(false);
90 }

```

Fonte: (Elaborada pelos autores)

Figura 19 – Módulo Wallet

```

1 | import { CommonModule } from '@angular/common';
2 | import { FormsModule } from '@angular/forms';
3 | import { NgModule } from '@angular/core';
4 |
5 | import { CoreModule } from '../core/core.module';
6 | import { MaterializeModule } from 'angular2-materialize';
7 | import { TransactionModule } from '../transaction/transaction.module';
8 | import { WalletEditorComponent } from './wallet-editor/wallet-editor.component';
9 | import { WalletListComponent } from './wallet-list/wallet-list.component';
10 | import { WalletRoutingModule } from './wallet.routing.module';
11 | import { WalletViewerComponent } from './wallet-viewer/wallet-viewer.component';
12 |
13 | @NgModule({
14 |   imports: [
15 |     CommonModule,
16 |     CoreModule,
17 |     FormsModule,
18 |     MaterializeModule,
19 |     TransactionModule,
20 |     WalletRoutingModule,
21 |   ],
22 |   declarations: [
23 |     WalletListComponent,
24 |     WalletEditorComponent,
25 |     WalletViewerComponent,
26 |   ],
27 |   exports: [
28 |   ]
29 | })
30 | export class WalletModule {}
31 |

```

Fonte: (Elaborada pelos autores)

Com a comunicação entre as duas camadas prontas, foi iniciado o desenvolvimento do módulo financeiro, o qual foi dividido em três outros módulos: *wallet*, *transaction-category* e *transaction*. Estes módulos foram divididos em três componentes, sendo eles: *wallet-editor*, *wallet-list*, *wallet-viewer*, *transaction-category-editor*, *transaction-category-list*, *transaction-category-viewer*, *transaction-editor*, *transaction-list* e *transaction-viewer*. Podemos visualizar exemplos destes elementos nas figuras 19, 20, 21 e 22 respectivamente.

Figura 20 – Componente wallet-editor

```
15 export class WalletEditorComponent implements OnInit {
16   ·wallet: Wallet = null;
17   ·loading: boolean = true;
18   ·constructor(
19     ·private route: ActivatedRoute,
20     ·private router: Router,
21     ·private location: Location,
22     ·private apiService: ApiService
23   ) {
24     ·this.wallet = new Wallet();
25     ·this.wallet.active = true;
26   }
27   ngOnInit() {
28     ·const walletId = this.route.snapshot.params['id'];
29     ·if (walletId) {
30       ·this.apiService
31         ·.params(walletId)
32         ·.get('getWallet')
33         ·.toPromise()
34         ·.then(resp => {
35           ·this.wallet = resp.json() as Wallet;
36           ·this.loading = false;
37         })
38         ·.catch(error => console.log(error));
39     }
40   }
41   submit() {
42     ·let request;
43     ·if (this.wallet.id) {
44       ·request = this.apiService
45         ·.data(this.wallet)
46         ·.params(this.wallet.id)
47         ·.put('updateWallet')
48         ·.toPromise();
49     } else {
50       ·request = this.apiService
51         ·.data(this.wallet)
52         ·.post('createWallet')
53         ·.toPromise();
54     }
55     ·request.then(resp => {
56       ·this.router.navigate(['/wallets', resp.json().id]);
57     });
58   }
59   cancel() {
60     ·this.location.back();
61   }
62 }
```

Fonte: (Elaborada pelos autores)

Figura 21 – Componente wallet-list

```

1 | import { ActivatedRoute } from '@angular/router';
2 | import { Component, OnInit, OnDestroy } from '@angular/core';
3 |
4 | import { ApiService } from '../api/api.service';
5 | import { Observable, Subscription } from 'rxjs/Rx';
6 | import { Page } from '../core/pagination/page';
7 | import { Wallet } from '../wallet';
8 |
9 | import 'rxjs/add/operator/toPromise';
10 |
11 | @Component({
12 |   selector: 'app-wallet-list',
13 |   templateUrl: './wallet-list.component.html',
14 |   styleUrls: ['./wallet-list.component.css']
15 | })
16 | export class WalletListComponent implements OnInit, OnDestroy {
17 |   wallets: Wallet[];
18 |   page: Page;
19 |   queryParamsSubscription: Subscription;
20 |   loading: boolean = true;
21 |   constructor(private activatedRoute: ActivatedRoute, private apiService: ApiService) {}
22 |
23 |   ngOnInit() {
24 |     this.queryParamsSubscription = this.activatedRoute.queryParams.subscribe(params => {
25 |       this.apiService
26 |         .queryParams(params)
27 |         .get('listWallets')
28 |         .toPromise()
29 |         .then(resp => {
30 |           const response = resp.json();
31 |           this.wallets = response.results as Wallet[];
32 |           this.page = response.page as Page;
33 |           this.loading = false;
34 |         });
35 |     });
36 |   }
37 |
38 |   ngOnDestroy() {
39 |     this.queryParamsSubscription.unsubscribe();
40 |   }
41 | }
42 |

```

Fonte: (Elaborada pelos autores)

Figura 22 – Componente wallet-viewer

```

1 | import { ActivatedRoute, Router } from '@angular/router';
2 | import { Component, OnInit } from '@angular/core';
3 |
4 | import { ApiService } from '../api/api.service';
5 | import { Wallet } from '../wallet';
6 |
7 | import 'rxjs/add/operator/toPromise';
8 |
9 | @Component({
10 |   selector: 'app-wallet-viewer',
11 |   templateUrl: './wallet-viewer.component.html',
12 |   styleUrls: ['./wallet-viewer.component.css']
13 | })
14 | export class WalletViewerComponent implements OnInit {
15 |   wallet: Wallet = null;
16 |   loading: boolean = true;
17 |   constructor(
18 |     private route: ActivatedRoute,
19 |     private router: Router,
20 |     private apiService: ApiService
21 |   ) {}
22 |
23 |   ngOnInit() {
24 |     const walletId = this.route.snapshot.params['id'];
25 |     if (walletId) {
26 |       this.apiService
27 |         .params(walletId)
28 |         .get('getWallet')
29 |         .toPromise()
30 |         .then(resp => {
31 |           this.wallet = resp.json() as Wallet;
32 |           this.loading = false;
33 |         })
34 |         .catch(error => console.log(error));
35 |     }
36 |   }
37 | }
38 |

```

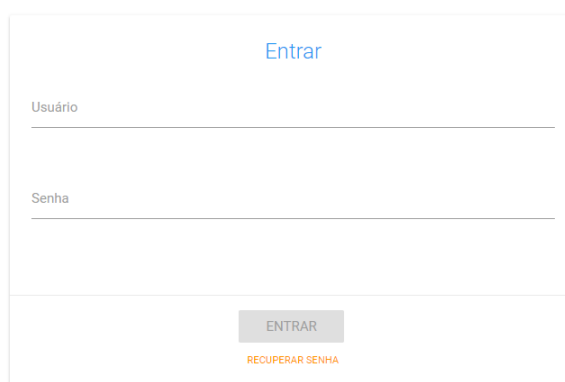
Fonte: (Elaborada pelos autores)

5.3.1 Capturas de Tela

Nessa subseção são apresentadas capturas de algumas telas da aplicação, todas preenchidas com dados de testes.

Para obter acesso ao sistema o usuário deverá realizar o login com seu C.P.F. e senha, como podemos visualizar na figura 23.

Figura 23 – Captura de tela do protótipo: tela de login

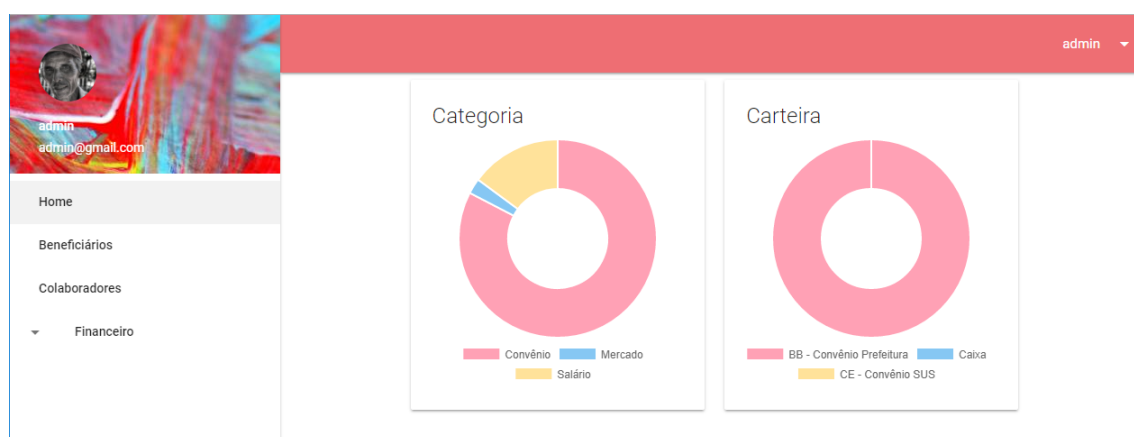


A captura de tela mostra uma interface de login com o título "Entrar" no topo. Abaixo, há dois campos de entrada: "Usuário" e "Senha". No rodapé, há um botão "ENTRAR" em cinza e um link "RECUPERAR SENHA" em laranja.

Fonte: (Elaborada pelos autores)

Na figura 24 podemos observar a tela inicial, a qual apresenta informações relevantes ao cargo do usuário. Neste caso um contador esta visualizando gráficos das movimentações separadas por categorias de despesas e por contas bancárias.

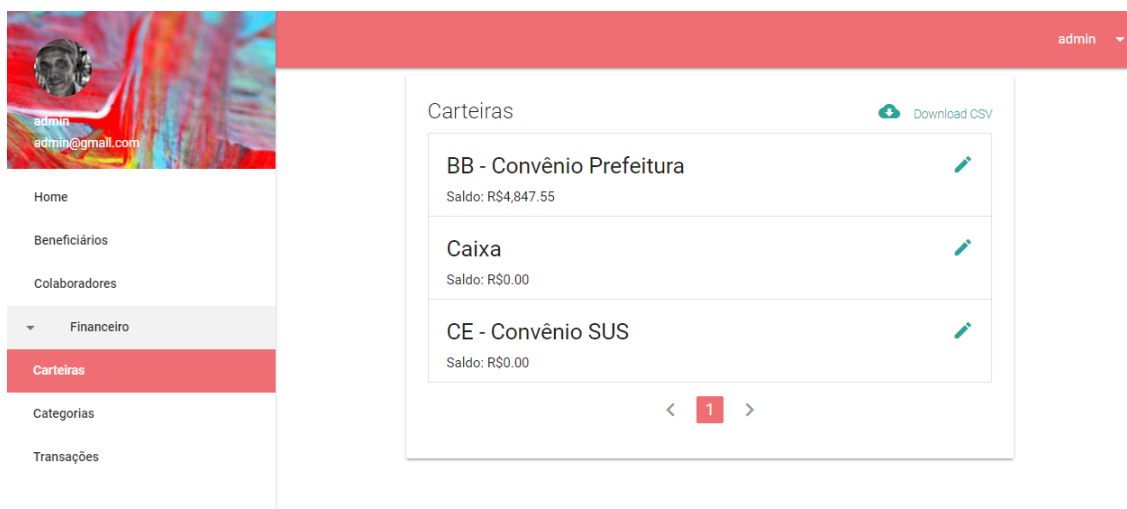
Figura 24 – Captura de tela do protótipo: tela inicial



Fonte: (Elaborada pelos autores)

O protótipo conta a listagem das contas bancárias e seus respectivos saldos, conforme na figura 25. Para ter acesso a esta tela é necessário estar incluso no setor financeiro.

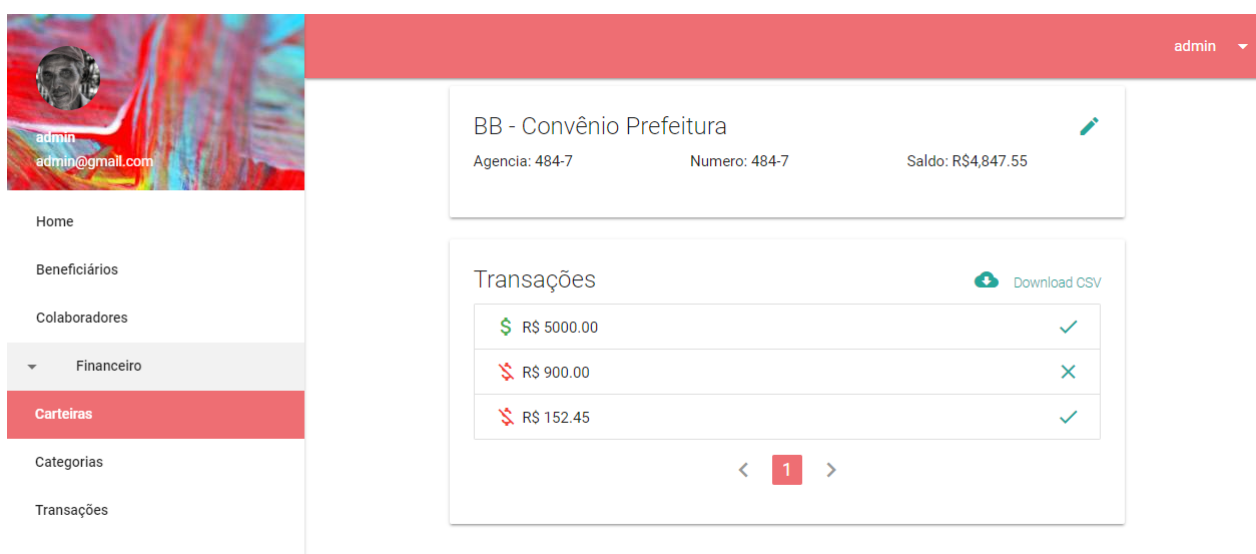
Figura 25 – Captura de tela do protótipo: listagem de contas bancárias



Fonte: (Elaborada pelos autores)

Já na figura 26, podemos observar as transações de uma determinada conta bancária e um botão para exportação das respectivas transações em *Comma-separated values* (CSV).

Figura 26 – Captura de tela do protótipo: listagem de transações de uma conta bancária



Fonte: (Elaborada pelos autores)

Na figura 27 encontra-se a tela para a edição das categorias de transação, onde o usuário tem total autonomia para a edição destes campos. Além de que, os campos obrigatórios estão identificados com um * (asterisco).

Figura 27 – Captura de tela do protótipo: tela de edição de uma categoria de transação

admin

admin@gmail.com

Home

Beneficiários

Colaboradores

▼ Financeiro

Carteiras

Categorias

Transações

Editor de categorias de transações

* campo obrigatório

Desabilitada Habilitada

Tipo de transação*

Entrada

Saída

Nome*

Convênio

Descrição

Precisa de nota fiscal

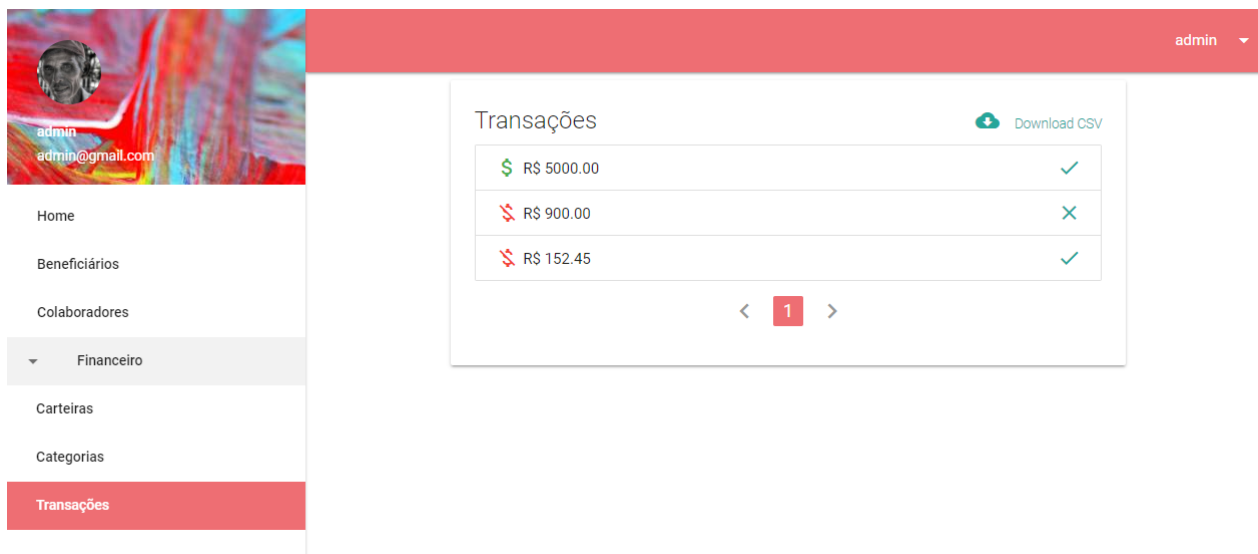
Não Sim

× CANCELAR SALVAR

Fonte: (Elaborada pelos autores)

Na figura 28 podemos observar a listagem de transações cadastradas no sistema. No ícone a esquerda é possível identificar rapidamente o tipo da movimentação, vermelho quando é saída e verde quando é entrada. No ícone da direita, podemos identificar o status da transação que tem um formato de *V* quando ainda foi efetuada e de um *X* quando não foi efetuada.

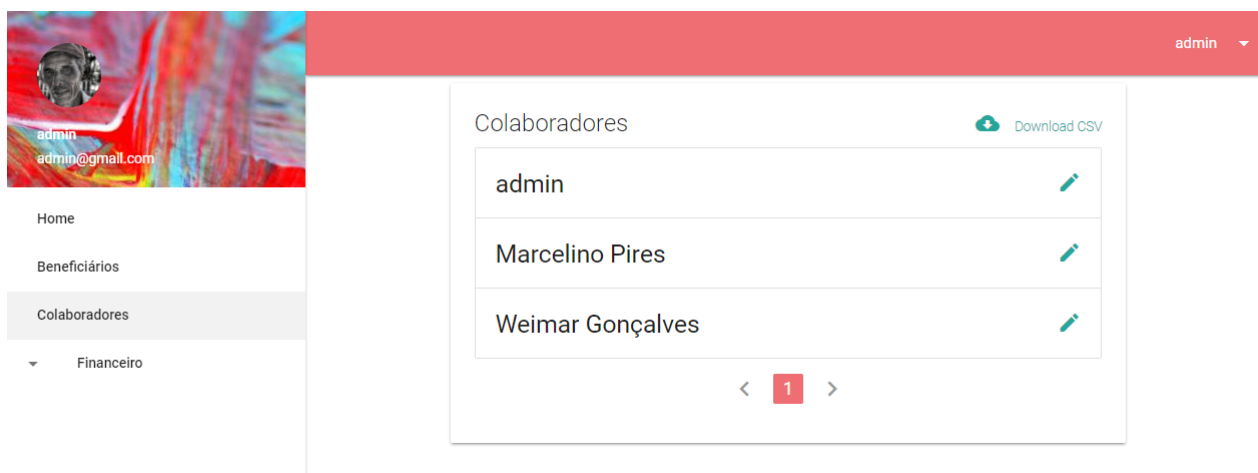
Figura 28 – Captura de tela do protótipo: listagem de transações



Fonte: (Elaborada pelos autores)

Encontra-se na figura 29 a listagem de colaboradores da organização cadastrados. Onde se encontra todas as informações pessoais de cada colaborador.

Figura 29 – Captura de tela do protótipo: listagem de colaboradores



Fonte: (Elaborada pelos autores)

Finalmente na figura 30, podemos observar uma ficha de cadastro de um beneficiário. Na parte superior se encontra o botão de *status*, onde se o beneficiário não recebe mais atendimento, o botão fica no modo desativado.

Figura 30 – Tela

The screenshot displays a web interface for editing a beneficiary. On the left is a sidebar with a user profile and navigation menu. The main content area is titled 'Editor de Beneficiario' and contains a form with the following fields:

- Ativo:** A toggle switch currently set to 'Ativo' (Active).
- Login*:** joaozinho
- E-mail*:** joaozinho@gmail.com
- Nome*:** Joaozinho
- Sobrenome*:** Silva Santos
- R.G.*:** 213123123
- C.P.F.*:** 058.171.427-08
- Data de nascimento*:** 1994-06-14
- Sexo*:** Masculino
- Responsáveis:** (empty dropdown)
- Endereço:**
 - C.E.P.*:** 79820-000
 - Logradouro*:** Rua Cornélia C. de Souza
 - Complemento:** (empty)
 - Bairro*:** Jardim Climax
 - Cidade*:** Dourados
 - Estado*:** MS

At the bottom of the form are two buttons: 'CANCELAR' (orange) and 'SALVAR' (green).

Fonte: (Elaborada pelos autores)

5.4 Considerações Finais do Capítulo

Neste capítulo retratamos o processo de desenvolvimento do protótipo, abordando os processos e ferramentas utilizadas, assim como, as telas do sistema.

6 RESULTADOS

Nesta seção são demonstrados os testes que foram realizados, público alvo e os resultados obtidos. Apresenta-se também, uma breve análise dos resultados obtidos.

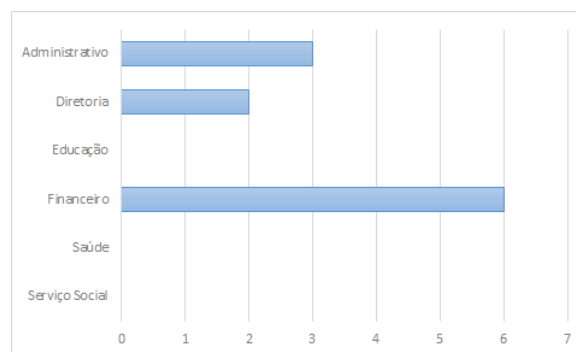
Para a obtenção dos resultados, disponibilizamos o protótipo para os colaboradores das organizações. Durante o tempo em que os colaboradores tiveram acesso ao sistema, os mesmos puderam usar todas as funções oferecidas pelo protótipo. Para a realização dos testes, o módulo financeiro estava totalmente implementado. Já os módulos de recursos humanos e beneficiários estavam parcialmente desenvolvidos.

Após a fase de testes, os colaboradores responderam um questionário *online*, onde compreendiam 11 perguntas. Para a obtenção de resultados sólidos, as perguntas foram divididas em dois segmentos: identificação básica e entendimento do protótipo. O questionário e a planilha de respostas da pesquisa encontram-se nos Apêndices A e B respectivamente.

6.1 Resultados: Identificação Básica

Inicialmente, obtivemos 9 questionários respondidos, no segmento de identificação básica tivemos as seguintes porcentagens. Inicialmente identificamos em qual área a pessoa atuava na organização. Esta era a única pergunta em que o entrevistado poderia escolher mais de uma opção, pois quando visitamos e conhecemos a realidade das organizações, tivemos relatos que uma pessoa obtinha mais de uma função, chegando assim aos seguintes dados: 54,5% atuam na área financeira, 27,3% na área administrativa e 18,2% na diretoria da organização.

Figura 31 – Resultados: Função dos colaboradores na organização



Fonte: (Elaborada pelos autores)

Em relação a faixa etária, 55,6% possui a idade de 15 e 29 anos e, o restante, 44,4%

tem mais de 30 anos. Já quando perguntado sobre a sua escolaridade, uma maioria (66,7%) já concluíram o nível superior e 33,3% possui o ensino médio completo. Finalizando este segmento, perguntamos o conhecimento dos mesmos em relação a informática. 11,2% dos entrevistados consideraram seu conhecimento como baixo. 44,4% entende que seu conhecimento é mediano e a outra porção de 44,4% tem conhecimento pleno.

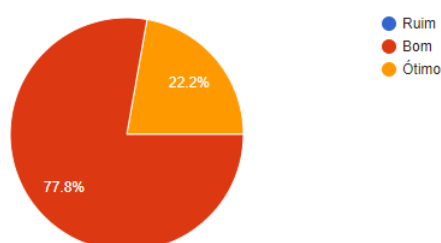
6.2 Resultados: Conhecimento do Protótipo

Após o conhecimento do público alvo, iniciamos o segmento para a avaliação do protótipo. Quando questionados se o protótipo era considerado compreensível, 100% chegou ao entendimento que era um protótipo de fácil manuseio. Outra questão, que também obteve a unanimidade foi a de que o protótipo atendia parcialmente as funções de seu setor.

Quando questionados se o protótipo era intuitivo, 77,8% entenderam que sim. E 22,2% compreenderam que era de forma mediana. Ao perguntar se o protótipo facilitaria suas funções, chegamos ao resultado de que 77,8%, facilitaria de uma forma mediana. 11,1% responderam que atendia completamente e por fim 11,1% chegou a conclusão que não era compatível para as suas funções.

A última pergunta de múltipla escolha foi qual a sua avaliação para o protótipo. 77,8% avaliaram com o conceito bom e 22,2% com o conceito ótimo e nenhuma avaliação ruim.

Figura 32 – Resultados: Avaliação final do protótipo



Fonte: (Elaborada pelos autores)

Na primeira questão em aberta, perguntamos quais funcionalidades poderiam ser melhoradas, dentre as sugestões colocadas podem ser citadas ter mais funções e campos no módulo financeiro. E na outra questão, perguntamos quais funcionalidades poderiam ser adicionadas, e como respostas tivemos o acréscimo dos módulos que não foram desenvolvidos.

6.3 Considerações Finais do Capítulo

Podemos observar que as pessoas que estão trabalhando na área financeira dessas organizações são jovens com idade entre 15 e 29 anos, com nível superior completo e um bom conhecimento relacionado a informática. Percebemos que, mesmo que a maioria das pessoas são graduadas, as mesmas não consideram dispor de um ótimo conhecimento na área de informática.

Também constatamos que foi unânime a fácil utilização do protótipo e que atendia de forma parcial as funções realizadas. Nas demais indagações, 77,8% entendeu que o protótipo era intuitivo, facilitaria de forma mediana as suas funções e que o protótipo estava bom.

7 CONCLUSÃO

Conforme o estudo realizado, podemos perceber que realmente há uma grande lacuna a ser preenchida neste setor. Percebemos que não há *softwares* acessíveis para as organizações. Com a falta de um maior amparo, a organização encontra uma grande dificuldade em gerir suas atividades, o que culmina em dificuldades de seu crescimento.

Tendo como objetivo o desenvolvimento de um protótipo para a gestão de organizações sem fins lucrativos, acreditamos que este trabalho atingiu grande parte das expectativas dos autores do projeto, quanto o apreço das organizações envolvidas. Devido ao fato de que a implementação da aplicação *Web* foi parcialmente desenvolvida, mas servidor de aplicação em sua versão atual, responde as requisições do cliente *Web* corretamente e em um tempo hábil.

Comparando-se com os *softwares* discutidos durante o estudo, podemos concluir que a aplicação desenvolvida demonstrou ser eficiente. Além disso, as melhores características da ferramenta são o fato de ser totalmente gratuita e aberta, o que permite a própria comunidade realizar correções, adaptações e melhorias.

Diante dos resultados obtidos por meio da aplicação dos questionários, podemos afirmar que o protótipo atingiu parcialmente seus objetivos. As respostas mostraram que os recursos da aplicação são compatíveis com as suas realidades e necessidades dos usuários das organizações, assim facilitando seu trabalho.

7.1 Melhorias Futuras

Como melhorias futuras deixamos como sugestão a implementação dos módulos *Educação e Saúde* que correspondem aos requisitos RF06, RF10 e RF11.

Sugerimos ainda, a possibilidade de expansão para organizações de outras áreas, como defesa aos animais, e defesa e proteção do meio ambiente.

REFERÊNCIAS

- ANGULAR.IO. *Angular*. 2017. Disponível em <<https://angular.io/>>. Acesso em 06/11/2017.
- ARAÚJO, F. S. et al. Avaliação da experiência do usuário: uma proposta de sistematização para o processo de desenvolvimento de produtos. 2014.
- BRAGA, A. S. et al. Design de interface-as origens do design e sua influência na produção da hipermídia. Pontifícia Universidade Católica de São Paulo, 2004.
- CAELUM.COM.BR. *Blog Caelum*. 2017. Disponível em <<https://www.caelum.com.br/apostila-html-css-javascript/javascript-e-interatividade-na-web/11-1-caracteristicas-da-linguagem>>. Acesso em 06/11/2017.
- CAELUM.COM.BR. *Blog Caelum*. 2017. Disponível em <<http://blog.caelum.com.br/como-anda-o-angular-js-devo-embarcar-nessa/>>. Acesso em 06/11/2017.
- DIEDRICH, C. *O que é Docker?* 2015. Disponível em <<http://www.mundodocker.com.br/o-que-e-docker/>>. Acesso em 17/07/2017.
- DOMENEGHETTI, A. M. *Voluntariado-gestão do trabalho*. [S.l.]: Editora Esfera, 2001.
- DOMINGUES, J. T. A importância dos sistema de informação gerencial para as empresas. 2014. Disponível em <<http://www.administradores.com.br/artigos/academico/a-importancia-dos-sistema-de-informacao-gerencial-para-as-empresas/78358/>>.
- FAZFÁCIL. *App que significa e Webapp ? É internautês... saiba tudo*. 2017. Disponível em <<http://www.fazfacil.com.br/manutencao/o-que-sao-apps/>>. Acesso em 28/05/2017.
- FERNANDES, J. H. C. O que é um programa (software)? 2002. Disponível em <<http://www.cic.unb.br/jhcf/MyBooks/iess/Software/oqueehsoftware.html>>. Acesso em 25/05/2017.
- FERREIRA, V. C. P. *ONGs no Brasil: um estudo sobre suas características e fatores que têm induzido seu crescimento*. Tese (Doutorado), 2005.
- FIELDING, R. T.; TAYLOR, R. N. *Architectural styles and the design of network-based software architectures*. [S.l.]: University of California, Irvine Doctoral dissertation, 2000.
- FRAMEWORK, D. *Django: the Web framework for perfectionists with deadlines*. 2015.
- FRANCISCO, W. d. C. e. *"Organização Não Governamental (ONG) "; Brasil Escola*. 2010. Disponível em <<http://brasilecola.uol.com.br/geografia/organizacao-nao-governamental-ong.htm>>. Acesso em 23/05/2017.
- GOHN, M. d. G. Mídia terceiro setor e mst: impactos sobre o futuro das cidades e do campo. *Cadernos de Pesquisa*, SciELO Brasil, n. 112, p. 213–213, 2001.
- GOMES, R.; SOUZA, R. Docker - infraestrutura como código, com autonomia e replicabilidade. 2015.

- HOLOVATY, A.; KAPLAN-MOSS, J. et al. The django book. <http://www.djangobook.com/en/1.0>, 2007.
- IBGE. *As Entidades de Assistência Social Privadas sem Fins Lucrativos no Brasil Voluntariado-gestão do trabalho*. [S.l.: s.n.], 2007.
- LIMA, M. B.; SILVA, K. M. Processo de prestação de contas de projetos firmados com as ong's do df por meio do governo federal. In: *Anais da Conferência Internacional de Estratégia em Gestão, Educação e Sistemas de Informação (CIEGESI)*. [S.l.: s.n.], 2013. v. 1, n. 1, p. 210–237.
- LOPES, E. C. Uma (re) visão do conceito de experiência do usuário: a experiência como narrativa. *Especialização em Comunicação Digital. Escola de Comunicações e Artes. São Paulo: USP*, 2012.
- LUCA, C. d. A. et al. O terceiro setor na economia brasileira. Florianópolis, 2008.
- LUTZ, M.; ASCHER, D. *Aprendendo Python, 2*. [S.l.]: Bookman, 2007.
- MATERIAL.IO. *Material Design*. 2017, tradução nossa. Disponível em <<https://material.io/>>. Acesso em 05/09/2017.
- MCGEE, J.; PRUSAK, L. Gerenciamento estratégico da informação: aumente a competitividade e a eficiência de sua empresa utilizando a informação como uma ferramenta estratégica. *Rio de janeiro: Campus*, p. 36–54, 1994.
- MEREGE, L. C. Terceiro setor: finalmente no pib. *Revista Integração. Centro de*, 2007.
- MICROSOFT, M. *ASP.NET MVC Overview*. 2017. Disponível em <<https://msdn.microsoft.com/en-us/library/dd381412%28v=vs.108%29.aspx?f=255MSPPErrror=-2147217396>>. Acesso em 04/09/2017.
- MORVILLE, P. *User Experience Strategy*. 2004. Disponível em <<http://semanticstudios.com/publications/semantics/000029.php>>. Acesso em 20/06/2017.
- NERY, R. *Visão geral do JavaScript*. 2017. Disponível em <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/JavaScript_Vis%C3%A3o_Geral>. Acesso em 12/07/2017.
- PYTHON SOFTWARE, F. *About Python | Python.org*. 2017. Disponível em <<https://www.python.org/about/>>. Acesso em 26/06/2017.
- ROBERTO, W. d. S. Restful web services e a api jax-rs. 2015.
- SOURCEFORGE. *Uma breve história do PostgreSQL*. 2015. Disponível em <<http://pgdocptbr.sourceforge.net/pg80/history.html>>. Acesso em 16/07/2017.

Apêndices

APÊNDICE A – QUESTIONÁRIO

Este questionário tem como objetivo avaliar o protótipo que foi desenvolvido para a gestão de organizações sem fins lucrativos.

- 1) Área de atuação na organização.
 - a. Administrativo;
 - b. Diretoria;
 - c. Educação;
 - d. Financeiro;
 - e. Saúde;
 - f. Serviço Social.
- 2) Faixa Etária.
 - a. 15 anos a 29 anos;
 - b. 30 anos a 45 anos;
 - c. 46 anos a 55 anos;
 - d. Mais de 56 anos.
- 3) Escolaridade.
 - a. Ensino Fundamental Completo;
 - b. Ensino Médio Completo;
 - c. Ensino Superior Completo.
- 4) Nível de conhecimento em informática.
 - a. Nenhum;
 - b. Pouco;
 - c. Médio;
 - d. Alto.
- 5) O sistema é de fácil utilização?
 - a. Fácil;
 - b. Mediano;
 - c. Difícil.
- 6) O sistema é intuitivo?
 - a. Pouco;
 - b. Mediano;
 - c. Muito.
- 7) O sistema atende as necessidades de sua função?
 - a. Não atende;

b. Parcialmente;

c. Integralmente.

8) O sistema facilitará as atividades da sua função?

a. Pouco;

b. Mediano;

c. Completamente.

9) Qual a sua avaliação para o sistema?

a. Ruim;

b. Bom;

c. Ótimo.

10. Quais funcionalidades podem ser melhoradas?

11. Quais funcionalidades podem ser adicionadas?

APÊNDICE B – RESPOSTAS DO QUESTIONÁRIO

Tabela 1 – Respostas dos questionários

Data e Hora	Q1	Q2	Q3	Q4	Q5	Q6	Q7	Q8	Q9
11/09/2017 20:50	d	a	c	d	a	c	b	b	b
12/09/2017 08:37	a	a	b	d	a	c	b	b	c
13/09/2017 18:31	b	c	c	b	a	c	b	c	b
14/09/2017 15:35	a,c	b	c	c	a	c	b	b	c
14/09/2017 15:46	d	a	c	d	a	c	b	b	b
14/09/2017 17:22	a	a	b	c	a	c	b	b	b
14/09/2017 18:18	d	b	b	c	a	c	b	b	b
14/09/2017 18:53	b,d	c	c	c	a	b	b	b	b
15/09/2017 15:20	d	a	c	d	a	b	b	a	b

Fonte: (Elaborada pelos autores)