



INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DO PIAUÍ  
CAMPUS TERESINA CENTRAL  
TECNOLOGIA EM ANÁLISE E DESENVOLVIMENTO DE SISTEMAS

RAFAEL MADUREIRA LINS DE ARAÚJO

**AJUDA.AI - UMA SOLUÇÃO EM SOFTWARE PARA CROWDFUNDING SOCIAL**

TERESINA, PIAUÍ

2017

RAFAEL MADUREIRA LINS DE ARAÚJO

AJUDA.AI - UMA SOLUÇÃO EM SOFTWARE PARA CROWDFUNDING SOCIAL

Projeto apresentado à Banca Examinadora como  
requisito para aprovação na disciplina de Traba-  
lho de Conclusão de Curso II do Curso Superior  
de Tecnologia em Análise e Desenvolvimento  
de Sistemas do Instituto Federal de Educação,  
Ciência e Tecnologia do Piauí.

**Orientador:** Prof. M<sup>e</sup>. Ely da Silva Miranda

TERESINA, PIAUÍ

2017

RAFAEL MADUREIRA LINS DE ARAÚJO

AJUDA.AI - UMA SOLUÇÃO EM SOFTWARE PARA CROWDFUNDING SOCIAL

Projeto apresentado à Banca Examinadora como requisito para aprovação na disciplina de Trabalho de Conclusão de Curso II do Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas do Instituto Federal de Educação, Ciência e Tecnologia do Piauí.

Aprovado pela banca examinadora em \_\_\_\_\_.

BANCA EXAMINADORA:

---

**Prof. M<sup>e</sup>. Ely da Silva Miranda**  
Instituto Federal de Educação,  
Ciência e Tecnologia do Piauí - IFPI

---

**Prof. M<sup>e</sup>. Rogério da Silva**  
Instituto Federal de Educação,  
Ciência e Tecnologia do Piauí - IFPI

---

**Prof. M<sup>e</sup>. Duany Dreyton  
Bezerra Sousa**  
Instituto Federal de Educação,  
Ciência e Tecnologia do Piauí - IFPI

---

**Prof. Dr. Ney Paranaguá de  
Carvalho**  
Instituto Federal de Educação,  
Ciência e Tecnologia do Piauí - IFPI

TERESINA, PIAUÍ

2017

Dedico este trabalho a todos que acreditam que ele sairia.

## **AGRADECIMENTOS**

Agradeço primeiramente a meus pais, família e Alaydes Morais que, assim como muitos, aguardaram pacientes pela conclusão do curso e me deram o apoio necessário para terminar a jornada.

Agradeço aos amigos, virtuais e reais, pelas risadas, apoio, dicas e vários “eu fiz meu TCC em  $N$  dias” — Em especial Rafael Soares, Rafarpo, Ebbitt, Rosenrot, Zenlee, Tajaro e Velinde.

Agradeço profundamente aos mestres e professores que partilharam de seus conhecimentos por todos esses anos.

Finalmente agradeço a meu orientador, Ely Miranda, por sua amizade, conselhos e excelentes revisões que tornaram esse trabalho possível.

*“Acredito que o poder de qualquer recurso - especialmente o financeiro - é potencializado quando vem com suporte da comunidade. O capital deixa de ser apenas capital, não mais apenas um meio para um fim. É uma coisa fundamentalmente diferente: uma expressão das melhores intenções de uma comunidade, suas esperanças para o futuro, seu desejo de participar em tornar esse futuro uma realidade. É uma asserção da sua confiança que quem receberá o capital vai liderar o caminho.”*

*(Jessica Jackley, tradução livre)*

## RESUMO

Empreendimentos sociais no Brasil historicamente têm bastante dificuldade em buscar o financiamento necessário para exercer de forma plena as atividades que necessitam e frequentemente se utilizam de campanhas de trabalho voluntário e doações de alimentos ou objetos usados. Apesar da redução de custos que trabalho voluntário e doações trazem é inegável a necessidade de dinheiro para custos como eletricidade, água e medicamentos. Campanhas de arrecadação financeira de organizações menores são comumente discretas, como pedir o troco de uma compra como uma pequena doação.

Para resolver esses problemas há soluções como buscar investidores, micro crédito junto a bancos comunitários, campanhas em mídias por doações e, mais recentemente e com o crescimento dessa modalidade no Brasil, *crowdfunding* (financiamento coletivo). Esta última demonstra bastante potencial às instituições graças ao crescimento do acesso a Internet, baixa relação custo/benefício e potencial de arrecadação. Campanhas de *crowdfunding* se utilizam de redes sociais e comunidades na Internet para arrecadar uma grande quantidade de pequenas doações as quais resultam em somas bastante significativas.

Este trabalho apresenta uma solução na forma de ferramenta online para *crowdfunding* social chamada Ajuda.Ai inspirada em parte pelo serviço Kiva [1], de código-fonte aberto e disponível para pequenas e médias organizações com custo mínimo a fim que elas possam ter um canal virtual, seguro, direto e cômodo para chegar a doadores e para os doadores para tornar o ato de doar a essas organizações tão simples quanto uma compra online.

**Palavras-chaves:** financiamento coletivo. impacto social. empreendimento social.

## ABSTRACT

Social entrepreneurs in Brazil historically have had great difficulties in finding the necessary funding to exert to the fully extent the activities they need to and so frequently have utilized of volunteer work campaigns and donations of food or used objects. Although the cost reductions that volunteer work and these kinds of donations are undeniable so is the necessity of money for costs like electricity, water and medical supplies. Campaigns of financial collection of smaller organizations are usually discreet, such as asking for the change of a purchase as a small donation.

In order to solve these problems there are solutions such as seeking investors, microcredits with community banks, media campaigns seeking donations and, more recently with the growth of this genre in Brazil, crowdfunding. The later shows great potential for collection. Crowdfunding campaigns make use of social networks and communities over the Internet to acquire a huge number of small donations which end up in significant sums.

This work presents a solution in the form an online tool for social crowdfunding called Ajuda.Ai inspired in part by the Kiva [1] service, with an open source code and available for small and medium organizations with minimal cost to enable a virtual, secure, direct and comfortable channel to reach donors and for donors to make giving to these organizations as easy as an online purchase.

**Key-words:** crowdfunding. social impact. social entrepreneurship.

## **LISTA DE ILUSTRAÇÕES**

Figura 1 – Diagrama UML de Casos de Uso do Ajuda.Ai . . . . .	18
Figura 2 – Captura de Tela da Página Inicial do Projeto Ajuda.Ai . . . . .	20
Figura 3 – Captura de Tela da Página Sobre do Projeto Ajuda.Ai . . . . .	21
Figura 4 – Captura de Tela da Página da Instituição AMA do Projeto Ajuda.Ai . . . . .	22
Figura 5 – Captura de Tela do Formulário de Doação . . . . .	23
Figura 6 – Captura de Tela da Página Inicial em Dispositivo Móvel . . . . .	24
Figura 7 – Captura de Tela de Página de Instituição em Dispositivo Móvel . . . . .	24
Figura 8 – Captura de Tela de Login do Ajuda.Ai . . . . .	25
Figura 9 – Captura do <i>Dashboard</i> de Doações, com dados de exemplo . . . . .	25
Figura 10 – Diagrama UML de Classes Resumido do Modelo do Ajuda.Ai . . . . .	28
Figura 11 – Arquitetura do Ajuda.Ai . . . . .	30
Figura 12 – Estrutura dos Processadores de Pagamento . . . . .	31
Figura 13 – Fluxo de um Pagamento via <i>Gateway</i> de Pagamento . . . . .	33
Figura 14 – Configuração de Notificação do <i>Gateway</i> de Pagamento MoIP . . . . .	34

## **LISTA DE ABREVIATURAS E SIGLAS**

API	Application Programming Interface
WWW	World Wide Web
REST	Representational State Transfer
HTML5	Hypertext Markup Language, versão 5
JSON	JavaScript Object Notation
CSS	Cascading Style Sheets
MVC	Model View Controller
CDI	Contexts and Dependency Injection
ORM	Object/Relational Mapping
JDBC	Java DataBase Connection
SGBD	Sistema de Gerenciamento de Banco de Dados
UUID	Universally Unique Identifier
CRUD	Create, Read, Update e Delete - Operações básicas de o SGBD

## SUMÁRIO

<b>1</b>	<b>INTRODUÇÃO . . . . .</b>	<b>11</b>
1.1	Contextualização . . . . .	11
1.2	Justificativa . . . . .	12
1.3	Objetivos . . . . .	14
1.3.1	Objetivo Geral . . . . .	14
1.3.2	Objetivos Específicos . . . . .	14
<b>2</b>	<b>FUNDAMENTAÇÃO TEÓRICA . . . . .</b>	<b>15</b>
2.1	<i>Crowdsourcing</i> . . . . .	15
2.2	Financiamento Coletivo: <i>Crowdfunding</i> . . . . .	15
<b>3</b>	<b>AJUDA.AI . . . . .</b>	<b>17</b>
3.1	Casos de Uso . . . . .	17
3.2	Características da Implementação . . . . .	18
3.3	Apresentação da Aplicação . . . . .	19
3.4	Organização do Projeto . . . . .	26
3.5	Arquitetura . . . . .	29
3.5.1	Processo Comum de Pagamentos Online . . . . .	32
3.6	API REST Web . . . . .	35
<b>4</b>	<b>CONCLUSÃO . . . . .</b>	<b>52</b>
4.1	Trabalhos Futuros . . . . .	52
	<b>REFERÊNCIAS . . . . .</b>	<b>54</b>
	<b>ANEXO A – ESPECIFICAÇÃO DE CASOS DE USO . . . . .</b>	<b>56</b>

## 1 INTRODUÇÃO

### 1.1 CONTEXTUALIZAÇÃO

Em sua essência, Financiamento Coletivo é parte de um conceito mais amplo chamado Contribuição Colaborativa (ou Colaboração Coletiva - do inglês *Crowdsourcing*). Em plataformas de Contribuição Coletiva utiliza-se do "coletivo" para se obter ideias, *feedback* e soluções para problemas através de uma chamada ampla via Internet e a custo zero, ou bastante reduzidos.

Sites como *The Mechanical Turk*<sup>1</sup> oferecem uma plataforma onde pessoas ou organizações podem colocar pedidos de micro-trabalhos, como votar na qualidade de traduções ou classificar vídeos em relação a seu conteúdo. Como recompensa a pessoa que faz essas tarefas recebe micro-pagamentos de, por exemplo, U\$ 0,15 (quinze centavos de dólar) por vídeo classificado. Outros como o *Kickstarter*<sup>2</sup> oferecem uma plataforma e rede social para arrecadar fundos para projetos normalmente relacionados a artes audiovisuais e atualmente é uma das mais prominentes plataformas de *crowdfunding*. Em Novembro de 2016, nove das dez mais bem financiadas campanhas de *crowdfunding* foram feitas via Kickstarter<sup>3</sup>, juntas essas campanhas arrecadaram mais de U\$ 102 milhões.

Além de casos consolidados como esses, grandes empresas da Internet como Google, Amazon e Facebook estão apostando cada vez mais em soluções de *Crowdsourcing* para inúmeras tarefas que necessitam de interação humana e para treinamento de Inteligências Artificiais. Um caso recente é o aplicativo Android Google *Crowdsourcing* [2] que dá aos usuários pequenas tarefas para auxiliar o aprendizado das inteligências artificiais por trás dos produtos Google. Tarefas como reconhecimento de escrita, interpretação de textos em fotografias, avaliação de traduções e várias outras estão disponíveis para serem resolvidas através do aplicativo e treinarem as várias inteligências artificiais por trás dos produtos da empresa.

Dessa nova forma de colaboração surgiu naturalmente um novo fenômeno: o Financiamento Colaborativo, ou *Crowdfunding*. Ambos utilizam o poder de várias pessoas engajadas e pequenas contribuições de um grande número de pessoas para atingirem seus objetivos [3]. Porém o fluxo de capital no *Crowdfunding* é o contrário do *Crowdsourcing*. Projetos que utilizam essa modalidade de financiamento pedem, através de plataformas online, pequenas contribuições financeiras de contribuidores individuais ou mesmo de investidores. O objetivo desses projetos normalmente é fazer algo mais pessoal como produção artística ou apoiar a produção de softwares como, por exemplo, jogos de forma mais independente. Assim é criado um novo modelo de investimento que circunver formas tradicionais de investimento como empréstimos junto a bancos, *venture capital* (capital de risco) e afins [4].

---

<sup>1</sup> <https://www.mturk.com>

<sup>2</sup> <https://www.kickstarter.com>

<sup>3</sup> Consultado em 31 jan. 2017 em <http://crowdfundingblog.com/most-successful-crowdfunding-projects/>

A primeira plataforma de *crowdfunding* a ter sucesso e ser responsável por iniciar de fato o mercado de *crowdfunding* nos Estados Unidos foi o *ArtistShare*<sup>4</sup> em 2003 [5]. De autoria de Brian Camelio, um músico e programador de Boston, o *ArtistShare* foi um *website* onde músicos podiam buscar doações de seus fãs para possibilitar aos artistas a gravação e produção digital de música. Eventualmente o site se tornou uma plataforma de financiamento coletivo para artistas audiovisuais, fotógrafos e músicos.

Seguindo essa tendência, em 2005, Matt e Jessica Flannery idealizaram e lançaram o que é considerada a primeira plataforma para *crowdfunding* social: Kiva<sup>5</sup>, uma agência de financiamento que utiliza *crowdfunding* para prover micro crédito a empreendedores pobres em países em desenvolvimento, mais notavelmente no Leste da África, Índia e Ásia Central. O Kiva é uma empresa sem fins lucrativos (*non-profit*) e plataforma tecnológica que liga pessoas que mesmo com pouco dinheiro, possuem a vontade de ajudar a pessoas que necessitam de recursos para ter um mínimo de qualidade de vida ou oportunidade [1]. A ideia começou quando os criadores do Kiva, durante uma viagem à África, conheceram o dono de uma peixaria na Etiópia que não tinha como melhorar seus lucros, pois não tinha dinheiro para comprar uma passagem de ônibus e dependia de um atravessador para comprar peixes.

No Brasil o mercado de *Crowdfunding* está ainda em seu começo, mas cresce a cada ano. Como exposto em [6], depois de passar mais de cinco anos procurando sem sucesso por investidores dispostos a financiar sua ideia, o arquiteto Márcio Cerqueira resolveu apostar em financiamento coletivo. A decisão foi fundamental para tirar do papel o Mola, espécie de *Lego* que ajuda estudantes de arquitetura a entender melhor as estruturas de edifícios. O objetivo inicial era de levantar R\$ 50 mil, mas o projeto teve mais de 1.500 apoiadores e acabou arrecadando R\$ 600 mil, mais de 10 vezes a meta inicial, algo que não teria obtido com grandes investidores. Já no projeto Catarse<sup>6</sup>, uma das maiores plataformas nacionais de *crowdfunding*, o volume arrecadado em 2016 foi de R\$ 16.2 milhões, um crescimento de 41% em relação a 2015 [7], e 134.827 pessoas apoiaram projetos na plataforma e desses 77.98% apoiaram pela primeira vez (105.150 pessoas).

Diante do cenário de crescimento da modalidade de *crowdfunding* no Brasil e da possibilidade de se utilizar dessa modalidade para financiamento de projetos sociais, uma ferramenta para fazer isso se faz não apenas oportuna como necessária.

## 1.2 JUSTIFICATIVA

A situação atual das ONGs<sup>7</sup>, como normalmente são chamadas organizações sem fins lucrativos, inclui dificuldades de várias ordens, mas as mais comuns, e que muitas vezes impedem

<sup>4</sup> <http://www.artistshare.com/>

<sup>5</sup> <http://www.kiva.org>

<sup>6</sup> <https://www.catarse.me>

<sup>7</sup> Organizações Não-Governamentais

a iniciativa de continuar ou até mesmo começar são dificuldades em identificar fontes de financiamento e captar recursos. Elas enfrentam críticas sobre o papel que ocupam na economia e na sociedade, sua relação com o governo e as empresas [8]. Além desses problemas, muitas vezes ONGs têm problemas para captação de recursos junto a pessoas físicas, pois muitos acreditam que trabalho voluntário é o suficiente. No Brasil, atualmente se vê um crescimento do trabalho voluntariado a um ponto que alguns autores [9] consideram isso como influência negativa a implementação de determinadas políticas sociais governamentais para diminuição da pobreza. Entende-se que se as próprias pessoas estão se mobilizando para resolver alguns problemas sociais, os mesmos se tornam menores e consequentemente menos recursos necessitem ser alocados para isso.

Apesar do crescimento das plataformas nacionais de *crowdfunding*, da quantidade de projetos e de apoiadores, essas plataformas têm taxas de uso da plataforma altas, comumente superiores a 10%. Parte deste percentual são custos do *gateway* de pagamentos utilizado pelo serviço como, por exemplo, o MoIP (*Money Over IP*) que cobra de 3,49% + R\$0,69 a 5,49% + R\$0,69 por transação<sup>8</sup>. Além desses custos, as plataformas nacionais não disponibilizam opção de escolha em relação a qual *gateway* de pagamento o projeto deseja utilizar. O Catarse, por exemplo, utiliza o *gateway* Pagar.me<sup>9</sup>. Outro grande serviço nacional, o Kickante utiliza MoIP como *gateway* de pagamento<sup>10</sup>. Em virtude dessa falta de escolha, o projeto deste trabalho trás a inovação da escolha do *gateway* de pagamento utilizado a fim de possibilitar a escolha do serviço com a melhor relação custo/benefício à instituição a qual pode ter, por exemplo, parceria com um determinado *gateway*.

Ante os custos apresentados, as dificuldades envolvidas em outras formas de financiamento e as facilidades e potenciais benefícios, este trabalho se propõe a criar uma plataforma de *crowdfunding* de código aberto chamada Ajuda.Ai que acarrete o mínimo custo possível para os projetos financiados pela plataforma, focada para organizações de pequeno porte. Para esse objetivo, um dos pontos principais e diferenciais da ferramenta é a possibilidade da escolha do projeto de qual *gateway* de pagamento será usado para processar as doações. Além disso, nenhum custo fora os custos embutidos pelo próprio *gateway* de pagamento será impresso às doações, dando assim uma maior margem ao projeto sob as doações recebidas.

Custos de manutenção e hospedagem do projeto serão custeados inicialmente através de capital pessoal e, com o crescimento do projeto, há a possibilidade de se utilizar a própria plataforma para captação de recursos para mantê-lo projeto ou, igualmente ao projeto Kiva, buscar investimento junto a investidores anjo ou filantropos.

<sup>8</sup> Consultado em 27/01/2017 em <https://moip.com.br/tarifas/>

<sup>9</sup> Consultado em 27/01/2017 em <http://crowdfunding.catarse.me/nossa-taxa>

<sup>10</sup> Consultado em 27/01/2017 em <https://www.kickante.com.br/termos/termos-de-uso>, item 9.1.2

## 1.3 OBJETIVOS

### 1.3.1 Objetivo Geral

O objetivo geral deste trabalho é prover um plataforma simples para facilitação de captação de recursos financeiros via Internet em modalidade *Crowdfunding*. A ferramenta Ajuda.Ai irá prover uma melhora em relação às disponíveis no mercado através da possibilidade de seleção e suporte a diferentes *gateways* de pagamento, isenção de taxas de uso da própria plataforma de *crowdfunding*, minimizando as taxas sobre as doações e provendo uma forma simples e cômoda para os doadores alcançarem as instituições de seus interesses.

### 1.3.2 Objetivos Específicos

- Construir uma ferramenta de código fonte aberto para captação de recursos financeiros em modalidade *crowdfunding*;
- Aprender sobre os fenômenos do *Crowdsourcing* e *Crowdfunding* e seus relacionamentos com empreendimentos sociais;
- Experimentar o desenvolvimento de uma aplicação web na plataforma Java que utilize de forma profissional recursos e ferramentas disponíveis pela e para a plataforma.

## RESUMO

Neste capítulo foi apresentada uma contextualização sobre o problema tratado neste trabalho e a justificativa de tal assunto, que pode ser resumida como sendo necessidade de uma alternativa moderna e simplificada para captação de recursos para ONGs. Ao final, foram detalhados os objetivos gerais e específicos do trabalho.

Os próximos capítulos estão organizados da seguinte forma:

- **Fundamentação Teórica:** Neste capítulo são apresentados todos os conceitos teóricos utilizados no desenvolvimento da solução proposta no presente trabalho;
- **Ajuda.Ai:** Capítulo dedicado a apresentação da solução implementada, detalhando funcionalidades, utilização e demais detalhes da implementação;
- **Conclusão:** Neste capítulo é feita a conclusão do trabalho dado seus objetivos propostos e são listados os trabalhos futuros para melhorar e/ou expandir a utilização da solução.

## 2 FUNDAMENTAÇÃO TEÓRICA

### 2.1 CROWDSOURCING

Antes de se falar de *Crowdfunding* é importante conhecer o movimento o qual originou. *Crowdsourcing* foi definido por Howe[10] em 2006 como o ato de uma companhia ou instituição pegar uma função desempenhada por seus funcionários e terceirizá-la para uma rede anônima, geralmente bastante grande, de pessoas na forma de uma chamada aberta. Isso pode tomar a forma de um sistema cooperativo, onde o trabalho é feito de forma colaborativa, mas também frequentemente o trabalho é executado por indivíduos. O pré-requisito crucial é o uso de um formato de chamada aberta e a grande rede de potenciais trabalhadores para atendê-lo.

Essa modalidade de terceirização não é recente. Por exemplo, em 1714, o Governo Britânico conduziu um concurso chamado Prêmio Longitude[11], que daria como recompensa 10 a 20 mil libras, dependendo da qualidade da solução, para quem resolvesse um dos maiores problemas da época: como determinar a longitude de uma embarcação em alto-mar. Um segundo exemplo notável é o de Hearn[12] que conta como Matthew Fontaine Maury, em 1848, distribuiu de forma gratuita 5000 cópias de seu catálogo de correntes marítimas e eólicas pedindo em troca que os marinheiros fizessem e entregassem seus diários de bordo ao retornar de suas jornadas.

Além desses projetos e, mais recentemente, com avanços nas técnicas de desenvolvimento de software, computação distribuída, processamento de dados e a popularização da Internet a partir da década de 1990 projetos como o GIMPS<sup>1</sup>, lançado em 1996, e o SETI@Home<sup>2</sup>, lançado em 1999. Esses projetos começaram a utilizar do poder de processamento latente nos computadores pessoais de inúmeros voluntários ao redor do planeta para executarem tarefas massivas que, sem isso, necessitariam de supercomputadores e bastante investimento.

Naturalmente esse aspecto da coletivização da ajuda evoluiu para além da realização de tarefas, grandes ou pequenas, e surgiram novas modalidades de *Crowdsourcing*. Dentre elas, a mais prominente e que mais cresce no mundo é o *Crowdfunding*, onde a tarefa em questão é a arrecadação de financiamento para um determinado fim, como descrito a seguir.

### 2.2 FINANCIAMENTO COLETIVO: CROWDFUNDING

Belleflamme, Lambert e Schwienbacher[4] definem *crowdfunding* como a utilização do *crowdsourcing* para arrecadar dinheiro para um empreendimento. Além disso, eles definem ainda que essa interação é feita via Internet, majoritariamente através de redes sociais, e podendo ou não haver um retorno ao investidor.

<sup>1</sup> Great Internet Mersenne Prime Search, Projeto para buscar Primos de Mersenne - <http://www.mersenne.org>

<sup>2</sup> Análise de ondas de rádio cósmicas em busca de vida extra terrestre - <https://setiathome.berkeley.edu>

Para Golán[13], o papel da comunidade é peça chave no funcionamento dessa modalidade de financiamento e a Internet indispensável para a formação dessas comunidades. Pessoas com interesses semelhantes facilmente se agregam e se organizam através de redes sociais, fóruns e outras ferramentas em comunidades. É através dessas comunidades que os potenciais apoiadores serão alcançados.

Campanhas de *crowdfunding* comumente se utilizam de duas formas para incentivar os apoiadores[14]. A primeira é através de uma espécie de venda ou pré-venda de algum produto, geralmente mostrado como uma recompensa pela ajuda ao projeto. A segunda é através da venda de participação na empresa a qual promove a campanha ou que será criada em decorrência da campanha.

Lehner[15] julga que a venda de participação, apesar de mais complexa e arriscada, pode ser benéfica no contexto de empreendimentos sociais, pois o nível de engajamento da comunidade pode servir como validação da ideia por trás do empreendimento. Além disso, um dos fatores motivacionais para apoiadores de projetos sociais é a participação dos mesmos nas ações daquela empresa ou organização.

Entretanto, dada a complexidade da criação de uma empresa sem fins lucrativos e regulamentação necessária para abertura de patrimônio, essa se torna uma opção inviável para as organizações não-governamentais que este trabalho propõe a atender. Tendo isso em consideração, fatores como validação do trabalho da ONG podem ser medidos pelo engajamento voluntário através da análise do volume e valor arrecadado estritamente na forma de doações que não acarretarão em um retorno direto, ou seja, nenhum produto ou serviço é oferecido ao apoiador do projeto. Dessa forma a motivação para ajuda ao projeto é a identificação do indivíduo com o trabalho proposto pela organização.

Utilizando o poder do *Crowdsourcing* e *Crowdfunding*, é proposta a ferramenta Ajuda.Ai para a arrecadação de investimento através dessa modalidade, com custos reduzidos e boa integração com plataformas de redes sociais provendo também um canal de comunicação com os doadores e comunidade interessada a fim de atender as necessidades e peculiaridades relacionadas a *crowdfunding* para empreendimentos sociais.

## RESUMO

Neste capítulo foi apresentada a base teórica utilizada para idealização da solução em software, que pode-se resumir em uma ferramenta para tornar a comunicação com uma comunidade e arrecadação de fundos da mesma.

No próximo capítulo será apresentada a ferramenta de código-fonte aberto que o trabalho propõe para atender a essas necessidades, de código fonte aberto e chamada Ajuda.Ai.

### 3 AJUDA.AI

O projeto Ajuda.Ai é um software de código-fonte aberto que visa ajudar instituições a arrecadar doações através da internet e aos usuários a doarem para instituições de forma simples, semelhante a adquirir um produto numa loja online, ter uma boa integração junto a plataformas de redes sociais e unir doadores e instituições. O projeto apresentado foi construído com uma interface para ser acessada pelo navegador através de uma aplicação web de página única (*Single Page Application*), além de contar com uma API que dá suporte a outros tipos de interface, e no lado do servidor uma aplicação Java em um servidor de aplicação RedHat WildFly 10.

O código-fonte da ferramenta está disponível sob licença MIT no endereço <https://github.com/g0dkar/ajuda-ai> e está publicada no endereço <https://ajuda.ai>. A API pode ser acessada através do *endpoint* <https://api.ajuda.ai/v1>. Apesar da ferramenta estar pronta para uso, por hora nenhuma instituição real está sendo contemplada, mas duas instituições reais encontram-se cadastradas apenas para testes. Elas serão as primeiras a serem contactadas para iniciarem o uso do Ajuda.Ai. A divulgação da ferramenta será feita inicialmente através de contato direto com as Instituições e através de redes sociais. Futuramente, publicidades digitais podem ser uma opção viável para ajudar a trazer mais doadores e novas instituições ao Ajuda.Ai.

#### 3.1 CASOS DE USO

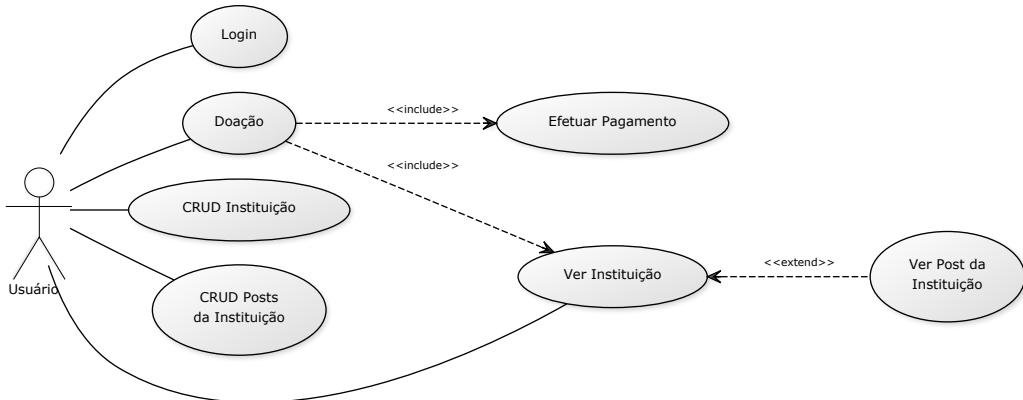
O levantamento de requisitos foi feito através da análise do mercado de *crowdfunding* nacional, do crescimento e popularização da prática dessa modalidade de investimento, tendências internacionais em relação ao mercado de empreendimentos sociais e leitura e análise da literatura exposta no capítulo 2. Os casos de uso relevantes resultantes do levantamento de requisitos foram:

- **Caso de Uso 01 - Doação:** O usuário se identifica com o trabalho de uma determinada instituição e quer fazer uma doação a mesma através da Internet;
- **Caso de Uso 02 - Comunicação com o Doador:** As instituições precisam de um canal para se comunicarem aos seus doadores e potenciais doadores;
- **Caso de Uso 03 - Acompanhamento das Doações:** É importante para a Instituição ter uma maneira de acompanhar a arrecadação de sua campanha e ter informações sobre quanto foi arrecadado e quando o dinheiro estará disponível;

As especificações dos casos de uso estão disponíveis no Anexo A - Especificação de Casos de Uso. Apesar de outros casos de uso existirem, por serem bastante simples e triviais, como *login*, *logout*, CRUD simples de entidades do projeto *model* (apresentado no item 3.4) eles não foram documentados. Esses casos de uso são inclusos no diagrama de Casos de Uso,

ilustrado na Figura 1. Como trabalho futuro, pretende-se implementar a Caso de Uso 03, pois atualmente os próprios *Gateways* disponibilizam ferramentas dessa natureza.

Figura 1 – Diagrama UML de Casos de Uso do Ajuda.Ai



Fonte: Elaborado pelo Autor

A seguir serão apresentados detalhes sobre a implementação do projeto, como tecnologias utilizadas, e em seguida uma apresentação visual da aplicação criada. Por fim, uma parte mais detalhada sobre partes importantes do sistema e sua API REST Web.

### 3.2 CARACTERÍSTICAS DA IMPLEMENTAÇÃO

O sistema utiliza a plataforma Java em sua versão 8. Para facilitar a configuração da aplicação e gestão de dependências se utilizou a ferramenta Apache Maven<sup>1</sup>, que provê um fácil acesso a uma grande quantidade de ferramentas e bibliotecas. Além da resolução de dependências, o Apache Maven facilita o processo de construção e montagem da aplicação, automatiza a produção da documentação da aplicação apartir do JavaDoc<sup>2</sup> e é considerado uma ferramenta padrão da indústria.

Como arcabouço para desenvolvimento de aplicação web se utilizou o Caelum VRaptor. O VRaptor é um arcabouço brasileiro MVC de código fonte aberto que traz alta produtividade para um desenvolvimento Java Web rápido e fácil com CDI<sup>3</sup>. Dentre as funcionalidades mais interessantes do VRaptor estão a integração completa e simples com o arcabouço JBoss Weld, implementação de referência da especificação JSR 365<sup>4</sup> e arcabouço utilizado pelo RedHat Wildfly 10, e o uso do estilo convenção sob configuração onde se promove que, seguindo convenções bem definidas, não é necessário arquivos de configuração.

<sup>1</sup> <https://maven.apache.org>

<sup>2</sup> Padrão de documentação Java que utiliza comentários com formatação para documentar a aplicação

<sup>3</sup> Retirado de <http://www.vraptor.org/pt/> em 17/02/2017

<sup>4</sup> Especificação Java para um sistema de Injeção de Dependências usando o Padrão de Projeto Inversão de Controle

Para persistência de dados foi utilizado o arcabouço JBoss Hibernate ORM. Como um arcabouço de mapeamento objeto-relacional o Hibernate ORM faz a persistência de dados em banco de dados relacionais via JDBC<sup>5</sup>. Como solução para banco de dados foi utilizado o SGBD MariaDB, banco de dados de código aberto que surgiu a partir do MySQL quando o mesmo foi comprado pela Oracle em 2003. Ele é mantido pelos desenvolvedores originais do MySQL e eles garantem que sempre será um projeto de código aberto. usuários notáveis incluem Wikipedia, WordPress.com e Google<sup>6</sup>.

Por fim, senhas são criptografadas no banco de dados se utilizando do algoritmo BCrypt. O BCrypt é uma função de *hashing* de senhas projetada em 1999 por Niels Provos e David Mazières baseada na cifra *Blowfish* que também incorporara um *salt*<sup>7</sup> para proteger as senhas contra ataques de tabelas arco-íris<sup>8</sup>. Nesse contexto, a maior vantagem do BCrypt é que esta é uma função cuja a quantidade de iterações pode ser configurada e assim, com o tempo, pode ser aumentada para torná-la mais lenta e resistente a ataques de força-bruta [16].

### 3.3 APRESENTAÇÃO DA APLICAÇÃO

Nesta seção será apresentada de forma ilustrada a solução desenvolvida neste trabalho. Parte do projeto visual, cores e detalhes estéticos são de autoria de Cândido Sales Gomes. A Figura 2 apresenta a página inicial do projeto Ajuda.Ai. Nela temos um painel rotativo informativo com âncoras para páginas informativas sobre o projeto, uma lista mesclada entre instituições mais recentemente cadastradas no projeto e outras escolhidas aleatoriamente e uma seção com um pequeno mapa de âncoras para todas as páginas informativas do projeto.

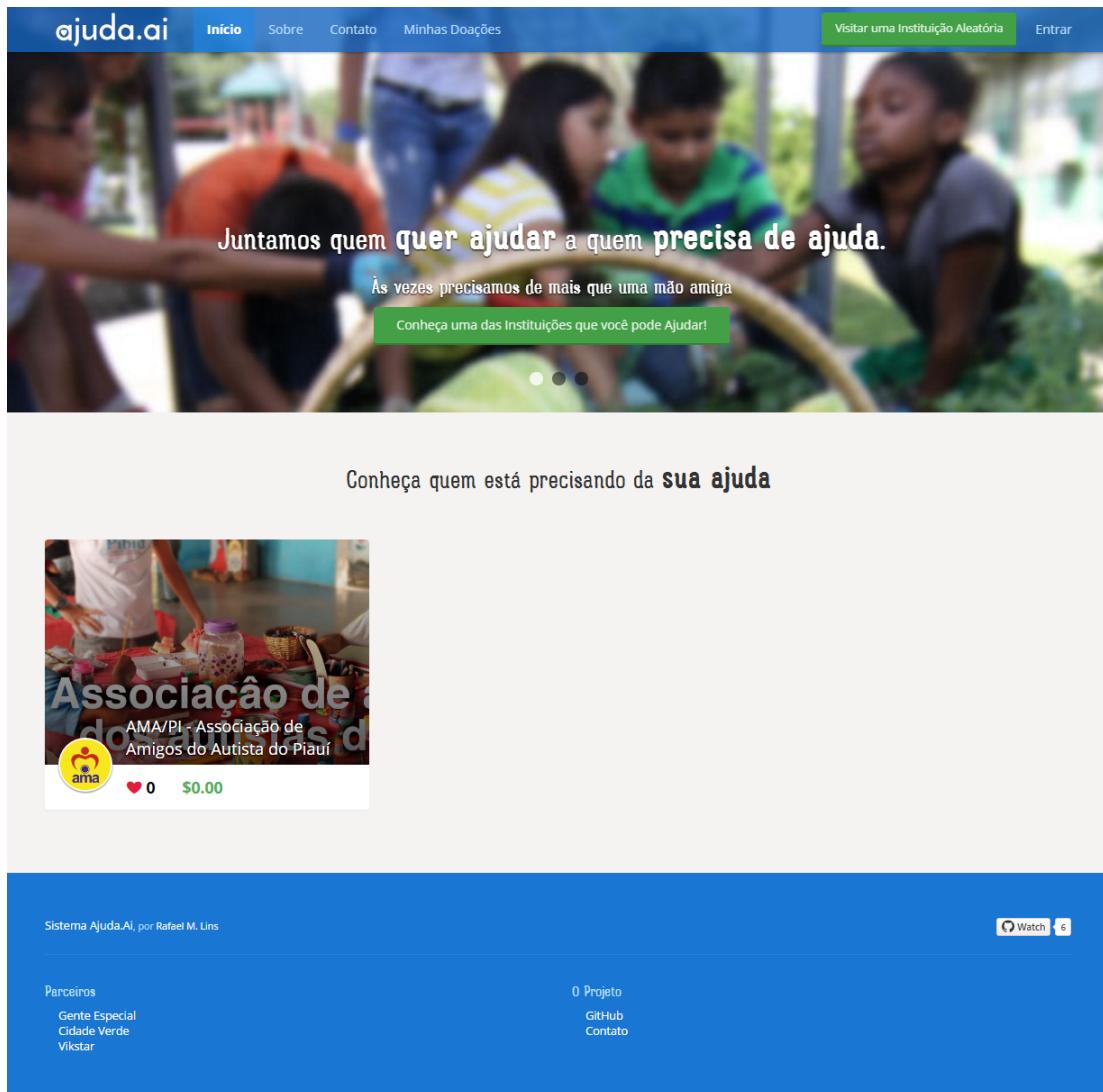
<sup>5</sup> Retirado de <http://hibernate.org/orm/> em 17/02/2017

<sup>6</sup> Retirado de <https://mariadb.org/about/> em 17/02/2017

<sup>7</sup> Bytes gerados de forma aleatória

<sup>8</sup> *Rainbow Table Attack*: Ataque que consiste em gerar um gigantesco banco de dados de *hashes* pré-processados onde é possível se buscar pelo texto simples de um determinado *hash* de um determinado algoritmo

Figura 2 – Captura de Tela da Página Inicial do Projeto Ajuda.Ai



Fonte: Elaborado pelo Autor

A estrutura das páginas segue em grande parte a mesma estrutura visual na Figura 2: uma barra de navegação de posicionamento fixo no topo da tela, um painel com uma grande figura e uma frase de cabeçalho, o conteúdo da página atual e o rodapé.

Na Figura 3 é apresentada uma página interna, especificamente a página “Sobre”, que fala em linhas gerais sobre o projeto. Esse mesmo modelo é utilizado para todas as páginas internas, representadas pela entidade `Page`, apresentada na seção 3.4.

Figura 3 – Captura de Tela da Página Sobre do Projeto Ajuda.Ai



Fonte: Elaborado pelo Autor

A seguir, na Figura 4 é apresentada uma página de Instituição. Elementos visuais como a imagem do painel e logomarca da instituição são personalizáveis através de configurações da interface de administração. Além disso, é apresentada ao navegante uma contagem atualizada de quantas doações a instituição recebeu e o valor total das doações. O projeto visual do Ajuda.Ai se utiliza uma técnica de comunicação chamada *Call-To-Action* onde elementos que invocam ações, como botões, têm seu texto escrito de forma imperativa a fim de incentivar quem o lê a tomar a ação. É possível observar um uso da técnica no botão “Quero Ajudar” na Figura 4, o qual leva o usuário a página onde é possível se fazer uma doação.

Figura 4 – Captura de Tela da Página da Instituição AMA do Projeto Ajuda.Ai

Fonte: Elaborado pelo Autor

Ao clicar no botão “Quero Ajudar” ou na aba “Fazer uma Doação”, o usuário é levado a página de doação, cujo conteúdo é exibido na Figura 5. Como forma de incentivo a doações maiores, uma singela forma de *gameification*<sup>9</sup> foi utilizada na forma de uma barra de progresso com marcadores que é preenchida de acordo com o valor da doação, inicialmente em R\$ 20,00. É dada a opção de anonimato através da proibição da exibição do nome do doador. O nome e e-mail serão gravados para a emissão da ordem de pagamento junto ao *Gateway*, mas se a publicação for proibida o nome não será exibido em lugar algum, nem mesmo para a instituição que recebeu a doação.

Por segurança, a página também inclui um teste CAPTCHA do serviço Google reCAPTCHA. O serviço reCAPTCHA é um sistema de caixa de diálogo para usuário baseado na interface do CAPTCHA, que pede para usuários digitarem palavras distorcidas exibidas na tela, para ajudar a digitalizar o texto de livros, enquanto protege *sites* de robôs tentando acessar áreas restritas. O serviço fornece, para os *sites* inscritos, imagens de palavras que o software de reconhecimento ótico de caracteres (OCR) não foi capaz de identificar as quais são apresentadas para humanos decifrarem como palavras CAPTCHAs, como parte do seu procedimento normal

<sup>9</sup> Técnica onde se introduz elementos de jogos a fim de otimizar a execução de certas ações

de validação. Depois eles retornam os resultados para o serviço reCAPTCHA, que envia esses resultados para a digitalização de seus projetos [17].

Figura 5 – Captura de Tela do Formulário de Doação

**AMA/PI - Associação de Amigos do Autista do Piauí**

Sua ajuda pode fazer a diferença

Sobre **Fazer uma Doação**

Qual o **tamanho** da sua Ajuda?

R\$  .00      Você é uma pessoa linda por ajudar! (● & ●)

Mínimo: \$5, Máximo: \$5000

\$10      \$25      \$50

**Opções**

Não publicar meu nome  
Marcando essa opção, o Ajuda.Ai não irá exibir seu nome e seu e-mail em lugar algum. Ainda assim precisamos de seu nome e e-mail para emitir e enviar seus recibos das doações. [Saiba Mais...](#)

Quero cobrir os custos operacionais  
\$1.85 (cartão de crédito)  
Adicionando esse valor, as taxas cobradas à Instituição/ONG pela Plataforma de Pagamento serão pagas por você. **Nenhum dinheiro vai para o Ajuda.Ai.** [Saiba Mais...](#)

**Seus Dados** Para emitir e enviar seus recibos

E-mail \*   
super.heroi@generosidade.com.br  
Vamos manter esse e-mail informado do andamento dessa doação.

Nome \*   
Como se chama essa maravilhosa pessoa?  
Precisamos de seu nome para emitir recibos das suas doações. Você receberá um lembrete em seu e-mail assim que o recibo estiver disponível.:)

**Você é uma Pessoa?**

I'm not a robot   
reCAPTCHA  
Privacy - Terms

Se não for: [Gort, Klaatu barada nikto!](#)

O último passo será na Plataforma de Pagamento

**Ok, continuar para lá... ❤**

Fonte: Elaborado pelo Autor

Essa interface com o usuário através de uma folha de estilos CSS adequada pode se adaptar ao dispositivo onde a página está sendo exibida. Nas Figuras 6 e 7 temos duas capturas de tela mostrando como a interface se adapta quando exibida em um telefone celular.

Figura 6 – Captura de Tela da Página Inicial em Dispositivo Móvel



Fonte: Elaborado pelo Autor

Figura 7 – Captura de Tela de Página de Instituição em Dispositivo Móvel



Fonte: Elaborado pelo Autor

Essa capacidade é uma das melhores práticas recomendadas pela indústria, pois melhora a gama de usuários que terão acesso adequado à página. Esse aspecto é importante para o Ajuda.Ai devido a sua proposta de ser um meio de acesso a uma grande comunidade de pessoas, através de redes sociais, e ao crescimento da quantidade de dispositivos móveis no Brasil e do acesso a Internet através destes dispositivos.

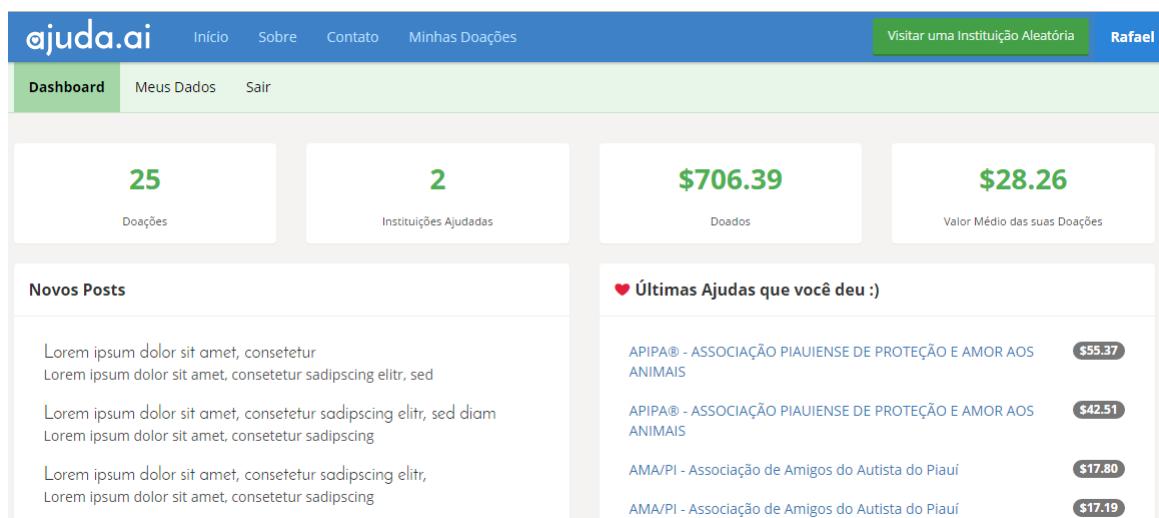
Na Figura 8 temos a tela de Login do usuário. O acesso a área administrativa pode ser feito através do nome de usuário escolhido ao se registrar ou e-mail. As senhas, como explicado na seção 3.2, são criptografadas com BCrypt e, como consequência, são conhecidas apenas pelos seus donos. Na mesma tela, caso o usuário tenha esquecido sua senha, é possível se utilizar o link “Esqueci minha senha” para que, usando o nome de usuário ou e-mail preenchido, um e-mail com instruções para reinicialização de senha será enviado para o usuário, a partir do qual ele poderá criar uma nova senha.

Figura 8 – Captura de Tela de Login do Ajuda.Ai



Fonte: Elaborado pelo Autor

Após o *login*, na Figura 9, um *dashboard* com informações relevantes são mostradas ao usuário: Suas últimas doações, total doado, quantidade de instituições ajudadas, média do valor das doações, últimas notícias das instituições ajudadas e quais as últimas doações feitas. Essa tela é também o que é exibido quando o usuário utiliza o link “Minhas Doações” do menu principal.

Figura 9 – Captura do *Dashboard* de Doações, com dados de exemplo

Fonte: Elaborado pelo Autor

### 3.4 ORGANIZAÇÃO DO PROJETO

O projeto está organizado numa estrutura de Projeto Pai e Projetos Filhos do Apache Maven. Essa organização facilita na manutenção e montagem da aplicação no momento da distribuição ou *deployment* da mesma. Os projetos filhos são:

- **util**: Classes utilitárias usado pelos outros projetos. As duas classes mais utilizadas são `StringUtils`, com utilitários para comparação e processamento de Strings, e `SendMail`, que utiliza a API `javax.mail` para enviar e-mails como HTML;
- **model**: Contém o modelo de dados do Ajuda.Ai, apresentado na Figura 10;
- **persistence-sql**: Provê classes que implementam a persistência do projeto model em um banco de dados SQL através do JBoss Hibernate;
- **backend**: Servidor de Back-end do Ajuda.Ai. Aplica as regras de negócio, organiza pagamentos e atende requisições dos usuários;
- **frontend**: Aplicação de Página Única do Front-end do Ajuda.Ai. Uma interface web simples e agradável para utilização do sistema;
- **payment-api**: Projeto pai para as implementações de pagamentos dos diferentes *Gateways* de pagamento;
- **payment-impl-moip**: Implementação de payment-api para o *gateway* MoIP;
- **payment-impl-pagseguro**: Implementação de payment-api para o *gateway* PagSeguro;

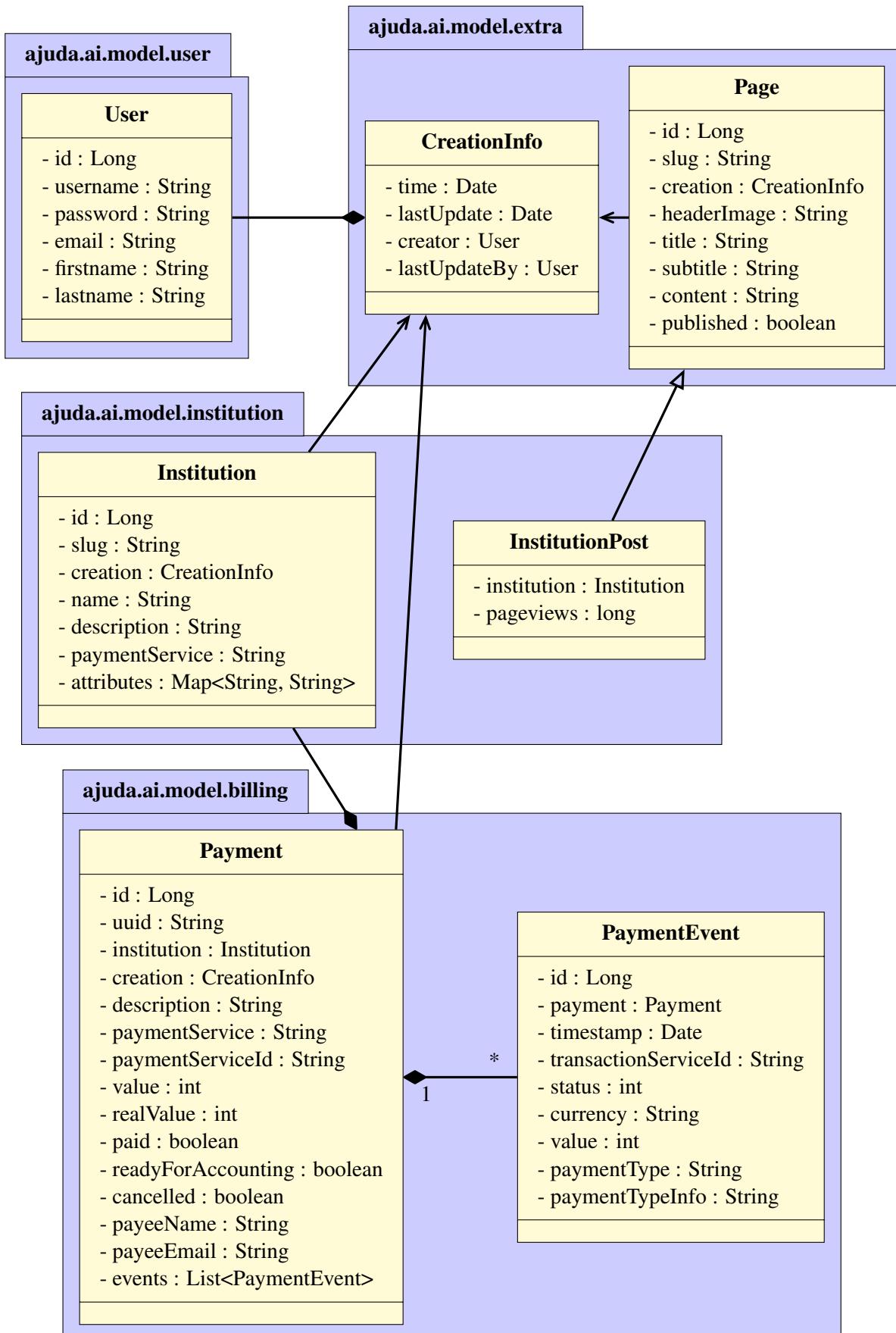
O modelo de dados da aplicação é apresentado na Figura 10, onde é apresentado um diagrama UML de classes do projeto *model*. Estas são as principais entidades do sistema e seus relacionamentos:

- **User**: Usuário simples. Todos os usuários podem ter Instituições e só poderão alterar informações vinculadas a ele mesmo;
- **CreateInfo**: Entidade embarcável<sup>10</sup> que guarda quando uma entidade foi criada, que usuário a criou, quando foi feita a última alteração na entidade e qual usuário fez a alteração. Caso o usuário que criou ou alterou seja nulo se considera que foi uma ação tomada pelo próprio Sistema;
- **Page**: Uma página de conteúdo livre do Ajuda.Ai. Representa uma página web cujo conteúdo mude com pouca frequência;
- **Institution**: Representa uma Instituição criada por um Usuário;
- **InstitutionPost**: Uma página de conteúdo livre vinculada a uma Instituição. Estende Page;

<sup>10</sup> Uma Entidade Embarcável é uma classe que, no esquema objeto-relacional da plataforma Java, não tem uma tabela no SGBD própria e sim faz parte da tabela das classes que se associam a esta classe.

- **Payment:** Representa uma ordem de pagamento feita pelo Sistema a um *Gateway* de Pagamento em nome de um Usuário;
- **PaymentEvent:** Representa uma mudança de estado de um pagamento como informado pelo *Gateway* de Pagamento;

Figura 10 – Diagrama UML de Classes Resumido do Modelo do Ajuda.Ai



Fonte: Elaborado pelo Autor

As principais classes do sistema são `Institution`, `Payment` e `PaymentEvent`. A primeira representa as Instituições cadastradas no sistema, as quais receberão doações de usuários, representados por `User`. Essas doações são representadas internamente como objetos `Payment` os quais abstraem de forma unificada pagamentos feitos através de diferentes *Gateways* de Pagamento. Por fim, o andamento do pagamento junto ao *Gateway* é informado através de notificações, como explicado em 3.5.1, e essas notificações são então abstruídas como objetos `PaymentEvent` vinculados a um `Payment`.

### 3.5 ARQUITETURA

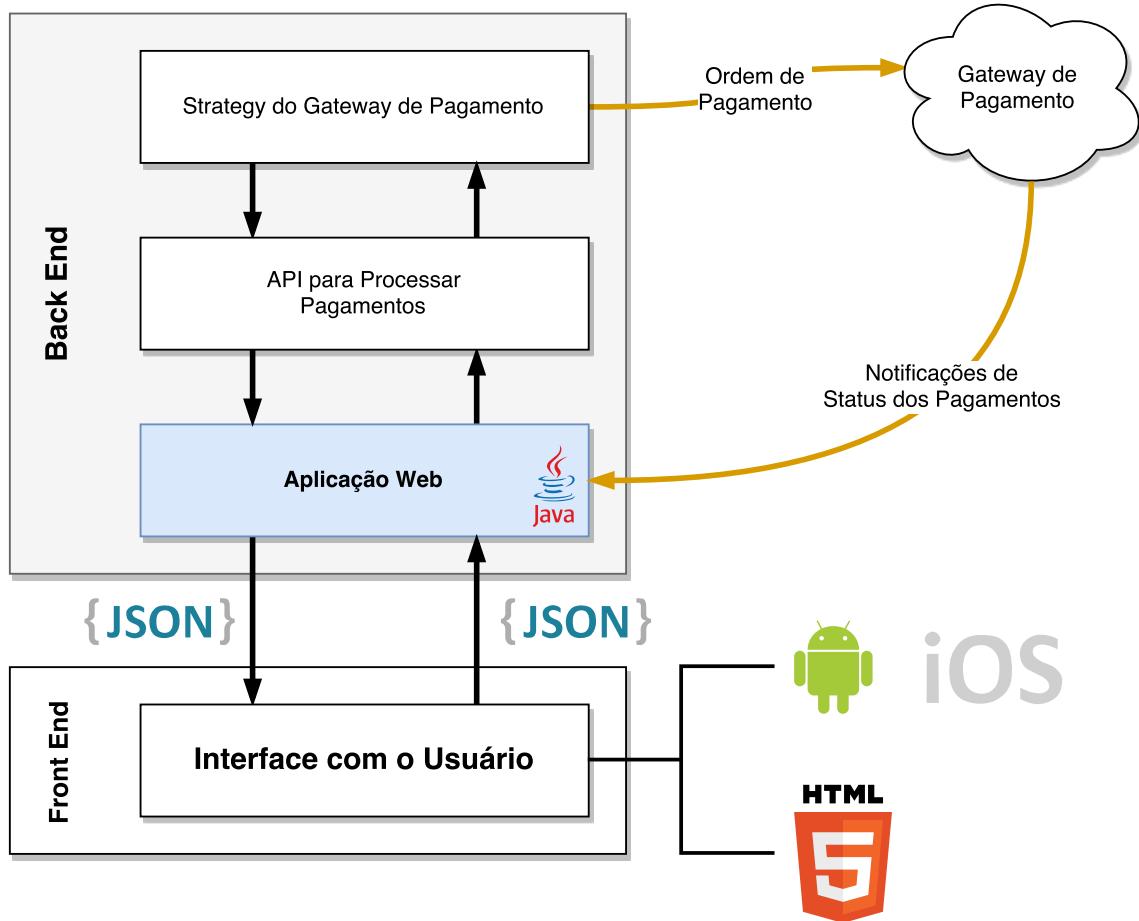
Como arquitetura do sistema se escolheu uma arquitetura REST Web onde o *Back End* é totalmente separado do *Front End*. Dessa forma, o *Back End* preocupa-se apenas em atender requisições menores e que envolvem dados e regra de negócio e o *Front End* preocupa-se apenas com a exibição de informações e a interação com o usuário. Assim, permite-se que seja feita uma separação completa entre *Front End* e *Back End* e a interface com o usuário pode ser feita se utilizando um navegador e HTML5 ou um aplicativo para plataformas móveis como Google Android e Apple iOS.

A Figura 11 ilustra a arquitetura do Ajuda.Ai, que utiliza uma arquitetura REST Web, arquitetura recomendada para aplicações web pela W3C<sup>11</sup> em [18]. Nessa especificação, REST Web é definido como um subconjunto da WWW, baseado em HTTP, onde o servidor provê uma interface e semântica uniforme para as aplicações, sem impor uma interface específica. Além disso, todas as ações são tomadas através do que se chama de troca de representações de estado, ou seja, a entrada são objetos que representam uma requisição e a saída são objetos que representam uma resposta.

---

<sup>11</sup> World Wide Web Consortium, consórcio de empresas e profissionais que buscam padronizar a WWW

Figura 11 – Arquitetura do Ajuda.Ai

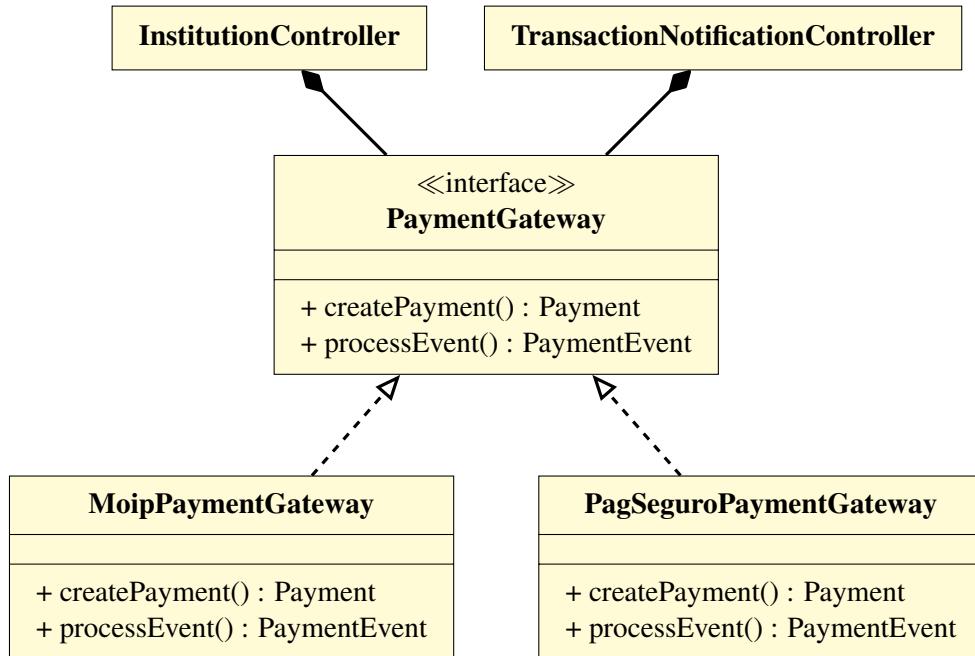


Fonte: Elaborado pelo Autor

Requisições são feitas ao sistema de duas origens: A interface com o usuário e os *gateways* de pagamento. As requisições feitas pela interface são padronizadas como JSON a fim de facilitar a criação de aplicações modernas e diminuir o tráfego de dados. Já requisições feitas pelos *gateways* são suportadas em XML, JSON, *x-www-form-urlencoded* (formulário HTML padrão) e *form-data* (formulário HTML com dados longos, definido pelo RFC2388).

Um dos diferenciais do Ajuda.Ai é o suporte a diferentes *gateways* de pagamento. Isso dá flexibilidade a instituição para usar o *gateway* mais adequado a mesma. Para implementação foi utilizado o Padrão de Projeto de Software *Strategy*, como ilustrado na Figura 12. Este Padrão de Projeto é definido por uma família de algoritmos encapsulados e que podem ser trocados dentro da família de objetos [19].

Figura 12 – Estrutura dos Processadores de Pagamento



Fonte: Elaborado pelo Autor

Cada implementação de *gateway* de pagamento suportado implementa a interface PaymentGateway. Esta interface tem dois métodos:

- **createPayment ()**: Cria uma ordem de pagamento junto ao *gateway* de pagamento. Esse método tem liberdade de redirecionar o usuário para o sistema do *gateway* (MoIP e a maioria dos gateways nacionais) ou enviar algum pedaço de informação, como JSON, como resposta ao usuário (PayPal + JavaScript para pagamento na própria página). Este método deve retornar um objeto Payment para ser persistido a fim de acompanhar o andamento do pagamento;
- **processPayment ()**: Gateways de pagamento enviam requisições de volta ao sistema informando sobre o status de um determinado pagamento feito na plataforma deles. Esse método é encarregado de processar essas mudanças de estado no pagamento. Este método deve retornar um objeto PaymentEvent que indica de forma padronizada o que aconteceu com este pagamento.

Ao criar uma ordem de pagamento através de `createPayment ()` a classe que a criou retorna um objeto Payment para ser persistido. Esse objeto, além de padronizar as informações relevantes dos pagamentos, guarda consigo a estratégia (serviço de pagamento) utilizada para criar a ordem de pagamento. Essa informação é usada futuramente quando o sistema recebe uma notificação de estado de pagamento.

Uma vantagem da utilização do padrão *Strategy* é que diferentes usos de um mesmo *gateway* podem ser facilmente implementados. Por exemplo, o *gateway* MoIP oferece várias

formas para criar ordens de pagamento. Três delas são interessantes para o que o Ajuda.Ai propõe:

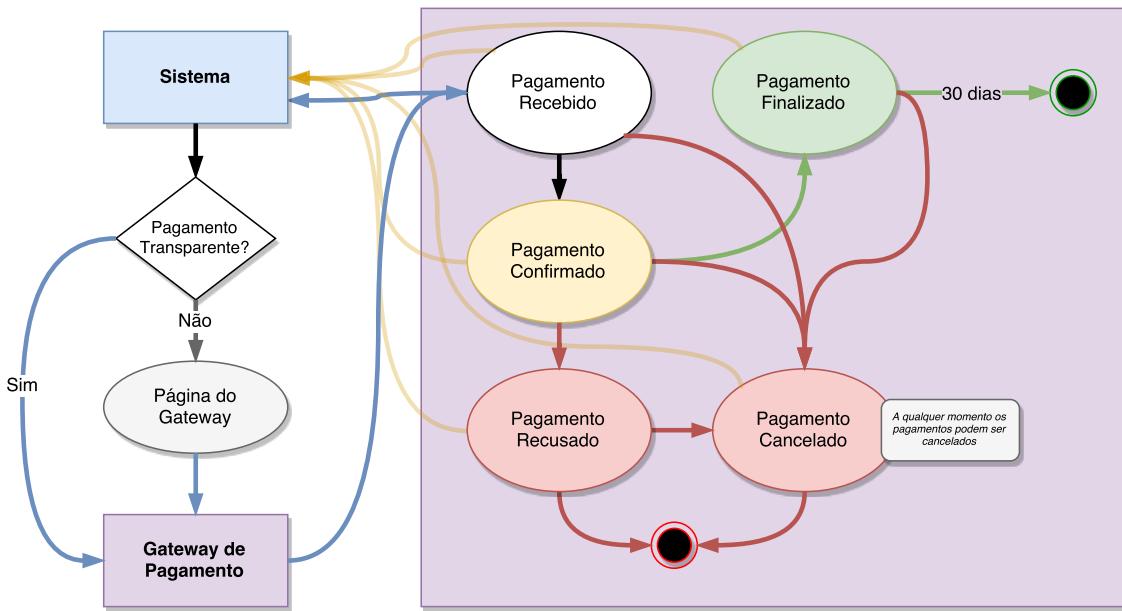
- **Ordem de pagamento via e-mail:** O MoIP envia um e-mail para o usuário onde, na conveniência do mesmo, ele tem 30 dias para efetuar o pagamento da forma como preferir;
- **Check-out Clássico:** O usuário é redirecionado para o *Gateway* onde ele selecionará forma de pagamento e fará todo o processo de pagamento antes de voltar ao Ajuda.Ai — Essa forma está disponível em todos os *gateways* e é a forma preferencial de se fazer a interação de pagamento junto ao usuário;
- **Check-out Transparente:** O usuário preenche tudo no próprio sistema o qual junto ao *gateway* e através de *webservices* faz uma ordem de pagamento e processamento de pagamento — Esta forma está disponível em quase todos os *gateways*.

O Ajuda.Ai se utiliza da segunda forma demonstrada, o *Check-Out* Clássico, pois é suportada por todos os *gateways* e recomendada pelos mesmos por prover um ambiente seguro e que os usuários demonstram maior confiança ao partilhar dados sigilosos e pessoais como números de cartão de crédito e CPF. Esse processo de pagamento será detalhado na próxima subseção.

### 3.5.1 Processo Comum de Pagamentos Online

A ilustração da Figura 13 mostra o fluxo de criação e processamento de pagamento comum a maioria dos *gateways* atualmente no mercado. Todo o processo acontece entre duas entidades que são o Sistema, que é o sistema que irá gerar as ordens de pagamento e ser notificado do andamento dos pagamentos, e o próprio *Gateway* de Pagamento. O processo começa com a geração de uma ordem de pagamento junto ao *gateway*, a qual pode ser feita de duas maneiras:

- **Pagamento Normal:** Nessa modalidade, o Sistema faz um HTTP POST com dados sobre o que está sendo comprado ao *Gateway* e o usuário é redirecionado até uma página do próprio *Gateway* de pagamento. Esse é o modo recomendado pelos *Gateways*, pois os *Gateways* podem servir a página de forma segura, protegida através de certificados digitais confiáveis, com uma interface de usuário adequada. Segundo os *Gateways*, os clientes ficam mais propensos a concluir a compra quando o pagamento utiliza esse formato, pois os mesmos confiam na marca do *Gateway* e na segurança do ambiente;
- **Pagamento Transparente:** Nessa modalidade, todos os dados do pagamento são preenchidos em uma interface provida pelo próprio Sistema e enviados de forma segura ao *Gateway* de Pagamento, o qual imediatamente cria e inicia o processamento do pagamento da uma ordem de pagamento.

Figura 13 – Fluxo de um Pagamento via *Gateway* de Pagamento

Fonte: Elaborado pelo Autor

O Ajuda.AI faz uso da primeira maneira, o Pagamento Normal, onde o usuário é redirecionado ao *Gateway*. Após a criação do pedido de pagamento e o preenchimento das informações necessárias, como dados pessoais, CPF e outros, o *Gateway* cria um pagamento em seu sistema, e a partir desse ponto começa a máquina de estados do *Gateway*. A máquina apresentada é uma simplificação e normalização do que ocorre na maioria dos *Gateways*.

Na ilustração da Figura 13 as setas de cor azul representam o fluxo como visto pelo usuário do Sistema. O usuário preenche algumas informações no próprio Sistema ou no *Gateway* e, ao enviar, é feita a ordem de pagamento e o usuário é redirecionado de volta ao Sistema em uma página de “Obrigado”. Esta é uma página do Sistema para onde o *Gateway* irá redirecionar o usuário ao final do fluxo de criação da ordem de pagamento.

Ao criar uma ordem de pagamento, inicia-se uma máquina de estados do pagamento que muda de acordo com o andamento do pagamento ou ações do usuário. Durante os estados na máquina de estados do *Gateway* cada alteração de estado resulta em uma notificação ao sistema, representadas pelas setas na cor laranja. Esse comportamento é típico do padrão de projetos *Event Notifier*, também conhecido como *Publish-Subscribe* ou *Pub-Sub*. Esse padrão de projeto permite que dois sistemas comuniquem eventos entre si sem necessariamente ter conhecimento um do outro [20].

Como parte da implementação deste padrão de projeto, os *gateways* de pagamento provêm alguma maneira de configurar quais URL do Sistema deverá ser chamada para receber as notificações geradas pelas trocas de estado. Na ilustração da Figura 14 está a configuração desta opção no sistema do *gateway* MoIP.

Figura 14 – Configuração de Notificação do *Gateway* de Pagamento MoIP

**Notificação de Alteração de Status de Pagamento**

\* Receber notificação instantânea de transação

\* URL de notificação:  
`https://api.ajuda.ai/exemplo/transaction-notification`  
(<http://...> ou <https://...>)

Clique nos links abaixo para ver como integrar seu sistema.

[\[manual em HTML\]](#) [\[manual em PDF\]](#)

**confirmar alterações** **cancelar**

Fonte: Elaborado pelo Autor

A cada alteração de estado do pagamento uma requisição específica a URL configurada será feita. Para facilitar a configuração dos vários diferentes *gateways* no mercado a URL do Ajuda.Ai a ser configurada neles é padronizada, mudando apenas um parâmetro que representa qual *gateway* fará requisições a URL. Na ilustração da Figura 14 temos a URL referente ao MoIP.

Uma vez criado um pagamento junto ao *gateway* ele deve ser confirmado. Esse estado é mais evidente quando o pagamento é feito por Boleto Bancário, pois o mesmo demora até 3 dias úteis para ser processado pelo banco e o retorno dado ao *gateway*. Em pagamentos via cartão de crédito a confirmação é simplesmente o recebimento da requisição de crédito através dos sistemas da mesma, o que quase sempre é imediato.

Na confirmação o *gateway* é informado do sucesso ou falha na execução do pagamento. Caso o pagamento tenha sido feito sem problemas, ele vai para o estado Finalizado, vai para Recusado. Como explicitado na Figura 13, a praticamente qualquer momento do processo o usuário poderá Cancelar o pagamento. Apenas após 30 dias no estado Finalizado, o cancelamento da operação fica indisponível, entretanto este prazo pode mudar de acordo com o *gateway*. Vale salientar também, que apenas após este prazo o dinheiro estará disponível para resgate pela instituição.

Cada mudança de estado resulta em uma chamada a URL configurada juntamente a um pacote de informações sobre o estado atual do pagamento. Cada *Gateway* envia informações distintas, mas bastante semelhantes. A mais importante delas é o que é chamado de ID do Cliente que é um número especificado pelo Sistema e que será vinculado aquele pagamento. Este campo deve ser utilizado pelo Sistema para identificar internamente o que está sendo pago. O Ajuda.Ai utiliza este campo para manter o ID do objeto Payment gerado no início do processo a fim de

manter o estado do pagamento e vincular a instituição correspondente.

### 3.6 API REST WEB

Para facilitar o desenvolvimento de diferentes interfaces com o usuário e demais consumidores da API e prevenir dúvidas quanto a mesma e seu funcionamento a API REST Web do *backend* do projeto Ajuda.Ai foi documentada. Esses são os métodos da API que estão disponíveis para todos os clientes, porém alguns necessitam de login. Os métodos que não necessitam de login são classificados como métodos públicos.

Apesar da importância de uma boa documentação ainda não há um padrão estabelecido para a forma de documentar APIs REST Web. Por esse motivo este trabalho se utiliza do formato proposto por Irene Ros[21], diretora da Bocoup, empresa de consultoria referência em desenvolvimento REST Web e produtora de diversas ferramentas de código fonte aberto para se trabalhar com REST Web e *Front-end*. A seguir está a API REST Web disponibilizada pelo Ajuda.Ai. Todos os parâmetros são obrigatórios, a menos que especificado que seja opcional. Parâmetros que fazem parte da própria URL são indicados com um sinal de dois-pontos no início de seu nome:

### Quadro 1: Autenticação do Usuário

Retorna informações sobre o usuário logado. Usado para, além de pegar informações sobre o usuário logado, manter a Sessão de autenticação ativa.

- **URL**

- /auth/login

- **Método**

- POST

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- username=String Nome de usuário ou E-mail do Usuário que quer se autenticar
  - password=String Senha do Usuário

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON de um objeto User apenas com os campos id, username, email, firstname e lastname do Usuário logado.

- **Resposta de Erro**

- **Código:** 200 OK

**Conteúdo:** Array com os erros de autenticação

**Motivo:** Usuário ou senha incorretos.

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método POST.

## Quadro 2: Dados do Usuário Logado

Retorna informações sobre o usuário logado. Usado para, além de pegar informações sobre o usuário logado, manter a Sessão de autenticação ativa. A sessão de autenticação dura por 30 minutos após a última requisição a API.

- **URL**

- /profile/me

- **Método**

- GET

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON do objeto User do usuário logado, apenas com os campos id, username, email, firstname e lastname.

- **Resposta de Erro**

- **Código:** 401 UNAUTHORIZED

**Conteúdo:** Nenhum

**Motivo:** Não há um Usuário logado

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

### Quadro 3: Dados de um Usuário

Retorna informações sobre o usuário especificado a partir de seu nome de usuário.

- **URL**

- /profile/:username

- **Método**

- GET

- **Parâmetros**

- username=String - (**opcional se id for especificado**) Nome único de usuário

- **Parâmetros de Dados**

- id=Integer - Identificador do usuário

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON do objeto User do usuário encontrado, apenas com os campos id, username, firstname e lastname.

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Usuário não existe

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

**Quadro 4: Dados de uma Instituição**

Retorna todos os dados de uma Instituição.

- **URL**

- /institution/:slug

- **Método**

- GET

- **Parâmetros**

- slug=String - Nome da URL da Instituição

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON com os dados da Instituição

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Instituição não existe

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

**Quadro 5: Dados de uma Instituição escolhida Aleatoriamente**

Retorna todos os dados de uma Instituição escolhida aleatoriamente.

- **URL**

- /institution/random

- **Método**

- GET

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON com os dados de uma Instituição

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Não há nenhuma instituição cadastrada no banco de dados

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

**Quadro 6: Lista de Instituições escolhidas Aleatoriamente**

Retorna uma lista de até 12 Instituições escolhidas aleatoriamente. Útil para exibir Instituições na Página Inicial.

- **URL**

- /institution/random-list

- **Método**

- GET

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON com um Array com dados de Instituições escolhidas aleatoriamente. Máximo de 12 instituições.

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Não há nenhuma instituição cadastrada no banco de dados

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

### Quadro 7: Estatísticas de Doações

Retorna quantas doações a Instituição recebeu e o valor total das mesmas.

- **URL**

- /institution/:slug/donation-stats

- **Método**

- GET

- **Parâmetros**

- slug=String - Nome da URL da Instituição

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON no seguinte formato:

```
{ count: Integer, value: Integer } - O valor deve ser dividido por 100 para ser exibido adequadamente.
```

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Instituição não existe

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

#### Quadro 8: Doação - Passo 01: Registrar o Pagamento

Inicia o processo de doação gravando os dados da mesma no Ajuda.Ai. Posteriormente o usuário deverá ser redirecionando para o *gateway* de pagamento vinculado a Instituição através do método `/pagar/:id`.

- **URL**

- `/institution/:slug/doar`

- **Método**

- POST

- **Parâmetros**

- `slug=String` - Nome da URL da Instituição

- **Parâmetros de Dados**

- `value=Integer` - Valor da Doação como um Inteiro (10000 = \$100,00)
  - `name=String` - (**opcional se feito o login**) Nome Completo do Doador
  - `email=String` - (**opcional se feito o login**) E-mail do Doador
  - `addcosts:Boolean` - Adicionar custos operacionais do Gateway ao valor da doação?
  - `addcoststype=Integer` - Tipo de pagamento que será utilizado (para cálculo do custo operacional)

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** Objeto Payment com os dados do pagamento criado

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Instituição não existe

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método POST.

**Quadro 9: Doação - Passo 02: Redirecionamento ao *Gateway***

Continua o processo de doação redirecionando o usuário para o *gateway* de pagamento vinculado ao Payment especificado.

- **URL**

- /pagar/:id

- **Método**

- GET

- **Parâmetros**

- id=UUID - Identificador do pagamento (UUID)

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 302 MOVED TEMPORARILY

**Conteúdo:** Nenhum

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Pagamento não existe

- **Código:** 409 CONFLICT

**Conteúdo:** Nenhum

**Motivo:** Notificações do *Gateway* informam que este pagamento já está ou já foi processado.

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

**Quadro 10: Posts de uma Instituição**

Retorna uma lista de tamanho limitado com os *posts* de uma Instituição.

- **URL**

- /institution/:slug/posts

- **Método**

- GET

- **Parâmetros**

- slug=String - Nome de URL da Instituição

- **Parâmetros de Dados**

- size=Integer - **(opcional)** Quantidade de Itens na lista. Padrão: 10.  
Máximo: 50.
  - offset=Integer - **(opcional)** Primeiro item da lista. Padrão: 0.

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON no seguinte formato:

```
{ size: Integer, total: Integer, offset: Integer,  
  items: [InstitutionPost] }
```

Nota: os *posts* terão apenas os campos id, slug, titulo, subtitle, criador e data e hora de criação.

- **Resposta de Erro**

- **Código:** 404 NOT FOUND

**Conteúdo:** Nenhum

**Motivo:** Instituição não existe

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método GET.

### Quadro 11: Post Único de uma Instituição

Retorna um único *post* de uma Instituição.

- **URL**

- /institution/:slug/:post-slug

- **Método**

- GET

- **Parâmetros**

- slug=String - Nome da URL da Instituição
  - post-slug=String - Nome da URL específico do *post* da Instituição

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK  
**Conteúdo:** JSON com os dados do *post* da Instituição

- **Resposta de Erro**

- **Código:** 404 NOT FOUND  
**Conteúdo:** Nenhum  
**Motivo:** Instituição ou *Post* não existem
  - **Código:** 405 METHOD NOT ALLOWED  
**Conteúdo:** Nenhum  
**Motivo:** A chamada HTTP não utilizou o método GET.

### Quadro 12: Dados sobre Doações do Usuário Logado

Retorna algumas informações sobre as doações feitas pelo usuário logado.

\* Este método requer quer o usuário tenha feito login.

- **URL**

- /profile/dashboard-data

- **Método**

- GET

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON no seguinte formato: { donations: Integer, institutions: Integer, value: Integer, meanValue: Integer, posts: [InstitutionPost], payments: [Payment] }

- **Descrição dos campos:**

- \* donations: Quantidade de doações feitas pelo usuário;
    - \* institutions: Quantidade de instituições únicas contempladas pelas doações;
    - \* value: Somatório dos valores das doações;
    - \* meanValue: Média dos valores das doações;
    - \* posts: Entre 0 e 10 dos *posts* mais recentes das instituições contempladas pelas doações (“Últimas Notícias”);
    - \* payments: Entre 0 e 10 das últimas doações feitas pelo usuário.

- **Resposta de Erro**

- **Código:** 401 UNAUTHORIZED

**Conteúdo:** Nenhum

**Motivo:** Não há um Usuário logado

### Quadro 13: Alterar Informações do usuário Logado

Altera informações do usuário logado como nome, e-mail, senha, etc.

\* Este método requer quer o usuário tenha feito login.

- **URL**

- /profile/save

- **Método**

- POST

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON do objeto User que foi alterado, apenas com os campos id, username, email, firstname, lastname.

- **Resposta de Erro**

- **Código:** 401 UNAUTHORIZED

**Conteúdo:** Nenhum

**Motivo:** Não há um Usuário logado

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método POST.

**Quadro 14: Dados sobre as Instituições pertencentes ao Usuário**

Retorna algumas informações sobre as doações feitas a todas as Instituições pertencentes ao usuário logado.

\* Este método requer que o usuário tenha feito login.

- **URL**

- /institution/dashboard-data

- **Método**

- GET

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- Nenhum Parâmetro

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON no seguinte formato: { donations: Integer, value: Integer, helpers: Integer, maxValue: Integer, meanValue: Integer, institutionCount: Integer, institutions: [Institution] }

- **Descrição dos campos:**

- \* donations: Quantidade de doações feitas às Instituições;
    - \* value: Somatório dos valores das doações;
    - \* helpers: Quantidade de usuários únicos que fizeram doações às Instituições;
    - \* maxValue: Maior valor doado;
    - \* meanValue: Média dos valores das doações;
    - \* institutionCount: Quantas Instituições pertencem ao usuário;
    - \* institutions: Entre 0 e 50 Instituições pertencentes ao usuário.

- **Resposta de Erro**

- **Código:** 401 UNAUTHORIZED

**Conteúdo:** Nenhum

**Motivo:** Não há um Usuário logado

**Quadro 15: Altera dados de uma Instituição pertencente ao Usuário**

Altera informações sobre uma Instituição pertencente ao Usuário.

\* Este método requer quer o usuário tenha feito login.

- **URL**

- /institution/save

- **Método**

- POST

- **Parâmetros**

- Nenhum Parâmetro

- **Parâmetros de Dados**

- JSON de um objeto Institution

- **Resposta de Sucesso**

- **Código:** 200 OK

**Conteúdo:** JSON do objeto Institution após ser alterado na persistência de dados.

- **Resposta de Erro**

- **Código:** 200 OK

**Conteúdo:** Array com erros de validação

**Motivo:** Um ou mais campos falhou na validação de dados.

- **Código:** 401 UNAUTHORIZED

**Conteúdo:** Nenhum

**Motivo:** Não há um Usuário logado

- **Código:** 405 METHOD NOT ALLOWED

**Conteúdo:** Nenhum

**Motivo:** A chamada HTTP não utilizou o método POST.

## **RESUMO**

Neste capítulo foi apresentada a solução Ajuda.Ai para *crowdfunding* social, seu funcionamento, tecnologias utilizadas e arquitetura do sistema. Ao final, a API REST Web foi documentada.

No capítulo seguinte temos a Conclusão do trabalho dado seus objetivos propostos.

## 4 CONCLUSÃO

Este trabalho apresentou a ferramenta Ajuda.Ai, uma ferramenta para captação de recursos em modalidade *Crowdfunding* para ONGs e Instituições sem fins lucrativos. Para isso, foi feito um estudo sobre o fenômeno do *crowdsourcing* na era digital e do *crowdfunding* que utiliza de *crowdsourcing* para arrecadar recursos financeiros. Além disso, o sistema desenvolvido para atender essa necessidade foi desenvolvido com boas práticas da indústria, utilizando técnicas e ferramentas de ponta como REST Web e CDI.

Considera-se que a solução alcança seu objetivo de ser uma alternativa de baixo custo, de fácil uso e flexível por se utilizar de uma arquitetura moderna que permite interfaces mais elaboradas e consequentemente adequadas aos usuários do serviço. A interface padrão disponibilizada junto ao projeto, uma página web no modelo SPA, tem estética agradável e tenta, através da aplicação de técnicas de publicidade, melhorar a performance das doações às instituições. Sua arquitetura permite que instituições criem, por exemplo, aplicativos para dispositivos móveis que utilizam o Ajuda.Ai como suporte para funcionamento da arrecadação de doações.

Por fim, o trabalho também contribui com a propagação do conhecimento em modelagem, utilização de ferramentas avançadas e boas práticas para implementação do suporte a diferentes *Gateways* de pagamento, além de apresentar uma arquitetura moderna e adequada a grande variedade de meios de acesso a páginas web disponíveis.

O código-fonte da ferramenta está disponível no endereço <https://github.com/g0dkar/ajuda-ai>, está disponível para acesso no endereço <https://ajuda.ai> e a API pode ser acessada através do *endpoint* <https://api.ajuda.ai/v1>.

### 4.1 TRABALHOS FUTUROS

O trabalho de Belleflamme, Lambert e Schwienbacher[4] discute algumas conclusões teóricas sobre porque organizações sem fins lucrativos tendem a ter mais sucesso utilizando *crowdfunding* examinando literatura sobre Teoria de Falha Contratual. Essa teoria se baseia no limite das motivações financeiras dos donos do negócio através de medidas como proibição ou limite dos dividendo de lucros ou doação compulsória dos lucros para projetos sociais. No contexto de empreendimentos sociais, Lehner[15] argumenta que isso pode ser visto como um forte sinal que a organização não visa lucro e pode convidar outras formas de participação dos apoiadores como, por exemplo, trabalho voluntário.

Neste sentido, se planeja a implementação de uma espécie de limite mensal para repasse financeiro onde o excedente é guardado para o mês seguinte. Por exemplo, uma meta de R\$ 1.000,00 mensais é proposta. Considerando que R\$ 1.250,00 tenha sido arrecadado ao final do mês apenas R\$ 1.000,00 seria repassado a organização e os R\$ 250,00 restantes seriam inclusos

no mês seguinte, que começaria com R\$ 750,00 restantes para alcançar a meta.

Em suas próximas versões, é pretendido dar a opção ao usuário para que sejam feitas doações recorrentes. Esse modelo é utilizado por plataformas como Patreon<sup>1</sup> para apoio, em sua maioria, a projetos artísticos e é uma modalidade de doação que vem crescendo bastante e é uma evolução natural do modelo tradicional de *crowdfunding*. Essa forma de doação é vantajosa para as Instituições uma vez que o montante mensal fixo oriundo das doações pode ser considerado como fluxo de caixa e também para o doador, pois a automação do processo de doação pode manter a ajuda permanente sem que o doador precise dedicar tempo a isso com frequência.

Como um último trabalho futuro, pretende-se adaptar e concluir a implementação do Caso de Uso 03 - Acompanhamento das Doações para que se tenha acesso às novas ferramentas mais robustas para acompanhamento dos pagamentos e planejamento financeiro, além das já disponibilizadas pelos *Gateways* de Pagamento.

---

<sup>1</sup> <https://www.patreon.com>

## REFERÊNCIAS

- 1 FLANNERY, M. Kiva and the birth of person-to-person microfinance. *Innovations*, MIT Press, v. 2, n. 1-2, p. 31–56, 2007. Disponível em: <<http://www.mitpressjournals.org/doi/pdf/10.1162/itgg.2007.2.1-2.31>>. Acesso em: 24 jan. 2017. Citado 3 vezes nas páginas 6, 7 e 12.
- 2 SHANKLAND, S. *New Crowdsource app lets you work for Google for free*. 2016. CNet. Disponível em: <<https://www.cnet.com/news/new-crowdsource-app-lets-you-work-for-google-for-free/>>. Acesso em: 24 jan. 2017. Citado na página 11.
- 3 BANNERMAN, S. Crowdfunding culture. *WI Journal of Mobile Media*, v. 7, n. 1, 2013. Disponível em: <<http://wi.mobilities.ca/crowdfunding-culture/>>. Acesso em: 24 jan. 2017. Citado na página 11.
- 4 BELLEFLAMME, P.; LAMBERT, T.; SCHWIENBACHER, A. Crowdfunding: An industrial organization perspective. *Digital Business Models: Understanding Strategies*, v. 1, n. 1, 2010. Disponível em: <[http://economix.fr/pdf/workshops/2010\\_dbm/Belleflamme\\_al.pdf](http://economix.fr/pdf/workshops/2010_dbm/Belleflamme_al.pdf)>. Acesso em: 23 jan. 2017. Citado 3 vezes nas páginas 11, 15 e 52.
- 5 FREEDMAN, D.; NUTTING, M. A brief history of crowdfunding. *Donation, Debt, and Equity Platforms in the USA*, p. 1–10, 2015. Disponível em: <<http://www.freedman-chicago.com/ec4i/History-of-Crowdfunding.pdf>>. Acesso em: 24 jan. 2017. Citado na página 12.
- 6 CORRÊA, M. *Financiamento coletivo cresce no Brasil e pode ajudar a tirar ideias do papel*. 2015. O Globo. Disponível em: <<http://oglobo.globo.com/economia/financiamento-coletivo-cresce-no-brasil-pode-ajudar-tirar-ideias-do-papel-15113776>>. Acesso em: 27 jan. 2017. Citado na página 12.
- 7 CATARSE. *Retrospectiva 2016 do Catarse*. 2017. Catarse. Disponível em: <<http://ano.catarse.me/2016>>. Acesso em: 27 jan. 2017. Citado na página 12.
- 8 GOUVEIA, F. Ongs enfrentam desafios e ocupam espaço da ação pública. *Ciência e Cultura*, scielocec, v. 59, p. 6 – 8, 06 2007. ISSN 0009-6725. Disponível em: <[http://cienciaecultura.bvs.br/scielo.php?script=sci\\_arttext&pid=S0009-67252007000200003&nrm=iso](http://cienciaecultura.bvs.br/scielo.php?script=sci_arttext&pid=S0009-67252007000200003&nrm=iso)>. Acesso em: 25 jan. 2017. Citado na página 13.
- 9 FAGUNDES, H. S. As repercuções do voluntariado e da solidariedade nas políticas sociais no brasil. *Sociedade em Debate*, v. 12, n. 1, p. 87–102, 2012. Disponível em: <<http://revistas.ucpel.tche.br/index.php/rsd/article/view/438>>. Acesso em: 25 jan. 2017. Citado na página 13.
- 10 HOWE, J. *The Rise of Crowdsourcing*. 2006. Wired. Disponível em: <<https://www.wired.com/2006/06/crowds/>>. Acesso em: 23 jan. 2017. Citado na página 15.
- 11 WIKIPEDIA. *Longitude rewards*. 2017. Disponível em: <[https://en.wikipedia.org/wiki/Longitude\\_rewards](https://en.wikipedia.org/wiki/Longitude_rewards)>. Acesso em: 27 jan. 2017. Citado na página 15.

- 12 HEARN, C. *Tracks in the Sea: Matthew Fontaine Maury and the Mapping of the Oceans*. International Marine/McGraw-Hill, 2002. ISBN 9780071368261. Disponível em: <<https://books.google.com.br/books?id=XasPAQAAIAAJ>>. Acesso em: 27 jan. 2017. Citado na página 15.
- 13 GOLÁN, M. L. Crowdfunding: A funding model that succeeds in times of revolt of the masses. p. 1–6, June 2015. ISSN 2166-0727. Citado na página 16.
- 14 BELLEFLAMME, P.; LAMBERT, T.; SCHWIENBACHER, A. Crowdfunding: Tapping the right crowd. *Journal of business venturing*, Elsevier, v. 29, n. 5, p. 585–609, 2014. Disponível em: <<https://ssrn.com/abstract=1578175>>. Acesso em: 01 fev. 2017. Citado na página 16.
- 15 LEHNER, O. M. Crowdfunding social ventures: a model and research agenda. *Venture Capital*, Taylor & Francis, v. 15, n. 4, p. 289–311, oct 2013. Disponível em: <<https://ssrn.com/abstract=2102525>>. Acesso em: 01 fev. 2017. Citado 2 vezes nas páginas 16 e 52.
- 16 WIKIPEDIA. *Bcrypt — Wikipedia, The Free Encyclopedia*. 2017. Disponível em: <<http://en.wikipedia.org/w/index.php?title=Bcrypt&oldid=760862949>>. Acesso em: 09 fev. 2017. Citado na página 19.
- 17 WIKIPEDIA. *ReCAPTCHA — Wikipédia, a encyclopédia livre*. 2017. Disponível em: <<https://pt.wikipedia.org/w/index.php?title=ReCAPTCHA&oldid=46375514>>. Acesso em: 09 fev. 2017. Citado na página 23.
- 18 BOOTH, D. et al. Web services architecture. 2004. Disponível em: <<https://www.w3.org/TR/2003/WD-webarch-20031209/>>. Acesso em: 09 fev. 2017. Citado na página 29.
- 19 GAMMA, E. et al. *Design patterns: elements of reusable object-oriented software*. [S.l.]: Pearson Education India, 1995. Citado na página 30.
- 20 GUPTA, S.; HARTKOPF, J. M.; RAMASWAMY, S. Event notifier: a pattern for event notification. p. 131–153, 2000. Citado na página 33.
- 21 ROS, I. *REST API Documentation Best Practices*. 2012. Disponível em: <<https://bocoup.com/blog/documenting-your-api>>. Acesso em: 09 mar. 2017. Citado na página 35.

## ANEXO A – ESPECIFICAÇÃO DE CASOS DE USO

### ESPECIFICAÇÃO DO CASO DE USO 01 — DOAÇÃO

#### **Objetivo**

Este documento tem por objetivo descrever todos os fluxos envolvidos no Caso de Uso 01 - Doação. São listados e detalhados todos os atores, fluxos, requisitos funcionais e não-funcionais.

#### **Identificação dos Atores**

Esta seção lista e descreve todos os atores envolvidos nos fluxos que compõem o Caso de Uso 01 - Doação.

- **Usuário:** Qualquer pessoa que utiliza o Ajuda.Ai;
- **Cliente:** Interface através da qual o Usuário utiliza o Ajuda.Ai.

#### **Identificação dos Fluxos**

Esta seção lista e descreve todos os fluxos que compõem o caso de uso.

- **Efetuar Doação:** Este fluxo descreve como se dá a efetuação de uma doação a uma Instituição.

#### **Detalhamento dos Fluxos**

- **Efetuar Doação:** Este fluxo especifica a ação de efetuar uma doação a uma Instituição. O usuário fornece dados ao cliente o qual, junto ao sistema, faz uma ordem de pagamento junto ao *Gateway* configurado para a Instituição.

- **Atores:** Usuário, Cliente;
- **Pré-Condições:** Nenhuma;
- **Pós-Condições:** Nenhuma;
- **Requisitos Funcionais:** O sistema deve prover uma interface para execução da doação.

#### **Fluxo Básico**

1. O ator Cliente mostra ao ator Usuário informações sobre uma Instituição;
2. O Usuário decide efetuar uma doação a Instituição;

3. O Cliente solicita os dados para doação: Valor, Se o nome de quem está doando pode ou não ser publicado, Se os custos operacionais do *Gateway* devem ser embutidos no valor da doação, E-mail e Nome;
4. O Usuário informa os dados solicitados;
5. O Cliente envia os dados ao Sistema;
6. O Sistema valida os dados;
7. O Sistema registra a ordem de pagamento no SGBD;
8. O Sistema retorna ao Cliente os dados do pagamento a ser efetuado pelo Usuário;
9. O Cliente direciona o Usuário ao *Gateway* para continuar o processo de pagamento;
10. O caso de uso se encerra.

### **Fluxo Alternativo A**

No Passo 5, caso exista algum erro de comunicação entre Cliente e Sistema:

1. Uma mensagem de erro é exibida ao Usuário informando da falha;
2. O fluxo retorna ao passo 1.

### **Fluxo Alternativo B**

No Passo 6, caso exista algum erro de validação das entradas:

1. Todos os erros de validação encontrados são retornados ao Cliente;
2. O fluxo retorna ao passo 1.

## ESPECIFICAÇÃO DO CASO DE USO 02 — COMUNICAÇÃO COM O DOADOR

### Objetivo

Este documento tem por objetivo descrever todos os fluxos envolvidos no Caso de Uso 02 - Comunicação com o Doador. São listados e detalhados todos os atores, fluxos, requisitos funcionais e não-funcionais.

### Identificação dos Atores

Esta seção lista e descreve todos os atores envolvidos nos fluxos que compõem o Caso de Uso 02 - Comunicação com o Doador.

- **Usuário:** Qualquer pessoa que utiliza o Ajuda.Ai;
- **Instituição:** Usuário representativo de uma Instituição cadastrada no Ajuda.Ai;
- **Cliente:** Interface através da qual o Usuário utiliza o Ajuda.Ai.

### Identificação dos Fluxos

Esta seção lista e descreve todos os fluxos que compõem o caso de uso.

- **Criar Post:** Este fluxo descreve como se dá a criação de um *Post* pela Instituição;
- **Editar Post:** Este fluxo descreve como se dá a alteração de um *Post* pela Instituição.

### Detalhamento dos Fluxos

- **Criar Post:** Este fluxo especifica a ação de criar um *post* vinculado a uma Instituição. A Instituição fornece dados ao cliente o qual, junto ao sistema, cria um *post* vinculado a Instituição que está usando o Sistema no momento.

- **Atores:** Instituição, Cliente;
- **Pré-Condições:** Instituição cadastrada no Sistema, Instituição estar autenticada junto ao Sistema;
- **Pós-Condições:** *Post* cadastrado;
- **Requisitos Funcionais:** O Sistema deve prover uma interface para criação de *posts*.

### Fluxo Básico

1. A Instituição decide criar um novo *post*;
2. A Instituição navega à página de criação de Novo *Post*;
3. O Cliente solicita os dados do Post: Texto da URL, Título, Subtítulo, Conteúdo (formato *Markdown*), imagem do cabeçalho e se o *post* será ou não publicado;

4. A Instituição informa os dados solicitados;
5. O Cliente envia os dados ao Sistema;
6. O Sistema valida os dados;
7. O Sistema registra o *post* no SGBD;
8. O Sistema retorna ao Cliente os dados do *post* criado;
9. O caso de uso se encerra.

### **Fluxo Alternativo A**

Em qualquer passo, caso exista algum erro de comunicação entre Cliente e Sistema:

1. Uma mensagem de erro é exibida ao Usuário informando da falha.

### **Fluxo Alternativo B**

No Passo 6, caso exista algum erro de validação das entradas:

1. Todos os erros de validação encontrados são retornados ao Cliente;
2. O fluxo retorna ao passo 2.

- **Editar Post:** Este fluxo especifica a ação de editar um *post* vinculado a uma Instituição que já existe. Após escolher qual *post* será alterado, a Instituição fornece dados ao cliente o qual, junto ao sistema, altera o *post* vinculado a Instituição com as novas informações.

- **Atores:** Instituição, Cliente;
- **Pré-Condições:** Instituição cadastrada no Sistema, Instituição estar autenticada junto ao Sistema, *Post* da Instituição cadastrado no Sistema;
- **Pós-Condições:** *Post* modificado;
- **Requisitos Funcionais:** O Sistema deve prover uma interface para alteração de *posts*.

### **Fluxo Básico**

1. A Instituição decide alterar um *post*;
2. A Instituição navega a uma página com uma lista de todos os seus *posts*;
3. A Instituição seleciona qual de seus *posts* será alterado;
4. O Cliente solicita os dados do Post: Texto da URL, Título, Subtítulo, Conteúdo (formato *Markdown*), imagem do cabeçalho e se o *post* será ou não publicado;
5. A Instituição informa os dados solicitados;
6. O Cliente envia os dados ao Sistema;

7. O Sistema valida os dados;
8. O Sistema altera o *post* no SGBD;
9. O Sistema retorna ao Cliente os dados do *post* alterado;
10. O caso de uso se encerra.

**Fluxo Alternativo A**

Em qualquer passo, caso exista algum erro de comunicação entre Cliente e Sistema:

1. Uma mensagem de erro é exibida ao Usuário informando da falha.

**Fluxo Alternativo B**

No Passo 7, caso exista algum erro de validação das entradas:

1. Todos os erros de validação encontrados são retornados ao Cliente;
2. O fluxo retorna ao passo 3.

## ESPECIFICAÇÃO DO CASO DE USO 03 — ACOMPANHAMENTO DAS DOAÇÕES

### Objetivo

Este documento tem por objetivo descrever todos os fluxos envolvidos no Caso de Uso 03 - Acompanhamento das Doações. São listados e detalhados todos os atores, fluxos, requisitos funcionais e não-funcionais.

### Identificação dos Atores

Esta seção lista e descreve todos os atores envolvidos nos fluxos que compõem o Caso de Uso 03 - Acompanhamento das Doações.

- **Instituição:** Usuário cadastrado no sistema e que possua pelo menos 01 (uma) Instituição vinculada a ele;
- **Gateway:** *Gateway* de Pagamento selecionado pela Instituição;
- **Cliente:** Interface através da qual o Usuário utiliza o Ajuda.Ai.

### Identificação dos Fluxos

Esta seção lista e descreve todos os fluxos que compõem o caso de uso.

- **Listagem Mensal de Doações:** Este fluxo descreve como se dá a criação de uma listagem mensal de Doações recebidas pela Instituição;
- **Dados de uma Doação:** Este fluxo descreve como se dá a exibição de dados sobre uma Doação em Particular.

### Detalhamento dos Fluxos

- **Listagem Mensal de Doações:** Este fluxo especifica a ação de listar as doações recebidas por uma Instituição em um determinado mês de um determinado ano. A Instituição fornece dados ao cliente o qual, junto ao sistema, cria uma lista, geralmente tabular, das doações recebidas.
  - **Atores:** Instituição, Cliente;
  - **Pré-Condições:** Instituição cadastrada no Sistema, Instituição estar autenticada junto ao Sistema;
  - **Pós-Condições:** Nenhuma;
  - **Requisitos Funcionais:** O Sistema deve prover uma interface para listagem das Doações feitas a uma Instituição.

### Fluxo Básico

1. A Instituição necessita de uma listagem das doações de um mês;
2. A Instituição navega a uma página para listagem de doações;
3. O Cliente solicita os seguintes dados: De qual Instituição deve ser a lista, Mês e Ano da listagem e um dos itens de uma lista de Estados das Doações: Todos, Pago, Pronto para Receber e Cancelado;
4. A Instituição informa os dados solicitados;
5. O Cliente envia os dados ao Sistema;
6. O Sistema valida os dados;
7. O Sistema retorna ao Cliente os dados da listagem requisitada;
8. O caso de uso se encerra.

### **Fluxo Alternativo A**

Em qualquer passo, caso exista algum erro de comunicação entre Cliente e Sistema:

1. Uma mensagem de erro é exibida ao Usuário informando da falha.

### **Fluxo Alternativo B**

No Passo 6, caso exista algum erro de validação das entradas:

1. Todos os erros de validação encontrados são retornados ao Cliente;
2. O fluxo retorna ao passo 2.

- **Dados de uma Doação:** Este fluxo especifica a ação de ver dados de uma única doação específica. Após escolher qual doação será exibida, a Instituição fornece os dados necessários ao cliente o qual, junto ao sistema, pega e exibe as informações requeridas.

- **Atores:** Instituição, Cliente;
- **Pré-Condições:** Instituição cadastrada no Sistema, Instituição estar autenticada junto ao Sistema, Haverem Doações feitas a Instituição;
- **Pós-Condições:** Nenhuma;
- **Requisitos Funcionais:** O Sistema deve prover uma interface para expor dados de um pagamento.

### **Fluxo Básico**

1. A Instituição decide ver dados de uma doação;
2. A Instituição seleciona qual doação será exibida;
3. A Instituição navega a uma página para exibição das informações da doação;
4. O Cliente requisita ao Sistema os dados da doação;

5. O Sistema busca os dados da doação especificada;
6. O Sistema retorna ao Cliente os dados da doação;
7. O caso de uso se encerra.

### **Fluxo Alternativo A**

Em qualquer passo, caso exista algum erro de comunicação entre Cliente e Sistema:

1. Uma mensagem de erro é exibida ao Usuário informando da falha.

### **Fluxo Alternativo B**

No Passo 5, caso a doação esteja marcada como Anônima (anonymous = true):

1. Todas as informações de identificação do doador são retiradas do objeto (payeeName e payeeEmail);
2. O fluxo continua ao passo 6.