

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Emerson Ribeiro Junior

**TaskMe: um sistema *web* para prestação de
serviços baseado em tarefas**

Uberlândia, Brasil

2021

UNIVERSIDADE FEDERAL DE UBERLÂNDIA

Emerson Ribeiro Junior

**TaskMe: um sistema *web* para prestação de serviços
baseado em tarefas**

Trabalho de conclusão de curso apresentado
à Faculdade de Computação da Universidade
Federal de Uberlândia, Minas Gerais, como
requisito exigido parcial à obtenção do grau
de Bacharel em Sistemas de Informação.

Orientador: Jean Roberto Ponciano

Universidade Federal de Uberlândia – UFU

Faculdade de Ciência da Computação

Bacharelado em Sistemas de Informação

Uberlândia, Brasil

2021

Emerson Ribeiro Junior

TaskMe: um sistema *web* para prestação de serviços baseado em tarefas

Trabalho de conclusão de curso apresentado à Faculdade de Computação da Universidade Federal de Uberlândia, Minas Gerais, como requisito exigido parcial à obtenção do grau de Bacharel em Sistemas de Informação.

Trabalho aprovado. Uberlândia, Brasil, 23 de março de 2021:

Jean Roberto Ponciano
Orientador

Bruno Augusto Nassif Travençolo

Claudio Douglas Gouveia Linhares

Uberlândia, Brasil
2021

Dedico esse trabalho a minha esposa Laura, meu filho Jack, a minha mãe Eliete, meus irmãos Alisson, Bruno e meu grande amigo Eliseu, que me deram o suporte necessário durante toda a minha caminhada rumo a conclusão dessa graduação.

Agradecimentos

Agradeço, em especial, ao professor Jean Roberto Ponciano, pela paciência e incentivo que tornaram possível a conclusão desta monografia.

Agradeço a todo o corpo docente da Faculdade de Computação da Universidade Federal de Uberlândia.

Agradeço a minha esposa e meu filho Jack que me motivaram até o fim na conclusão deste trabalho, e que me deram o suporte que precisei durante toda a graduação.

“You can’t connect the dots looking forward; you can only connect them looking backwards. So you have to trust that the dots will somehow connect in your future. You have to trust in something – your gut, destiny, life, karma, whatever. This approach has never let me down, and it has made all the difference in my life.” — Steve Jobs

Resumo

As plataformas de prestação de serviço por mais completas que sejam, ainda sim carecem de funções que poderiam facilitar a disposição das informações e a utilização do sistema pelos usuários. Algumas dessas funções são: ausência de uma organização em que o usuário pode consultar um tipo de agenda para saber quais dias da semana ele ainda não tem atividades para executar; *feedback* mais rápido através de um *chat* para comunicação interna entre o criador da tarefa e os interessados na tarefa; e a distinção entre tarefas pontuais e periódicas. Este trabalho tem como objetivo propor uma aplicação web direcionada a pessoas que procuram oportunidades de renda extra e pessoas que optem por delegar tarefas, sejam elas simples como ir a um mercado ou mais complexas como prestar um serviço de diarista, jardineiro, encanador, etc. Durante o desenvolvimento desse trabalho foi construída a aplicação proposta, utilizando a metodologia ágil Scrum, e tecnologias web como HTML, CSS, JavaScript e o *framework* Angular, com um servidor em Java com o auxílio do *framework* Spring, base de dados em PostgreSQL, e uma arquitetura *Model View Controller*. Toda a aplicação foi disponibilizada na nuvem, tanto a interface quanto o servidor, por meio da plataforma Heroku. O sistema resultante é reflexo do emprego das motivações obtidas por meio de aplicações correlatas e das tecnologias apresentadas.

Palavras-chave: Desenvolvimento de Software, Aplicação Web, Prestação de Serviços.

Lista de ilustrações

Figura 1 – Tela de Pedidos de prestação de serviços na aplicação móvel da plataforma GetNinjas	16
Figura 2 – Tela de Meus pedidos na plataforma GetNinjas	17
Figura 3 – Tela da aplicação móvel da plataforma Habitíssimo que exibe os interessados em prestar determinado tipo de serviço	18
Figura 4 – Tela principal da aplicação web da plataforma Iguanafix	19
Figura 5 – Exemplo de Gráfico de <i>Burndown</i>	22
Figura 6 – Estória para criação de novos campos na entidade usuário, que serão utilizadas no cálculo da classificação como prestador	29
Figura 7 – Estória para criação da tela principal do sistema	29
Figura 8 – Diagrama de entidade relacionamento estilo de notação Chen	30
Figura 9 – Diagrama de entidade relacionamento estilo IDEF1X (<i>Integration Definition for Information Modeling</i>)	31
Figura 10 – Diagrama de sequência - Registro de usuário	32
Figura 11 – Diagrama de sequência - Criação de uma nova tarefa com upload de imagem	32
Figura 12 – Tela inicial da aplicação <i>TaskMe</i>	33
Figura 13 – Formulário de cadastro de usuário	34
Figura 14 – Página principal da aplicação	34
Figura 15 – Opções do <i>dropdown</i>	35
Figura 16 – Formulário de cadastro de usuário	35
Figura 17 – <i>Upload</i> de nova foto de perfil	36
Figura 18 – Acesso a tela <i>Task Someone</i> através da barra de navegação da aplicação	36
Figura 19 – Formulário para criação de tarefa	37
Figura 20 – <i>Upload</i> de imagens relacionadas à tarefa	37
Figura 21 – Tela de detalhes da tarefa	38
Figura 22 – Tela de tasks criadas pelo usuário	38
Figura 23 – Diálogo para alteração de informações da tarefa	39
Figura 24 – Nova tarefa sendo exibida na tela <i>Find Tasks</i>	40
Figura 25 – Botão de interessado na tela de detalhes da tarefa	40
Figura 26 – Nova tarefa sendo exibida na tela <i>Find Tasks</i>	41
Figura 27 – Dica ao usuário - acesse suas aplicações	41
Figura 28 – Tela de aplicações do usuário	41
Figura 29 – Tela de mensagens	42
Figura 30 – Botões para visualização de candidatos (1) e alteração da situação da tarefa (2) na tela de gerenciamento de tarefas criadas	42

Figura 31 – Diálogo de exibição de candidatos da tarefa	43
Figura 32 – Diálogo de mensagem de aprovação	44
Figura 33 – Diálogo de aplicantes após refinamento de candidatos	44
Figura 34 – Diálogo de participantes da tarefa	45
Figura 35 – Opções de mudança de estado da tarefa	45
Figura 36 – Tela de tarefas pontuais	46
Figura 37 – Confirmação de início de tarefa	46
Figura 38 – Opções de mudança de estado quando a tarefa está em estado de execução	47
Figura 39 – Diálogo de classificação de participantes da tarefa	47
Figura 40 – Informações da tarefa periódica sendo criada	48
Figura 41 – Informações na tela de detalhes da tarefa	49
Figura 42 – Candidatos à tarefa	49
Figura 43 – Conclusão da escolha de candidatos	50
Figura 44 – Confirmação de início de tarefa periódica	51
Figura 45 – Botão para exibição das sub-tarefas	51
Figura 46 – Diálogo de sub-tarefas	52
Figura 47 – Diálogo de edição de sub-tarefas	53
Figura 48 – Sub-tarefas pós edição de título e estado	54
Figura 49 – Opção para acesso à tela de agenda	54
Figura 50 – Tela de agenda	55
Figura 51 – Navegação e filtro de datas na tela de agenda	55
Figura 52 – Detalhes da sub-tarefa na tela de agenda	56
Figura 53 – Comentários feitos na sub-tarefa	56
Figura 54 – Tarefa finalizada, com comentário da criadora	57
Figura 55 – Diálogo de sub-tarefas contendo somente uma finalizada	57
Figura 56 – Erro após tentativa de finalização de tarefa periódica sem ter finalizado suas sub-tarefas	58
Figura 57 – Todas as sub-tarefas finalizadas	58
Figura 58 – Mudança de estado da tarefa para finalizada (<i>Done</i>)	58
Figura 59 – Sub-tarefas concluídas exibidas na tela de Agenda (<i>Task Agenda</i>) . . .	59
Figura 60 – Classificação dos participantes da tarefa	59

Lista de abreviaturas e siglas

SMS	<i>Short Message Service</i>
HTML	<i>Hyper Text Markup Language</i>
CSS	<i>Cascading Style Sheets</i>
JSON	<i>JavaScript Object Notation</i>
XML	<i>Extensible Markup Language</i>
MVC	<i>Model View Controller</i>
DOM	<i>Document Object Model</i>
POJOS	<i>Plain Old Java Objects</i>
SGBDOR	Sistema de Gerenciamento de Banco de Dados Objeto-Relacional
MER	Modelo Entidade Relacionamento
DER	Diagrama Entidade Relacionamento
IDEF1X	<i>Integration Definition for Information Modeling</i>

Sumário

1	INTRODUÇÃO	12
1.1	Objetivos	14
1.1.1	Objetivo Geral	14
1.1.2	Objetivos Específicos	14
1.2	Organização do Trabalho	14
2	REFERENCIAL TEÓRICO	15
2.1	Aplicações de mercado correlatas	15
2.1.1	GetNinjas	15
2.1.2	Habitíssimo	16
2.1.3	IguanaFix	17
2.2	Tecnologias	18
2.2.1	Scrum	19
2.2.1.1	Funcionamento do Scrum	20
2.2.2	Model Viewer Controller (MVC)	22
2.2.3	HTML5	23
2.2.4	CSS3	23
2.2.5	JavaScript	24
2.2.6	Bootstrap 4	25
2.2.7	Angular	25
2.2.8	Java	26
2.2.9	Spring 2	26
2.2.10	PostgreSQL	26
3	TASKME	28
3.1	Desenvolvimento da aplicação	28
3.2	Modelagem do sistema	29
3.2.1	Diagramas de entidade relacionamento (DER)	30
3.2.2	Diagramas de sequência	32
3.3	Descrição do sistema	33
3.3.1	Registro de Usuário	33
3.3.2	Alterando as informações de perfil e adicionando foto	33
3.3.3	Fluxo de criação e completude de tarefas pontuais	35
3.3.4	Fluxo de criação e completude de tarefas periódicas	47
4	CONCLUSÃO	60

4.1	Trabalhos Futuros	61
	REFERÊNCIAS	62

1 Introdução

De acordo com (IBGE, 2020), a taxa de desemprego aumentou no segundo trimestre de 2020, tendo como base o trimestre anterior, e isso tende a gerar um aumento na busca por oportunidades de trabalho e/ou renda extra. Atualmente essa procura é facilitada devido a imensa quantidade de meios de comunicação *online*, que tornam mais expostas as pessoas necessitadas por um serviço, e disponibilizam essas informações de forma acessível aos indivíduos com a capacidade de executá-los.

Essa disposição de informações na internet requer atenção, já que para pessoas em busca de oportunidades, nem sempre é trivial distinguir atividades reais de fraudulentas. Segundo (GLOBO, 2017), é necessário cautela quando nos referimos a ofertas de empregos presentes no Facebook ou em outra rede social, pois existem ofertas de emprego fraudulentas onde os supostos recrutadores pedem um depósito de um determinado valor para que a pessoa comece a trabalhar, quando na verdade não há emprego algum.

Muitas redes sociais, como Instagram e Facebook, contêm uma quantidade imensa de material e que tratam dos mais diversos assuntos, o que é justificável visto que são redes sociais sem nicho. Mas no âmbito profissional, os usuários convergem para o LinkedIn, uma rede social de negócios fundada em dezembro de 2002 que desde então mantém seu foco principal em negócios. De acordo com (ESTÁGIOS, 2019), o LinkedIn é o canal mais utilizado para quem busca uma vaga de estágio ou *trainee*, o que demonstra maior confiança por parte dos usuários ao buscar oportunidades quando toda a interação acontece pela plataforma.

O LinkedIn é o mais utilizado na busca por posições geralmente duradouras e que necessitam de no mínimo uma graduação em curso. De acordo com (SOCIAL, 2020), 51% dos usuários da plataforma possuem pós-graduação, enquanto 26% são graduados seguidos de 9% de usuários com ensino médio ou inferior. Assim observa-se que não são publicadas oportunidades para pessoas de qualquer perfil e escolaridade no LinkedIn, demonstrando que pessoas sem estudo tendem a ter dificuldade de encontrar algo na plataforma, pois renda-extra e trabalhos informais não são o foco da mesma.

Existem plataformas que não têm esse aspecto de rede social, mas seu objetivo é conectar clientes aos prestadores de serviço, como a Habitissimo¹ e a getNinjas². Nessas plataformas, o usuário pode solicitar serviços desde manutenção de ar condicionado até passeador de cães, e os executantes desses serviços podem ser pessoas ou empresas, todas contendo uma classificação (um tipo de ranking) baseada em serviços prestados anterior-

¹ <https://www.habitissimo.com.br/>

² <https://www.getninjas.com.br/>

mente. Existe também o IguanaFix³ que possui a mesma proposta porém voltado para domicílio, como montagem de móveis, e também manutenção para empresas e escritórios. Nessas plataformas alguns serviços necessitam de formação técnica, como dar manutenção em ar condicionado, por exemplo, mas em diversos outros casos não. Mesmo sendo plataformas distintas, o que elas tem em comum é que sempre ao finalizar uma busca por um serviço você deve aguardar pelos orçamentos, mas em diversos momentos não temos tempo para aguardar, e precisamos do serviço feito o mais rápido possível.

Consideremos um exemplo de atividade na qual um jardineiro deverá manter a grama do contratante cortada, prestando serviços mensalmente. Nas plataformas mencionadas, não existe uma organização onde o jardineiro possa consultar um tipo de agenda para saber quando tem disponibilidade para trabalhar. Então, tanto para serviços urgentes pontuais (de ocorrência única), quanto periódicos, essas plataformas ainda não se adequaram, requirindo assim que o contratado se organize num meio externo, onde a aplicação não tem controle sobre falhas humanas, como por exemplo, esquecer de anotar que tem um trabalho na terça-feira às 15:00 e acabar se comprometendo com outro cliente no mesmo horário.

Este trabalho tem por foco o desenvolvimento da aplicação TaskMe, que irá potencializar a organização do trabalho para que os contratados não se percam em meio a diversos serviços. São diferenciais da aplicação:

- Serviços periódicos, por meio dos quais será possível criar diversas tarefas a partir de uma única, de acordo com a periodicidade (semanalmente, mensalmente, trimestralmente).
- Agenda, onde o usuário é capaz de se organizar verificando seus horários livres, onde tanto os serviços periódicos quanto os pontuais (de ocorrência única) seriam exibidos.
- Chat integrado na aplicação para que o interessado em fazer o serviço, logo após se aplicar, possa dialogar com o criador da tarefa, assim ele já recebe o *feedback* e inicia a resolução do problema.
- Comentários privados vinculados à tarefa, onde apenas os participantes da tarefa possam ler e até incluir novos comentários privados, com o objetivo de atualizar o criador em relação ao andamento da tarefa, ou caso haja a necessidade de compartilhar informações entre todos os participantes de iniciar algum tipo de discussão.

³ <https://www.iguanafix.com.br/>

1.1 Objetivos

1.1.1 Objetivo Geral

Desenvolver uma aplicação para oferta de serviços (tanto periódicos quanto pontuais) que terá como principais funcionalidades uma organização de informações elaborada, um meio de comunicação embutido e mecanismos para aumento da confiabilidade.

1.1.2 Objetivos Específicos

- Realizar o levantamento de requisitos e identificar funcionalidades faltantes e pontos a serem melhorados em outras plataformas de mercado (GetNinjas, IguanaFix, Habitissimo).
- Desenvolver a solução com base nos requisitos levantados, e utilizando tecnologias amplamente difundidas no mercado.
- Hospedar a aplicação e banco de dados em uma plataforma em nuvem, a fim de abstrair problemas de armazenamento de dados e hardware obsoleto ou danificado.

1.2 Organização do Trabalho

Este documento está organizado da seguinte forma: o Capítulo 2 apresenta os projetos relacionados e o referencial teórico, composto pela exibição das aplicações correlatas GetNinjas, Habitissimo e IguanaFix, e as ferramentas utilizadas neste trabalho, sendo elas, HTML5, CSS3, JavaScript, framework Angular, banco PostgreSQL, arquitetura MVC e Scrum como metodologia ágil; o Capítulo 3 mostra o processo de desenvolvimento da aplicação, sua modelagem e a descrição do mesmo através de fluxos detalhados; por fim, no Capítulo 4, são apresentadas as considerações finais relacionadas ao trabalho, que inclui o resultado esperado e as possibilidades de trabalhos futuros que possam aprimorar a aplicação proposta.

2 Referencial Teórico

Este capítulo irá apresentar as aplicações correlatas GetNinjas, Habitíssimo e IguanaFix, bem como suas interfaces e fluxos, para que a aplicação a ser desenvolvida possa conter funcionalidades não existente nelas ou até mesmo melhorar algum fluxo já existente.

Serão apresentadas também as ferramentas utilizadas neste trabalho, que foram HTML5, CSS3, JavaScript e o *framework* Angular na parte da interface do usuário. Já no servidor, utilizamos a linguagem Java com o auxílio do *framework* Spring, e como sistema gerenciador de banco de dados, utilizamos o PostgreSQL. Para o desenvolvimento da aplicação foi utilizado a metodologia *Scrum* e o padrão de arquitetura *Model Viewer Controller* (MVC) foi utilizado para simplificar a criação de novas funcionalidade e descomplicar futuras manutenções e melhoria dos códigos.

2.1 Aplicações de mercado correlatas

A fim de motivar, foi necessária uma análise de mercado, na qual verificamos como é a apresentação de dados dessas plataformas, e em quais pontos é possível haver uma melhoria. As plataformas analisadas foram GetNinjas, Habitíssimo e IguanaFix, tais quais serão descritas com clareza a seguir.

2.1.1 GetNinjas

Trata-se de um aplicativo para contratação de serviços que em 2012 ganhou o prêmio de melhor *startup* brasileira no prêmio *The Next Web* (DEMETRIO, 2012). A plataforma, disponível para Android, iOS e web, conta com uma variedade de serviços de mais de 200 tipos, que vão desde reformas e reparos, serviços domésticos até fotógrafos e designers.

O funcionamento da plataforma consiste em dois cadastros distintos, para prestadores de serviço e para clientes, isto é, pessoas em busca destes. Caso o usuário opte por ser um prestador, ele deverá fazer um cadastro inicial com e-mail e número de celular, para então receber um SMS (*Short Message Service*) e, por fim, baixar o aplicativo no dispositivo móvel e utilizar o código recebido para continuar a criação da conta de prestador de serviço. Após finalizar o cadastro, todos os clientes que pedem por serviço na região aparecem no aplicativo, como mostra a Figura 1. O GetNinjas não tem mensalidade, e nem mesmo taxa de cadastro. O usuário só se candidata para serviços que lhe interessam

ou de acordo com sua competência, e ao final o valor total cobrado pelo serviço não sofre deduções pela plataforma.

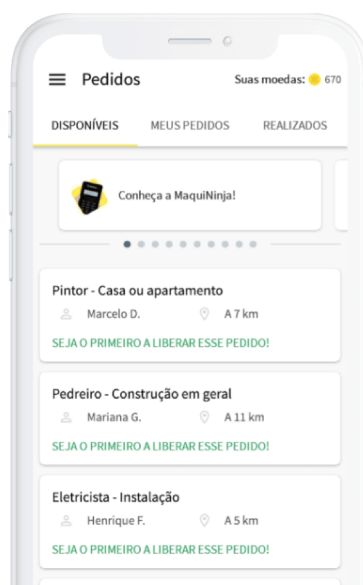


Figura 1 – Tela de Pedidos de prestação de serviços na aplicação móvel da plataforma GetNinjas (fonte: [GetNinjas](https://www.getninjas.com.br/))

Caso o usuário seja apenas uma pessoa que necessita de algum serviço, ele pode apenas utilizar a página inicial do *website* do GetNinjas¹ para solicitar um orçamento. Não há necessidade de criar uma conta, pois ao prover seu e-mail, todos os orçamentos serão enviados para o mesmo.

De acordo com o site da plataforma, em 60 minutos haverá o recebimento por e-mail do primeiro orçamento para o serviço solicitado, sendo que o limite é de quatro orçamentos por pedido, então caso seja algo mais complexo, apenas quatro orçamentos podem não ser o bastante pra decidir o melhor prestador pra determinado serviço. E através dos *links* presentes no e-mail, é possível ter acesso aos pedidos de orçamento que já foram solicitados, como na Figura 2, que mostra inclusive os prestadores que fizeram o envio de orçamentos para aquele determinado pedido.

2.1.2 Habitíssimo

Logo ao acessar a página inicial do *website*², vemos a tela que apresenta todas as categorias de serviço que estão ao nosso dispor na plataforma. O processo de solicitação de orçamento é bem similar ao GetNinjas, onde é necessário preencher qual a localização onde o serviço será executado, selecionar o serviço e posteriormente, de acordo com o serviço que for selecionado, haverá algumas perguntas relacionadas ao serviço, por exemplo, caso

¹ <https://www.getninjas.com.br/>

² <https://www.habitissimo.com.br/>

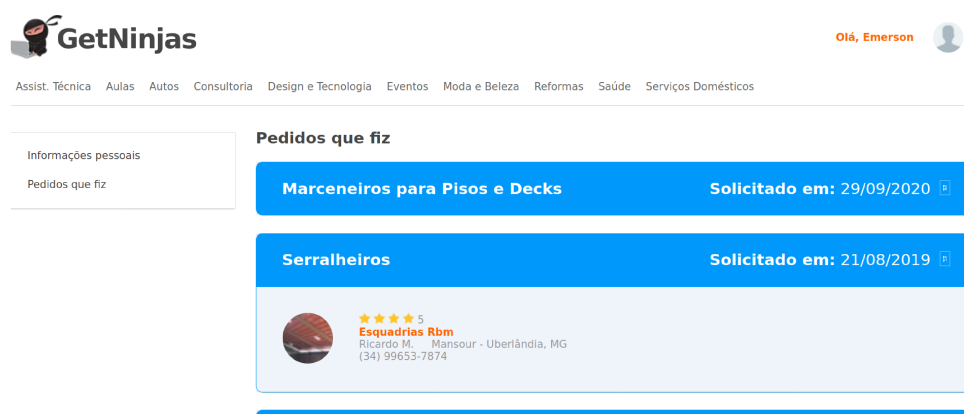


Figura 2 – Tela de Meus pedidos na plataforma GetNinjas (fonte: [GetNinjas](#))

seja selecionado o serviço de Pintura, as perguntas seriam algo como: a pintura é interna ou externa? Qual o tipo do imóvel? Qual a área a ser pintada? Quais superfícies serão pintadas (teto e paredes)? E logo após essas perguntas basta preencher os dados pessoais e fazer a confirmação de seu número de celular, pois a aplicação envia notificações tanto para o e-mail quanto para o celular do usuário.

Assim que os interessados encontrarem um pedido de orçamento, elas podem manifestar o interesse, e o usuário deve então analisar os interessados de acordo com sua classificação e fichas, e então entrar em contato com o prestador escolhido. As informações mostradas na Figura 3 são acessíveis através de um *link*, sem a necessidade de fazer *login* na plataforma (a criação de uma conta é opcional).

2.1.3 IguaaFix

Trata-se de uma plataforma mais específica que as anteriores, porém com a mesma ideia de oferta de serviços. Nesse caso, como é exibido em seu *website*³, Ela oferece dois tipos de serviços para casa: montagem de móveis e fixação. Já para serviços destinados a empresas, a plataforma tem diversos serviços de manutenção, como ar condicionado, elétrica, hidráulica, pintura, pedreiro e luminárias.

A diversidade de serviços é maior para empresas pois a empresa foca bastante na plataforma para gerenciar os serviços contratados, o que é importante para uma empresa saber o que está pagando, quando os prestadores de serviço estão trabalhando e qual está sendo o resultado. Para o perfil empresarial, essa plataforma conta com as seguintes características:

- Contrate todos os serviços em um só lugar.
- Acompanhe e agende todos os trabalhos em cada filial.

³ <https://www.iguanafix.com.br/>

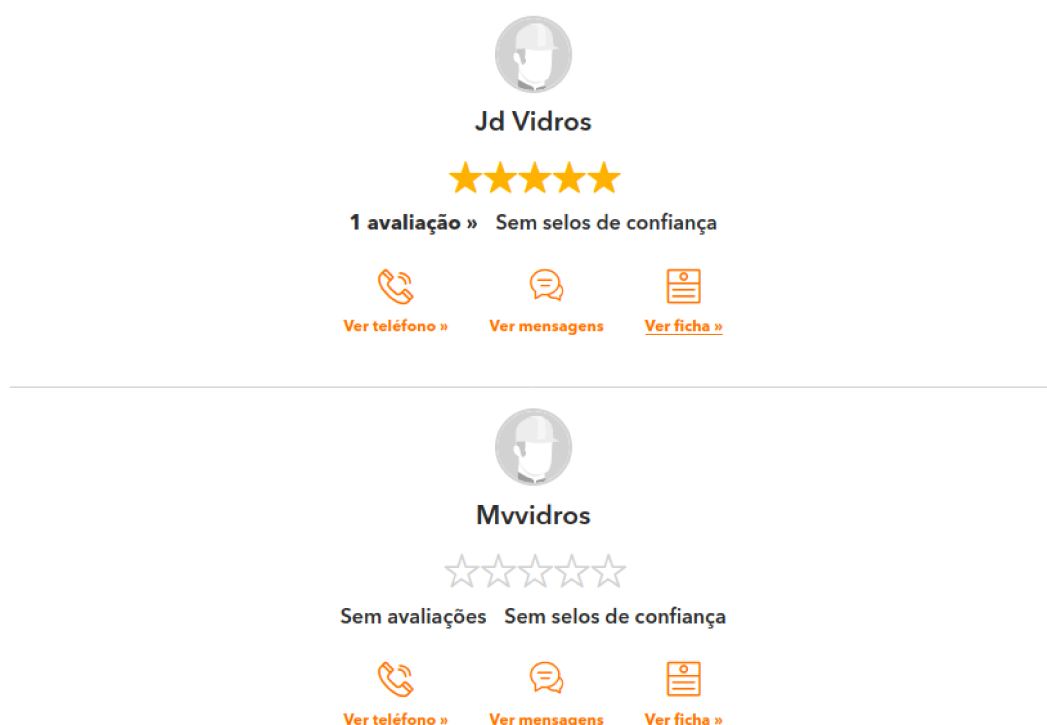


Figura 3 – Tela da aplicação móvel da plataforma Habitíssimo que exibe os interessados em prestar determinado tipo de serviço (fonte: [Habitissimo](#))

- Relatórios com fotos e informação de toda manutenção realizada.
- Detecção, gestão e reparo de incidências por filial.
- Visibilidade e controle de gastos.

A plataforma é bastante organizada para que empresas possam estar a parte dos serviços contratados e contratar mais serviços através da mesma. Por outro lado, a variedade de serviços ofertados é muito pequena.

Ao se cadastrar na plataforma, o perfil do usuário é imposto como pessoal (ao invés de empresarial), ou seja, poderá contratar apenas os dois tipos de serviços citados acima (montagem de móveis e fixação) e a plataforma pessoal não contém todas as funcionalidades da empresarial, como mostra a Figura 4. Para se ter acesso à plataforma de gerenciamento, é necessário efetuar um cadastro empresarial e aguardar a solicitação ser aprovada.

2.2 Tecnologias

Neste trabalho o *framework* de desenvolvimento ágil implementado foi o Scrum, e o padrão de projeto adotado, a nível de código, foi o *Model Viewer Controller* (MVC), para



Figura 4 – Tela principal da aplicação web da plataforma Iguanafix (fonte: [Iguanafix](#))

facilitar a reusabilidade de código, testes e alterações futuras. As tecnologias utilizadas para o desenvolvimento da aplicação foram: HTML5; CSS3; JavaScript e Java. O sistema gerenciador de banco de dados objeto relacional utilizado foi o PostgreSQL. Paralelamente às tecnologias citadas, foram utilizados os *frameworks*: Bootstrap 4 e Spring 2. Todos esses frameworks e tecnologias são apresentados em detalhes a seguir.

2.2.1 Scrum

O método Scrum, de acordo com (DUARTE, 2020), foi criado por Ken Schwaber e Jeff Sutherland, e se trata do framework ágil mais utilizado no mundo atualmente, sendo considerado por muitos como o passo inicial para a implantação de uma cultura ágil nas empresas que desenvolvem software.

No artigo “*The New Product Development Game*” (TAKEUCHI, 1986), os autores Hirotaka Takeuchi e Nonaka Ikujiro definiram uma estratégia flexível e completa para a fabricação de automóveis e produtos de consumo. Eles propuseram que o desenvolvimento de um produto não deve ser algo sequencial, mas sim semelhante a um jogo de rugby onde o time trabalha em conjunto, passando a bola para frente e para trás e se movendo através do campo, com simetria. Foi percebido que, quando o desenvolvimento de um produto utilizava essa solução, as equipes pequenas e multidisciplinares produziram resultados melhores. Eles então associaram estas equipes altamente eficazes à formação “Scrum” do rugby, que como visto em (BERNARDO, 2019) consiste em um grupo de jogadores que se reúnem para reiniciar uma determinada jogada. Nela o trabalho em grupo é muito valorizado e importante, pois se um dos integrantes não cumprir com sua parte, toda a

jogada pode ser perdida.

A utilização deste método pode gerar benefícios, como o aumento da satisfação de clientes por meio da diminuição das reclamações (Mann; Maurer, 2005), melhoria na comunicação e aumento da colaboração entre envolvidos nos projetos (Berczuk, 2007), crescimento do retorno do investimento em projetos de novos produtos (Sulaiman; Barton; Blackburn, 2006), acréscimo da motivação da equipe de desenvolvimento de produtos (Kniberg; Farhang, 2008), avanço da qualidade do produto produzido (Sutherland et al., 2008), redução dos custos de produção (Sutherland et al., 2007), alta na produtividade da equipe de desenvolvimento (Marcal et al., 2007), redução do tempo gasto para terminar projetos de desenvolvimento de novos produtos (SANDERS, 2007) e diminuição do risco em projetos de desenvolvimento de novos produtos (EDWARDS, 2008).

2.2.1.1 Funcionamento do Scrum

De acordo com (CARVALHO; MELLO, 2012), este método não requer ou fornece qualquer técnica específica para a fase de desenvolvimento, apenas estabelece conjuntos de regras e práticas gerenciais que devem ser adotadas para o sucesso de um projeto.

O início do Scrum se dá no *Backlog* do Produto (*Product Backlog*), que consiste em organizar e gerenciar os requisitos coletados (SCHWABER; BEEDLE, 2002). De acordo com (CARVALHO; MELLO, 2012), nesta prática, por meio de reuniões com todos os envolvidos, investidores, clientes e parceiros no projeto, são apontadas todas as necessidades do negócio e as funcionalidades a serem desenvolvidas. Assim, o *Backlog* do Produto é uma lista de funcionalidades, ordenadas por prioridade, que provavelmente serão desenvolvidas durante o projeto.

Diariamente os membros do time realizam uma reunião rápida, denominada reunião diária de Scrum (*Daily Scrum*), que se trata de um encontro breve para determinar quais serão as tarefas do dia corrente e verificar o status de tarefas do dia anterior no que se diz respeito a bloqueios, problemas e porcentagem de conclusão. Como colocado por (CARVALHO; MELLO, 2012), esta reunião é também chamada de *Stand Up Meeting* (reunião em pé), já que é de praxe que todos os membros a realizem de pé, de forma a conseguir maior agilidade. Três perguntas são respondidas por cada membro sobre suas responsabilidades (RISING; JANOFF, 2000): O que foi feito ontem? O que será feito hoje? Há algum obstáculo à realização das atividades?

As respostas para as diversas perguntas feitas durante as reuniões não são direcionadas à gerência, mas sim ao time, com o objetivo de tornar formal o envolvimento com o resto da equipe. Desta forma toda a equipe tem conhecimento das metas e bloqueios (que podem vir a gerar riscos) de cada membro, gerando uma cobrança multilateral.

Os blocos contidos na linha de tempo de uma equipe Scrum são chamados de

Sprints, períodos de tempo onde os itens definidos no *Backlog* do Produto são implementados. Conforme (ABRAHAMSSON et al., 2017), ele normalmente dura de uma a quatro semanas, mas não há uma regra para isto, as equipes que decidem a duração a ser adotada para o projeto. Para nosso contexto, onde será desenvolvido um *software*, o *Sprint* inclui as fases tradicionais do desenvolvimento de software: requisitos, análise, projeto e entrega.

Todas as atividades a serem feitas durante a *Sprint* vem de um subconjunto do *Backlog* do Produto, chamado *Backlog* do *Sprint*. De acordo com (CARVALHO; MELLO, 2012), ele é uma lista de atividades a serem desenvolvidas durante o *Sprint*. Sua definição acontece durante a Reunião de Planejamento do *Sprint*. Já a Reunião de Revisão do *Sprint* (*Sprint Review Meeting*) é a reunião que acontece após cada *Sprint*. Nela, a equipe discute sobre seus erros, acertos e lições aprendidas.

No começo do projeto, de acordo com (ABRAHAMSSON et al., 2017), os clientes descrevem quais funcionalidades desejam ver na primeira versão do sistema, e ao mesmo tempo os desenvolvedores se familiarizam com as práticas e tecnologias que serão utilizadas no projeto. Após isso, é feita uma ordenação de prioridades acordada entre cliente e desenvolvedor, e a lista resultante pode ser considerada o *Product Backlog*, pois já foram estimados os custos e datas de entrega de resultados. Conforme (CARVALHO; MELLO, 2012), para assegurar que o processo Scrum está sendo seguido e tentar resolver impedimentos que possam comprometer o trabalho dos membros da equipe, um dos desenvolvedores deve ser eleito “*Scrum Master*”, cujo papel se assemelha a um gerente de projetos. Suas principais funções são cuidar de obstáculos apontados no *Daily Scrum* e lidar com potenciais riscos, adicionando-os em uma lista chamada *Backlog* de Impedimentos. Assim os demais desenvolvedores podem se concentrar principalmente nas questões técnicas.

Outro papel essencial é o do Dono do Produto (*Product Owner*). Este membro reflete o cliente (interno ou externo) e define quais são os requisitos e qual é o grau de importância e prioridade de cada um deles. Geralmente o *Product Owner* é escolhido pelo *Scrum Master*, cliente e gerência. Para desempenhar sua função, ele precisa conhecer muito bem as regras de negócios do cliente, de forma a solucionar dúvidas que o time possa ter em relação as funcionalidades do produto (CARVALHO; MELLO, 2012).

Ao iniciar cada *Sprint* e após decidir o *Sprint Backlog*, as responsabilidades são distribuídas dentre os integrantes da equipe. Os desenvolvedores debatem os padrões a serem adotados e as atividades de análise, codificação e testes começam. Ao final de cada *Sprint*, deve ser entregue ao cliente uma versão funcional do produto, para obtenção de *feedback*, que por sua vez pode vir a criar novas tarefas no *Backlog* sobre funcionalidades ou defeitos a serem endereçadas para próximas *Sprints*.

Um componente recorrente do Scrum é o gráfico *burndown* (*Burndown Chart*), que se trata de uma representação gráfica do trabalho restante em comparação com o

trabalho já realizado. Geralmente, coloca-se a quantidade de trabalho no eixo vertical (Y) e o tempo no eixo horizontal (X). Conforme (CARVALHO; MELLO, 2012), ele é muito útil para prever quando todo o trabalho será completado e para alarmar o time em caso de atraso (que ficará bastante aparente). Geralmente é traçada uma linha com a representação da execução do trabalho. Conforme ilustrado pela Figura 5, na medida em que o tempo passa (maior valor para o eixo X), menos trabalho pendente há para ser feito (menor valor para o eixo Y), de forma que, ao fim da *Sprint*, espera-se que todo o trabalho tenha sido executado conforme planejado.

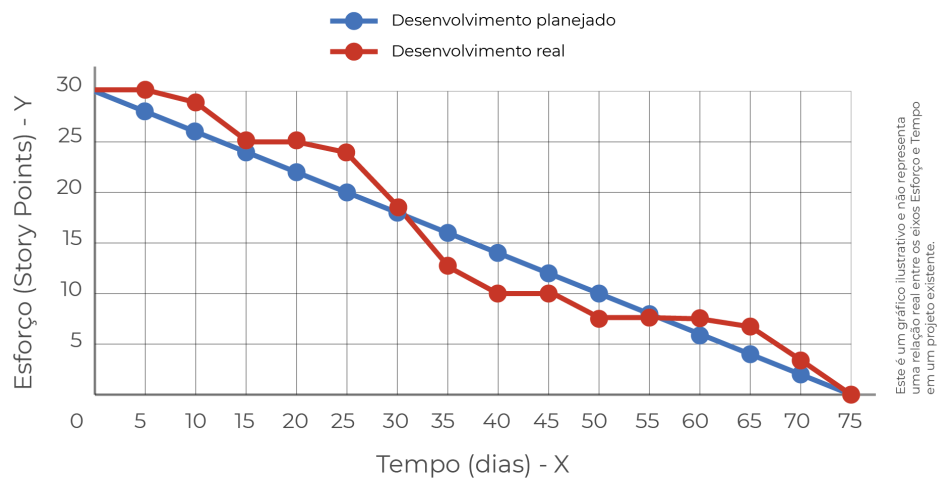


Figura 5 – Exemplo de Gráfico de *Burndown* retirado de (RIBEIRO, 2019)

Durante este trabalho houve a necessidade de adaptar o *framework* ágil Scrum para que este se adequasse ao contexto do projeto, visto que este será desenvolvido apenas pelo autor, o mesmo deverá desempenhar os papéis de um *Scrum Master*, *Product Owner* e da equipe de desenvolvimento.

2.2.2 Model Viewer Controller (MVC)

De acordo com (BASTOS, 2011), o MVC (*Model*, *View* e *Controller*) é uma arquitetura ou padrão que permite dividir as funcionalidades do sistema em camadas, de modo a facilitar resolução de um problema maior. O MVC possui três camadas básicas e cada uma delas apresenta características próprias e funções bem definidas para facilitar a reciclagem e manutenção do sistema, garantindo escalabilidade, reusabilidade e eficiência dos componentes.

Como descrito por (BASTOS, 2011), a primeira camada, chamada de *Model* (modelo), é utilizada para manipular informações de forma mais detalhada, sendo recomendado que, sempre que possível, se utilize dos modelos para realizar consultas, cálculos e todas as regras de negócio do sistema. É o modelo que tem acesso a toda e qualquer infor-

mação advinda de um banco de dados ou de arquivos JSON (*JavaScript Object Notation*) ou XML (*Extensible Markup Language*). Já a camada *View* (visão), é responsável por tudo que o usuário final visualiza, toda a interface, informação, não importando sua fonte de origem, é exibida graças a camada de visão. E por fim, a camada *Controller* (controladora), que como o nome já sugere, é responsável por controlar todo o fluxo de informação que passa pelo sistema. É na controladora que se decide “se”, “o que”, “quando” e “onde” deve funcionar. Ela define quais informações devem ser geradas, quais regras devem ser acionadas e para onde as informações devem ir. Em resumo, é a controladora que executa uma regra de negócio (modelo) e repassa a informação para a visualização (visão).

Devido ao fato de que nos dias de hoje os sistemas desenvolvidos tem uma complexidade alta, essa arquitetura tem como objetivo partilhar um problema complexo em várias partes simples de menor complexidade. Dessa forma, a dependência entre camadas deixa de ser um problema existente, visto que qualquer tipo de mudança em uma das camadas não irá afetar as demais, o que torna a atualização de layouts, alteração nas regras de negócio e adição de novos recursos, tarefas mais simples. E para projetos grandes, o MVC simplifica a divisão de tarefas entre equipes.

2.2.3 HTML5

De acordo com a definição apresentada por (PERRIER, 2013), HTML (*HiperText Markup Language* ou Linguagem de Marcação de Hipertexto) é o bloco de construção mais básico da *web*. Define o significado e a estrutura do conteúdo da *web*. O termo “Hipertexto” refere-se aos links que conectam páginas da *web* entre si, seja dentro de um único site ou entre sites. Links são um aspecto fundamental da *web*. Ao carregar conteúdo na Internet e vinculá-lo a páginas criadas por outras pessoas, o indivíduo se torna um participante ativo na *world wide web*.

O HTML5 é a mais recente evolução do padrão que define o HTML (ALMEIDA, 2012). O termo representa dois conceitos diferentes:

- É uma nova versão da linguagem HTML, com novos elementos, atributos, e comportamentos e um conjunto maior de tecnologias que permite o desenvolvimento de aplicações e web sites mais diversos e poderosos.
- Este conjunto é chamado “*HTML5 and friends*” e muitas vezes abreviado apenas como HTML5.

2.2.4 CSS3

CSS (*Cascading Style Sheets* ou Folhas de Estilo em Cascata) é uma linguagem de estilo usada para descrever a apresentação de um documento escrito em HTML ou em

XML. O CSS descreve como elementos são mostrados na tela, no papel, na fala ou em outras mídias. Desenvolvido em níveis, o CSS1 está atualmente obsoleto, o CSS2.1 é uma recomendação e o CSS3, agora dividido em pequenos módulos, está progredindo para a sua padronização (LIMA, 2012).

CSS3 é a mais nova versão de *Cascading Style Sheets*, onde se define estilos para páginas web com efeitos de transição, imagem, e outros, que dão um estilo novo às páginas web em todos os aspectos de design do layout. (PONTES, 2015).

A maioria dos navegadores (*browsers*) atuais já fornecem suporte a essa linguagem de estilo, como o Google Chrome, Opera, Internet Explorer 9, Safari e Mozilla Firefox. O CSS3 busca facilitar o trabalho de desenvolvedores web e também dos usuários, pela variedade de transformações na apresentação de um *website*.

2.2.5 JavaScript

Outro pilar muito importante do desenvolvimento *front-end* é o JavaScript, uma linguagem de programação que permite implementar funcionalidades mais complexas em páginas web (MILLS, 2016). Sempre que uma página web faz mais do que apenas mostrar informações estáticas (exibindo, por exemplo, informações em tempo real conteúdos atualizados, interativos, animações gráficas em 2D/3D, vídeos, etc), certamente o Javascript está envolvido.

A princípio o uso de JavaScript era puramente para programar funções incluídas em páginas HTML e que interagem com o DOM (*Document Object Model* ou Modelo de Objeto de Documentos), que representa a página de forma que os programas possam alterar a estrutura, o estilo e conteúdo do documento (NOAM, 2019). Alguns exemplos deste uso são:

- Abrir uma nova janela de diálogo com controle programático sobre seu tamanho, posição e atributos.
- Validar valores de um formulário para garantir que são aceitáveis antes de serem enviados ao servidor.
- Mudar imagens à medida que o mouse se movimenta sobre elas.

A utilização dessa linguagem será feita através do *framework* Angular, que será explicado na Seção 2.2.7.

2.2.6 Bootstrap 4

Segundo a página oficial do Bootstrap⁴, trata-se de um *framework*, originalmente criado por um designer e desenvolvedor do Twitter, e que tem se tornado a ferramenta de *front-end*, e um dos repositórios de código aberto mais popular do mundo (KRILL, 2013).

Originalmente publicado em 19 de agosto de 2011, a ferramenta vem sendo melhorada desde então, contando com mais de 20 versões, incluindo versões completamente reescritas. Com Bootstrap 2, por exemplo, o suporte à responsividade foi incluído de forma opcional (em um arquivo de estilização distinto). E na sequência, o Bootstrap 3 reescreveu a biblioteca mais uma vez, para torná-la responsiva por padrão.

Bootstrap é mantido por um time de fundadores e por um pequeno grupo de contribuidores, com um suporte e envolvimento massivos da comunidade, caracterizando um melhoramento contínuo conforme *feedbacks* são recebidos e novas versões são lançadas.

O Bootstrap 4 em relação a versão anterior conta com o recurso *Flexbox*, o qual facilita a criação e posicionamento de elementos em grade inteligentemente dimensionados. Outro recurso importante são os *Cards*, recursos visuais muito úteis para apresentação de informações. Os avanços desde a versão 3 são diversos, mas para esse projeto os mais impactantes foram os citados.

2.2.7 Angular

A definição encontrada na documentação presente no site do Angular⁵ o define como um *framework* de design de aplicação e uma plataforma de desenvolvimento para criar aplicações de página única (*single-page apps*) eficientes e sofisticadas.

A primeira versão do *framework* foi criada por Misko Hevery e Adams Abrons em 2009. Inicialmente, o projeto tinha como objetivo facilitar a criação de aplicações web. Alguns anos depois, Misko foi trabalhar no Google, dentro do projeto Google *Feedback*, e ele encontrou um cenário um pouco complicado: o projeto já contava com 17.000 linhas de código. Então ele apostou com seu gerente que reescreveria o código do projeto em 2 semanas, contanto que pudesse utilizar o *framework* que tinha desenvolvido. Eles terminaram de reescrever o código em 3 semanas, mas reduziu o projeto para 1.500 linhas. Com isso, aos poucos a Google acabou adotando esse *framework* para outros projetos internos entre 2014 e 2015 (GAVIGAN, 2018).

Pouco tempo depois do AngularJS (a primeira versão) ter sido publicadas, o lançamento da segunda versão do *framework* aconteceu, e pelo fato da segunda versão ser completamente diferente da versão inicial, o nome escolhido foi Angular 2, que atual-

⁴ <https://getbootstrap.com>

⁵ <https://angular.io>

mente é comumente chamado de Angular 2+ pelos desenvolvedores, devido às diversas novas versões lançadas após a versão 2.

Neste trabalho será utilizado o Angular 6 como *framework* Javascript, levando em conta que para todas as versões do Angular é necessário um conhecimento básico de HTML, CSS e Javascript, já que a ferramenta não passa de um facilitador para tarefas que seriam mais complexas se utilizássemos puramente as três tecnologias citadas.

2.2.8 Java

Java é uma linguagem de programação tipada, que utiliza do paradigma de programação orientada a objetos. De acordo com o site da linguagem Java⁶, mais de 9 milhões de desenvolvedores em todo o mundo programam em Java, demonstrando assim que é uma ferramenta amplamente utilizada. Ele foi projetado para permitir o desenvolvimento de aplicações portáteis de alto desempenho para a mais ampla variedade possível de plataformas de computação, e para o nosso caso utilizaremos ela para programação de rotinas no lado do servidor (*backend*).

2.2.9 Spring 2

O Spring é um *framework* de código aberto (*open source*) que visa atender demandas de desenvolvimento de forma rápida dentro da própria aplicação, não havendo necessidade, na grande maioria dos casos, de intervenções externas (ROCHA, 2018). É largamente utilizado em aplicações Java por ter uma arquitetura simplificada baseada em interfaces e POJOS (*Plain Old Java Objects*), pelo seu reconhecido bom desempenho e mecanismo de injeção de dependência, bem como uma lista de módulos extensa.

No desenvolvimento de nossa aplicação serão usados os seguintes módulos do Spring:

- spring-jdbc: módulo que torna mais simples a comunicação com banco de dados.
- spring-security: responsável por diminuir a complexidade nos processos de autorização e autenticação de usuários.
- spring-data: utilizado para facilitar a comunicação com diferentes bancos de dados não relacionais.

2.2.10 PostgreSQL

O PostgreSQL é um sistema de gerenciamento de banco de dados objeto-relacional (SGBDOR - sistema de gerenciamento de banco de dados objeto-relacional) criado a

⁶ <https://www.java.com/pt-BR/about/>

partir do POSTGRES Versão 4.2 que foi desenvolvido pelo Departamento de Ciência da Computação da Universidade da Califórnia em Berkeley. O POSTGRES foi o primeiro SGBDOR a apresentar vários conceitos que se tornaram disponíveis posteriormente em alguns sistemas de banco de dados comerciais.

De acordo com a tradução da documentação do PostgreSQL⁷, devido à sua licença gratuita e open-source, o PostgreSQL pode ser utilizado, modificado e distribuído por qualquer pessoa para qualquer finalidade, seja particular, comercial ou acadêmica, livre de encargos. Esse tipo de disponibilidade faz com que esse banco de dados seja ofertado em diversos serviços de hospedagem em nuvem, já que ele é um banco de dados gratuito. O que está em acordo com desenvolvimento de aplicações para portfólio pessoal ou para serem usadas academicamente, como é o caso do TaskMe.

⁷ <http://pgdocptbr.sourceforge.net/pg82/intro-what-is.html#FTN.AEN62>

3 TaskMe

Este capítulo apresenta o TaskMe, aplicativo desenvolvido para potencializar a organização do trabalho para que os contratados não se percam em meio a diversos serviços a serem prestados, acelerar o *feedback* através de um chat interno da aplicação e aumentar a integração entre participantes de uma mesma tarefa. Inicialmente serão apresentadas algumas considerações do processo de desenvolvimento da aplicação (Seção 3.1). Posteriormente, são apresentados os detalhes da modelagem do sistema (Seção 3.2) e a descrição do sistema como um todo através de casos de uso (Seção 3.3).

3.1 Desenvolvimento da aplicação

Diversas funcionalidades e características da aplicação tiveram como motivação verificações de apresentações e fluxos das aplicações de mercado correlatas. Com base nelas, criamos mecanismos que tornassem o *feedback* (ao criador da tarefa) mais rápido, através do chat interno da aplicação e também telas para organização e apresentação das tarefas atuais dos prestadores mais elaborada, funcionalidade que não foi encontrada nas demais plataformas correlatas.

A aplicação foi desenvolvida utilizando as tecnologias apresentadas na Seção 2.2, e sobre a organização do desenvolvimento da aplicação utilizando Scrum, foram demandados 9 sprints de 2 semanas para concluir as 27 histórias propostas (3 histórias por *Sprint*). A seguir as Figuras 6 e 7, são exemplos dessas histórias no JIRA (ferramenta que permite o monitoramento de tarefas e acompanhamento de projetos).

O modelo de arquitetura MVC foi de grande valia no desenvolvimento do *back-end*, privando a aplicação cliente de receber dados sensíveis em um JSON retornado ao solicitar informações atualizadas do usuário para, por exemplo, a tela de edição de perfil (*Edit Profile*) da aplicação.

Projetos /  Task Me /  TaskMe /  TM-25

Adicionar campo na entidade User e criar endpoints para user-rating - Back-end

 Anexar  Adicionar item secundário  Vincular item 

Descrição

Adicionar campos na entidade de usuário, que iram ser utilizados para o cálculo de sua nota como prestador de serviço.

RATE_SUM O primeiro campo guardará a soma de todas as notas que já foram dadas a este usuário;

NUM_RATES O segundo campo guardará a quantidade de vezes que esse usuário recebeu uma classificação;

E então será possível, calcular a classificação atual do usuário através da seguinte operação:

CURRENT_RATE = **RATE_SUM**/**NUM_RATES**

Figura 6 – Estória para criação de novos campos na entidade usuário, que serão utilizadas no cálculo da classificação como prestador

Projetos /  Task Me /  TaskMe /  TM-27

Criação da Main Screen- Front-end

 Anexar  Adicionar item secundário  Vincular item 

Descrição

Criar a main screen para a aplicação. Não é necessário consultar a API, será somente uma tela de apresentação das funcionalidades do sistema. Utilize o desenho anexado à tarefa pra se basear.

Anexos (1)

... +

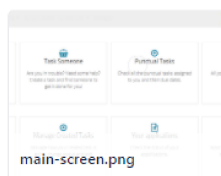


Figura 7 – Estória para criação da tela principal do sistema

3.2 Modelagem do sistema

Para se alcançar o sucesso no desenvolvimento de um software, um dos elementos que contribuem para isso é a utilização da modelagem. Um modelo é uma simplificação da realidade e os modelos podem realizar planos detalhados, assim como planos mais gerais com uma visão panorâmica do sistema. Qualquer projeto será beneficiado pelo uso de algum tipo de modelagem. Os modelos auxiliam a equipe a ter uma visão mais abrangente do funcionamento do sistema, e assim, desenvolvê-lo de forma mais rápida e correta (ESPÍNDOLA, 2007).

3.2.1 Diagramas de entidade relacionamento (DER)

O Modelo Entidade Relacionamento (também chamado Modelo ER, ou simplesmente MER), é um modelo conceitual utilizado para descrever os objetos (entidades) envolvidos em um domínio de negócios, com suas características (atributos) e como elas se relacionam entre si (relacionamentos).

Mas enquanto ele é um modelo conceitual, o Diagrama Entidade Relacionamento (Diagrama ER ou ainda DER) é a sua representação gráfica e principal ferramenta. Em situações práticas, o diagrama é tido muitas vezes como sinônimo de modelo, uma vez que sem uma forma de visualizar as informações, o modelo pode ficar abstrato demais para auxiliar no desenvolvimento do sistema. Dessa forma, quando se está modelando um domínio, o mais comum é já criar sua representação gráfica (RODRIGUES, 2014). As Figuras 8 e 9 apresentam diferentes estilos de diagramas entidade relacionamento (DER) da aplicação desenvolvida, no qual é possível visualizar a representação das entidades e seus relacionamentos no banco de dados.

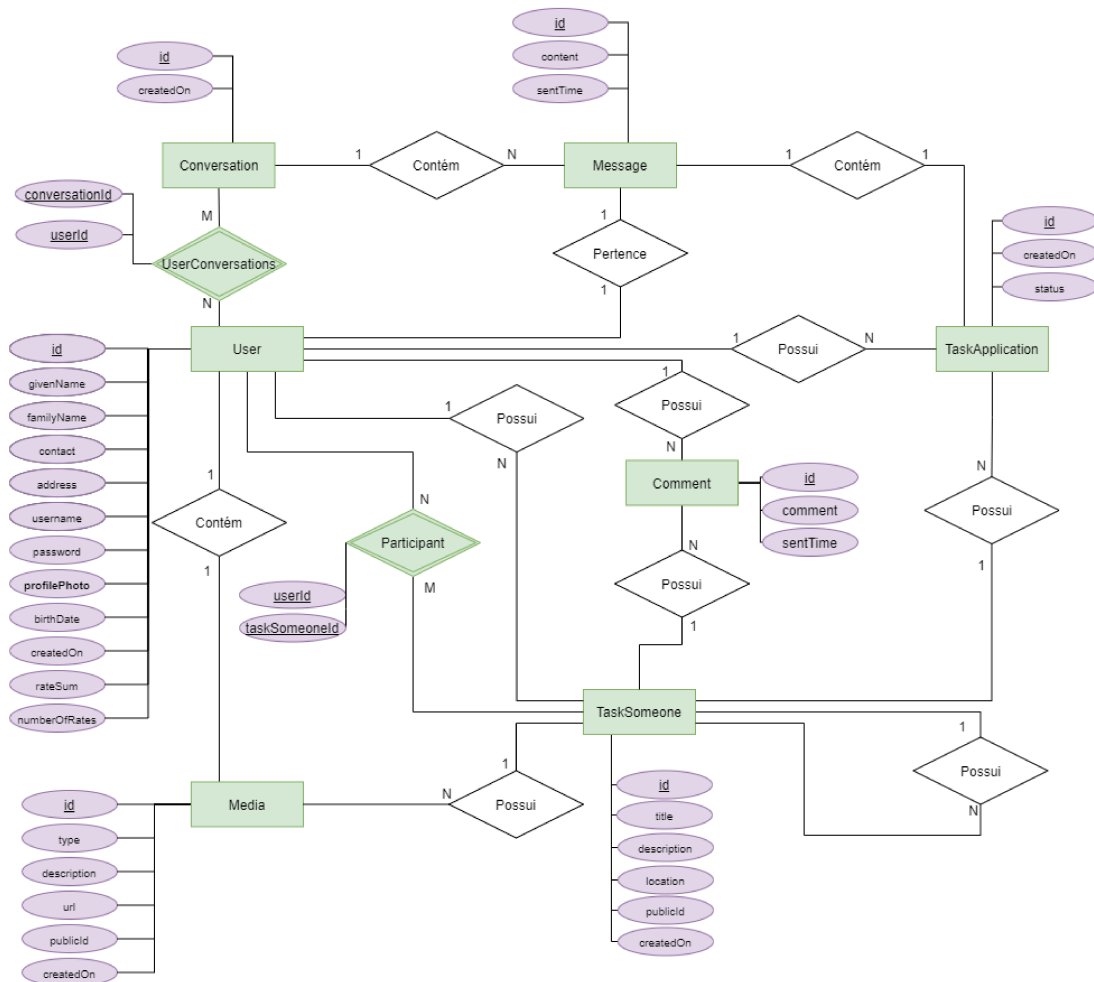


Figura 8 – Diagrama de entidade relacionamento estilo de notação Chen

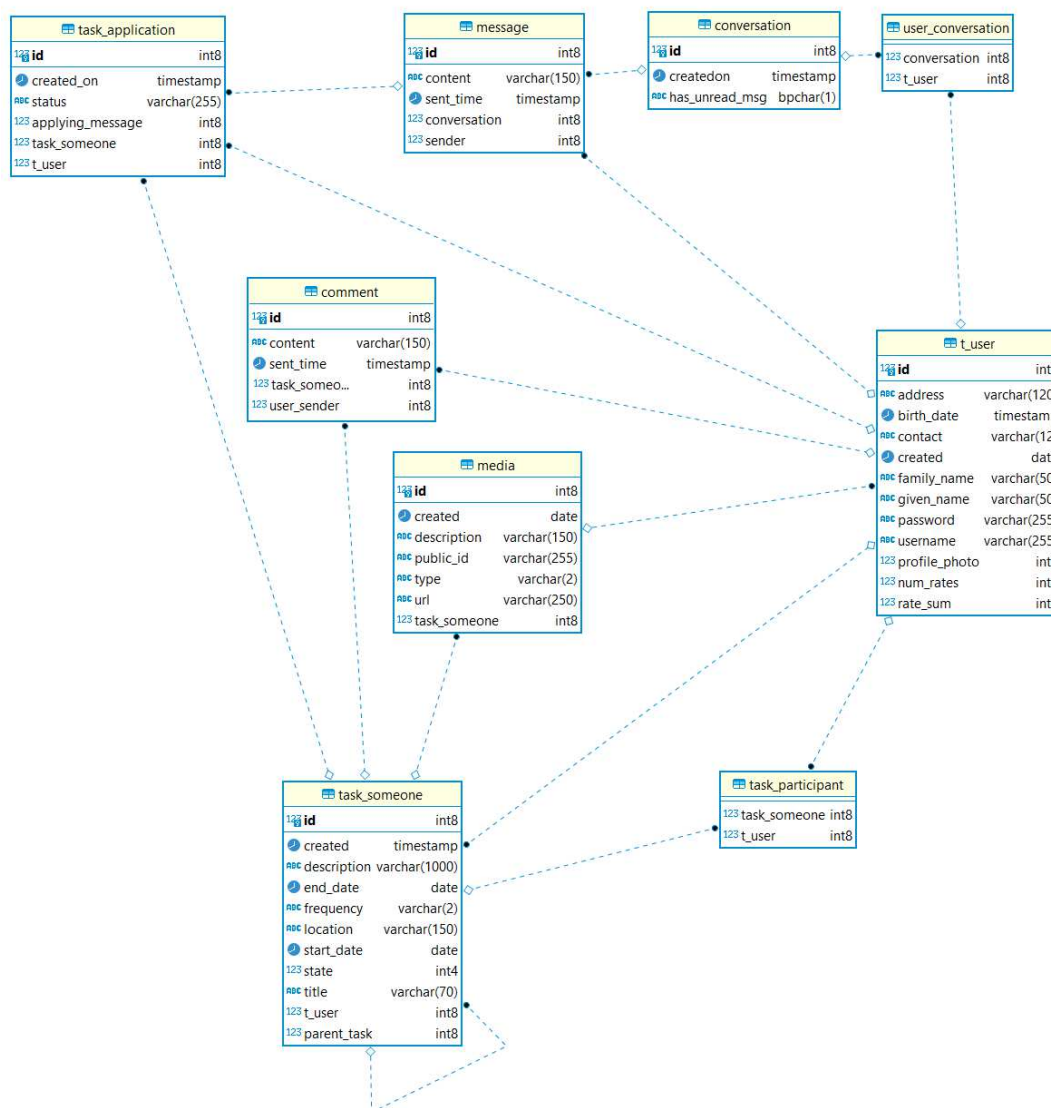


Figura 9 – Diagrama de entidade relacionamento estilo IDEF1X (*Integration Definition for Information Modeling*)

3.2.2 Diagramas de sequência

Uma maneira de representar fluxo de utilização do sistema é através de diagramas de sequência. Seu principal objetivo é demonstrar em linhas de tempo quais são as interações entre os objetos de um determinado cenário representado pelo diagrama. Este diagrama possui dois eixos: o eixo vertical que indica a sequência das mensagens e o tempo de vida dos objetos e o eixo horizontal, este último indica quais são os objetos que participam do diagrama (OLIVEIRA, 2013).

Os fluxos representados nos diagramas a seguir são o de registro na aplicação (Figura 10) e de criação de uma nova tarefa com *upload* de imagens (Figura 11).

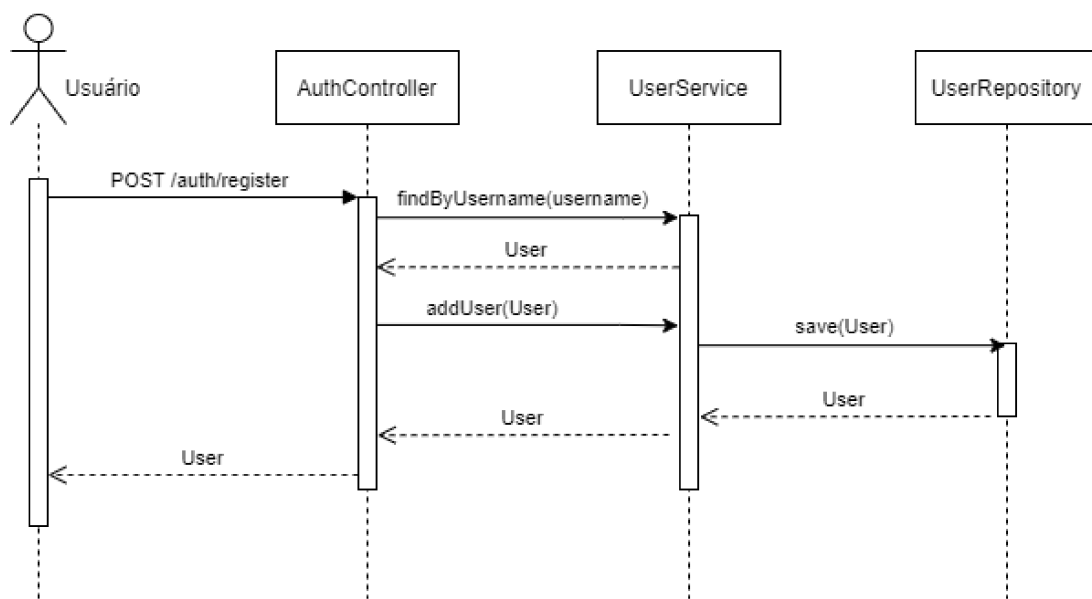


Figura 10 – Diagrama de sequência - Registro de usuário

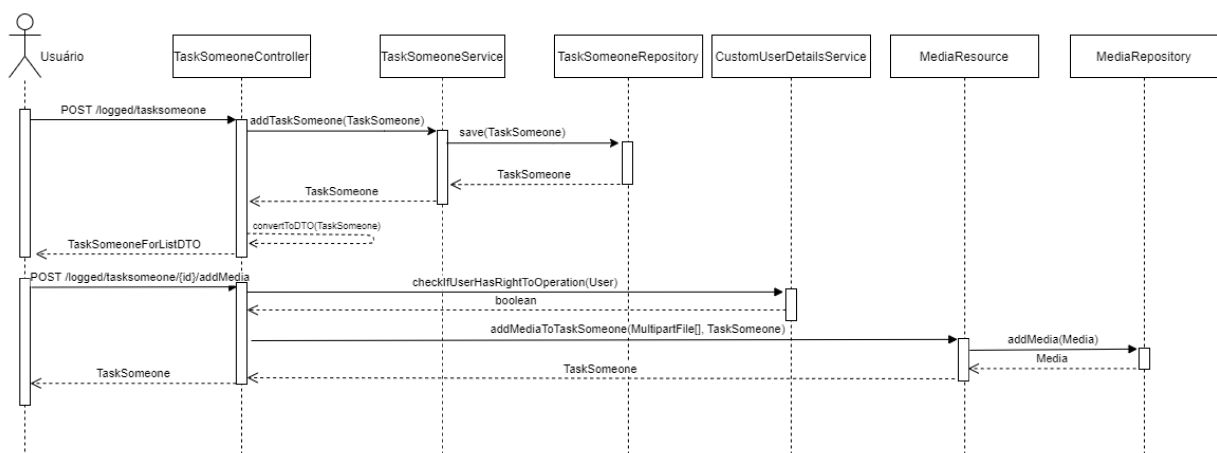


Figura 11 – Diagrama de sequência - Criação de uma nova tarefa com upload de imagem

3.3 Descrição do sistema

A descrição do sistema apresenta a forma de uso por meio de suas telas, ou seja, a interação do usuário. Essa descrição é feita por meio das funcionalidades principais do sistema e é apresentada a seguir.

3.3.1 Registro de Usuário

A funcionalidade descrita no diagrama apresentado na Figura 10 segue o seguinte fluxo na interface de usuário. Na página inicial da aplicação, logo abaixo da introdução, existe um botão onde o usuário pode clicar para se registrar, como mostra a Figura 12.

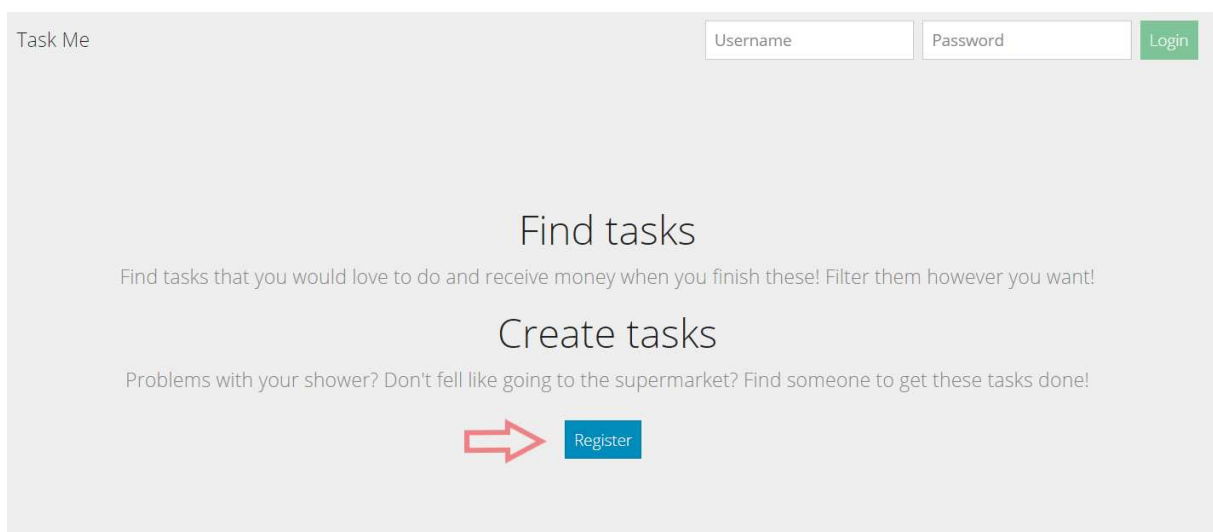


Figura 12 – Tela inicial da aplicação *TaskMe*

Ao clicar no botão de registro, um formulário será mostrado (Figura 13), e então o usuário poderá fazer seu cadastro preenchendo-o com suas informações pessoais e uma descrição que será exibida caso ele se candidate para alguma tarefa.

Quando o cadastro é finalizado, o usuário será logado automaticamente e redirecionado para a página principal da aplicação, onde terá ao seu dispor, todas as funcionalidades do sistema (Figura 14).

3.3.2 Alterando as informações de perfil e adicionando foto

Ainda relacionado com o perfil do usuário, a tela para alteração do mesmo está acessível através da funcionalidade *Profile*, presente na Figura 14 ou através do *dropdown* no canto superior direito (Figura 15), possibilitando fazer alterações em seus dados e até mesmo adicionar uma imagem ao seu perfil.

Sign up

Given Name	Family Name
Laura	Hilarino Campos
Username	Password
laura	
Address	
Rua Francisco Sales, 155, Uberlândia, MG	
Contact	Birth date
34992629321	27/09/1995
Description	
Está cansado e precisa de ajuda nas tarefas cotidianas como ir ao mercado ou buscar os meninos na escola? Eu posso te ajudar! Problemas com encanamentos em casa? Tenho 4 anos de experiência com encanamentos e hidráulica! Basta me mandar uma mensagem! 251/1000	

Figura 13 – Formulário de cadastro de usuário

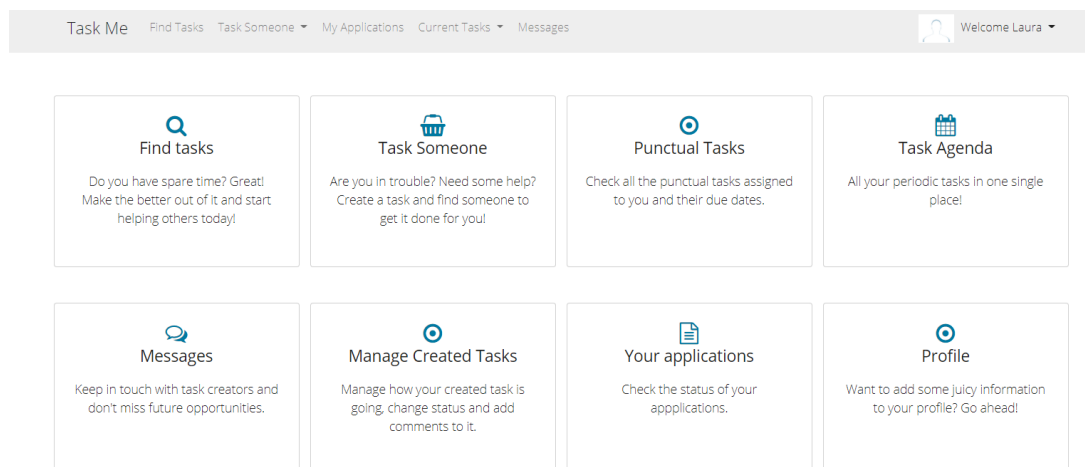


Figura 14 – Página principal da aplicação

Clicando na imagem de seu perfil indicada na Figura 16, é possível que o usuário faça *upload* de uma nova foto e então tenha seu perfil o mais atualizado possível (Figura 17).

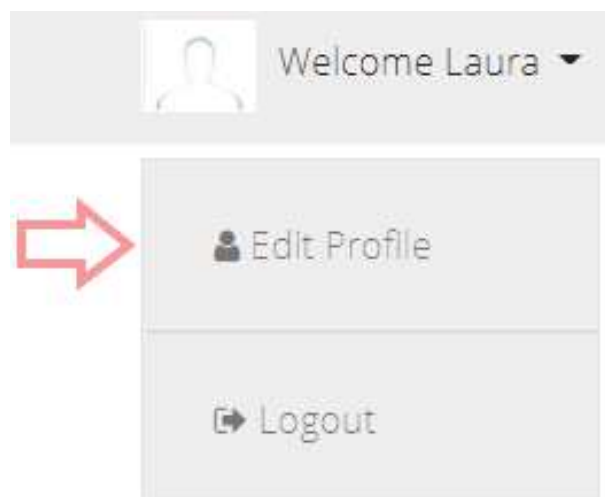
Figura 15 – Opções do *dropdown*A screenshot of the 'Edit Profile' form. The form is titled 'Edit Profile' in blue. It contains several input fields: 'Given Name' (filled with 'Laura'), 'Family Name' (filled with 'Hilarino Campos'), 'Username' (filled with 'laura'), 'Password' (filled with '*****' and a password strength indicator), 'Birth date' (filled with '27/09/1995'), 'Contact' (filled with '34992629321'), and 'Address' (filled with 'Rua Francisco Sales, 155, Uberlândia, MG'). There is also a 'Description' text area with two lines of text: 'Está cansado e precisa de ajuda nas tarefas cotidianas como ir ao mercado ou buscar os meninos na escola? Eu posso te ajudar!' and 'Problemas com encanamentos em casa? Tenho 4 anos de experiência com encanamentos e hidráulica! Basta me mandar uma mensagem!'. A 'Save Changes' button is at the bottom right. A red arrow points to the 'Given Name' field.

Figura 16 – Formulário de cadastro de usuário

3.3.3 Fluxo de criação e completude de tarefas pontuais

Agora que o usuário realizou seu cadastro ele pode se candidatar a realizar tarefas ou, se caso precise, ele pode criar uma tarefa para que algum usuário possa concluí-la. No exemplo a seguir, a recém cadastrada usuária teve problemas com sua geladeira, e precisa que alguém a arrume. Então ela irá criar uma nova tarefa, através da opção *Task Someone* na tela principal (Figura 14), ou utilizando a barra de navegação, como mostra a Figura 18

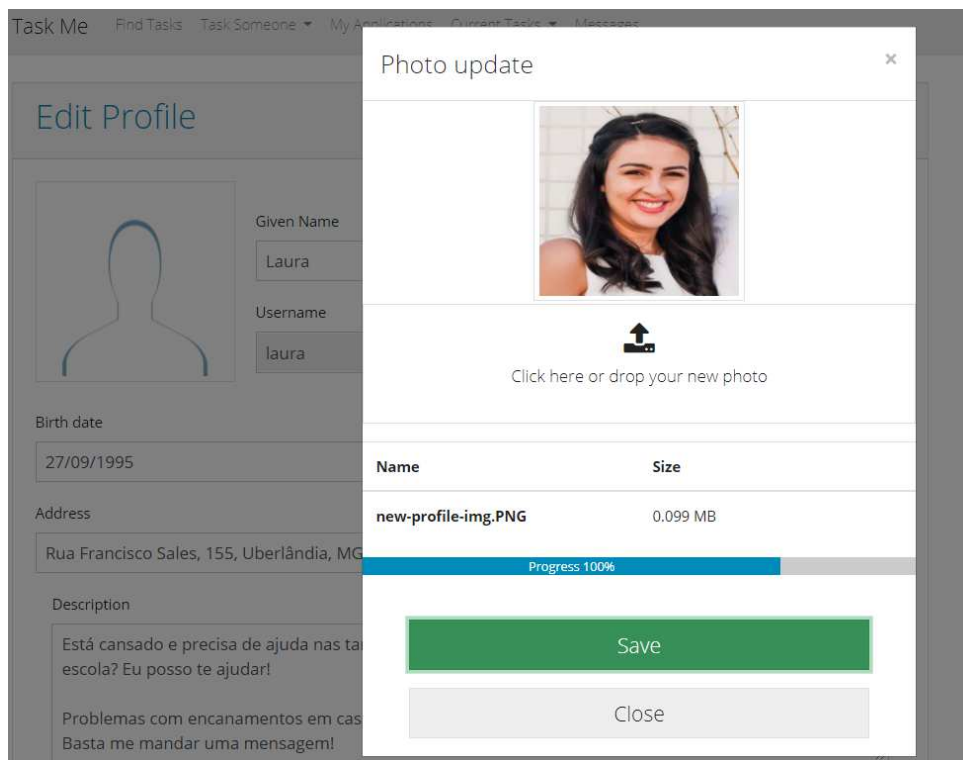
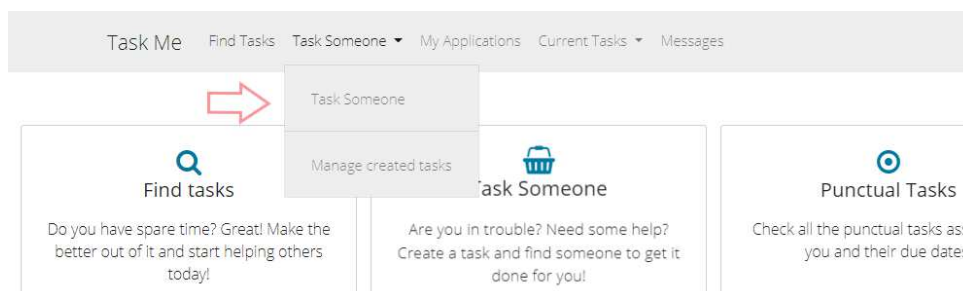


Figura 17 – Upload de nova foto de perfil

Figura 18 – Acesso a tela *Task Someone* através da barra de navegação da aplicação

Acessando a tela, o usuário deve preencher os dados, como mostra a Figura 19. Um detalhe importante desse preenchimento é que o campo *Repeat* não está marcado, ou seja, essa tarefa é uma tarefa que ocorrerá somente uma vez, ou em outras palavras, será uma tarefa **pontual**.

Após isso, o usuário pressiona o botão *Next*, para avançar para a próxima etapa. Nela o usuário poderá adicionar imagens para que os interessados em cumprir a tarefa tenham o máximo de detalhes possíveis, ou caso não haja necessidade de fazer upload de imagens, o usuário pode clicar no botão *Skip*, para concluir a criação da tarefa. Como o usuário em questão teve um problema em sua geladeira, é importante que ele adicione algumas fotos, como mostra a Figura 20.

Posteriormente a esse passo, o usuário será redirecionado para a tela de detalhes

Task Me

Find Tasks

Task Someone ▾

My Applications

Current Tasks ▾

Messages

 Welcome Laura ▾

Task Someone

Title

Geladeira não está parecendo ligada

Repeat ☐

Due date

16/12/2020

Description

Minha geladeira parou de funcionar de ontem pra hoje. Não acende a luz interna, não faz barulho, nada! Tenho ela a apenas 6 meses!
Vários alimentos estão dentro dela e preciso que ela volte a funcionar.

201/1000

Address

Rua Francisco Sales, 155, Uberlândia, MG

Next

Figura 19 – Formulário para criação de tarefa

Task Someone

Media

The more details you provide, the faster you will find someone to accomplish your task!

 Click here or drop your files



geladeira-1.jpg
0.06 MB



geladeira-2.jpg
0.02 MB



geladeira-3.jpg
0.018 MB

Remove all

Upload all

Skip

Figura 20 – Upload de imagens relacionadas à tarefa

da tarefa (Figura 21), onde ele poderá verificar como ficou a tarefa após sua criação.

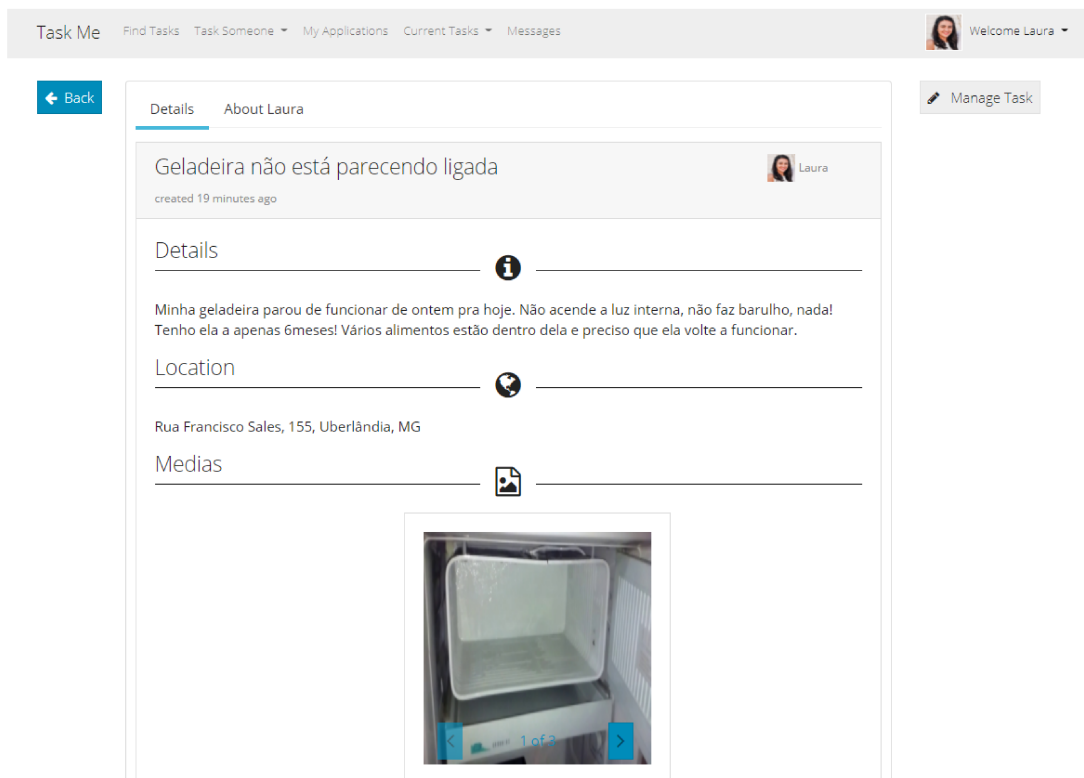


Figura 21 – Tela de detalhes da tarefa

Caso o usuário queira fazer alterações na tarefa, ele pode acessar a tela com todas suas tarefas criadas através do botão *Manage Task*, localizado no canto direito superior da tela apresentada na Figura 21. Ao acessar a tela através desse botão, o sistema fará um filtro para que o usuário veja somente a tarefa que quer editar na lista. E então o usuário pode editar os detalhes da tarefa clicando no botão *Edit information*, indicado na Figura 22. Ao fazê-lo, o sistema exibirá um diálogo (Figura 23), onde será possível alterar as informações da tarefa e adicionar/remover imagens também.

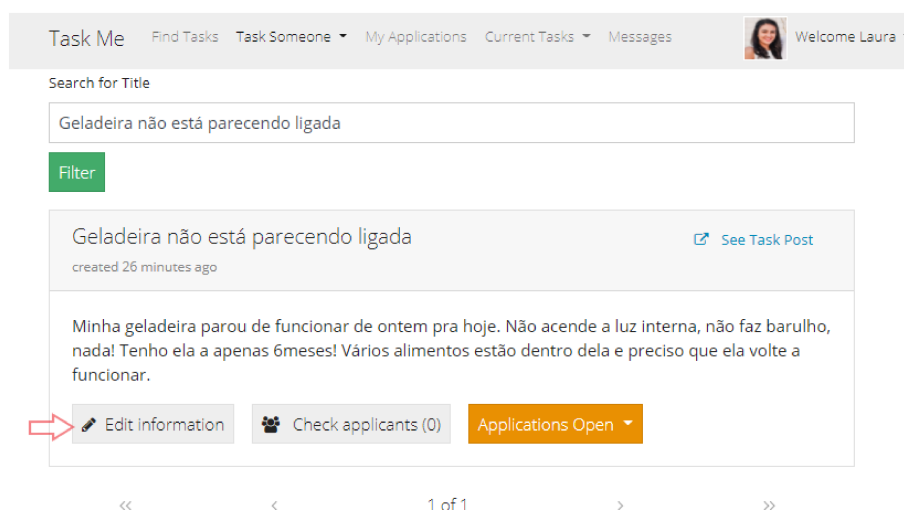
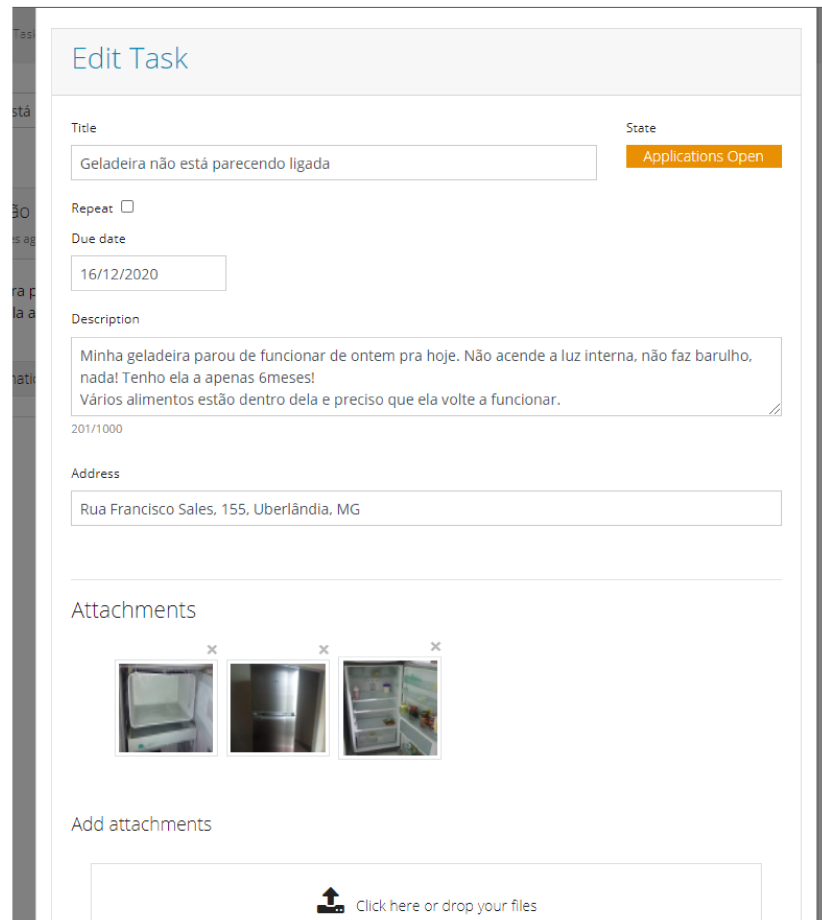


Figura 22 – Tela de tasks criadas pelo usuário



Edit Task

Title: Geladeira não está parecendo ligada

State: Applications Open

Repeat: ☐

Due date: 16/12/2020

Description: Minha geladeira parou de funcionar de ontem pra hoje. Não acende a luz interna, não faz barulho, nada! Tenho ela a apenas 6 meses! Vários alimentos estão dentro dela e preciso que ela volte a funcionar.

201/1000

Address: Rua Francisco Sales, 155, Uberlândia, MG

Attachments

Add attachments

Click here or drop your files

Figura 23 – Diálogo para alteração de informações da tarefa

Nas Figuras 22 e 23, podemos verificar que a situação da tarefa está como *Applications Open*, pois o sistema já define a situação da tarefa como **aberta para aplicações** no momento em que o usuário cria a tarefa, isso faz com que a tarefa seja exibida para outros usuários na tela *Find Tasks*, como mostra a Figura 24.

Como essa tarefa está disponível para os demais usuários, um usuário interessado, ao acessar os detalhes dessa tarefa, verá o botão *I'm interested!* (Figura 25), que ao ser clicado, irá exibir um diálogo com uma mensagem padrão a ser enviada para o criador da tarefa (Figura 26), mas caso ache necessário o usuário poderá alterar a mensagem.

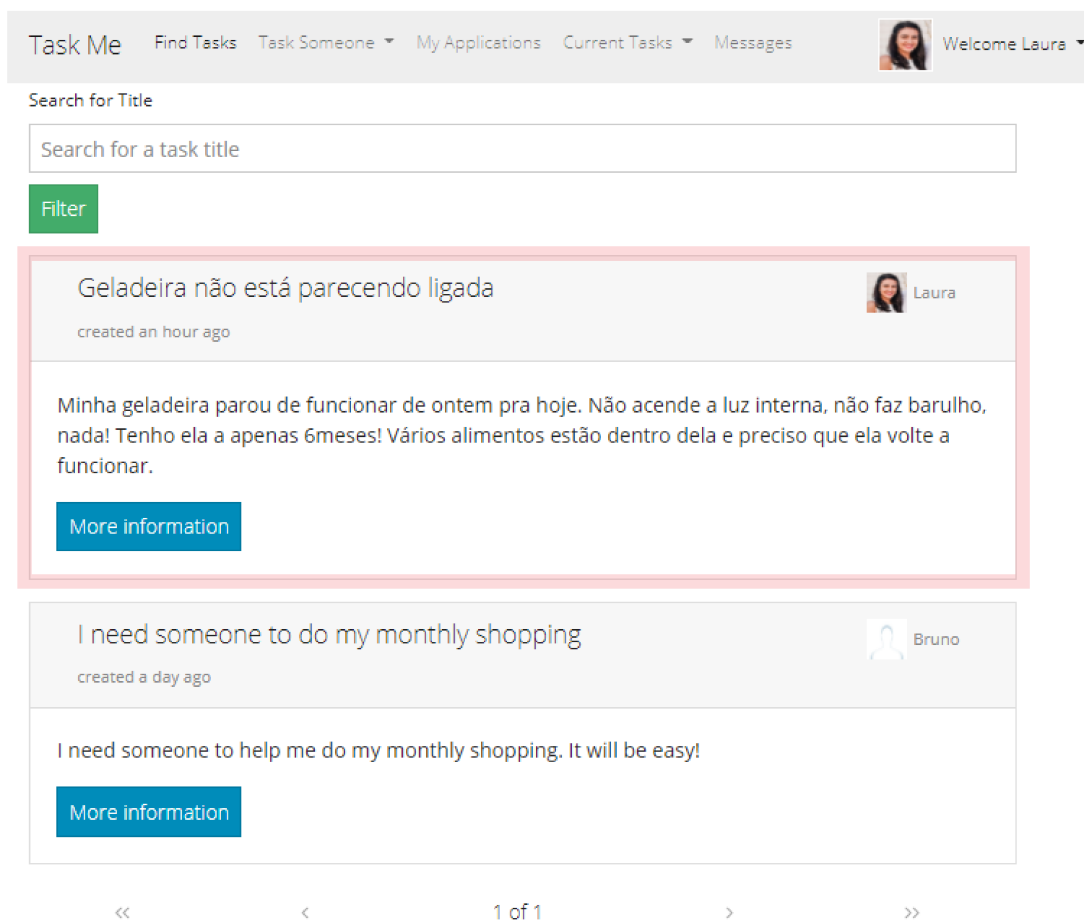
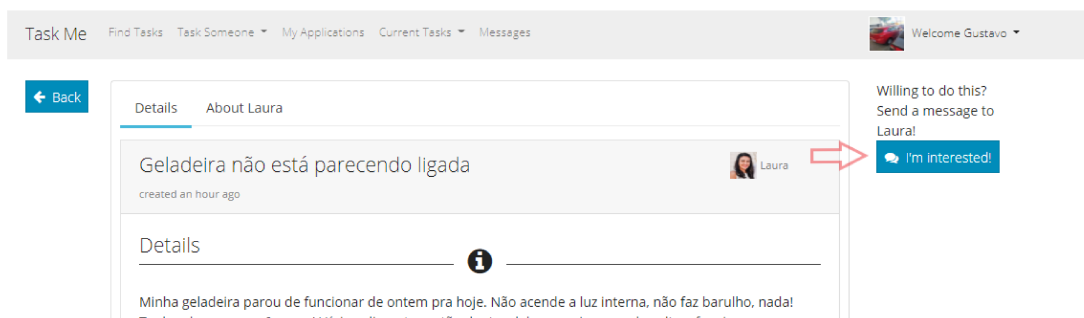
Figura 24 – Nova tarefa sendo exibida na tela *Find Tasks*

Figura 25 – Botão de interessado na tela de detalhes da tarefa

Ao confirmar o envio da mensagem o usuário terá se aplicado para aquela tarefa e será informado que ele pode verificar todas as suas aplicações na tela *Your Applications*, como mostra a Figura 27.

Acessando a tela *Your Applications* o usuário tem ao seu dispor todas as aplicações e suas respectivas situações (Figura 28), podendo cancelar a aplicação, se ele não estiver mais interessado em participar da tarefa. É possível também navegar até a tela de mensagem através do botão *Messages*. Assim, o usuário é redirecionado para a tela de mensagens (Figura 29) para que possa verificar a conversa gerada pela mensagem de

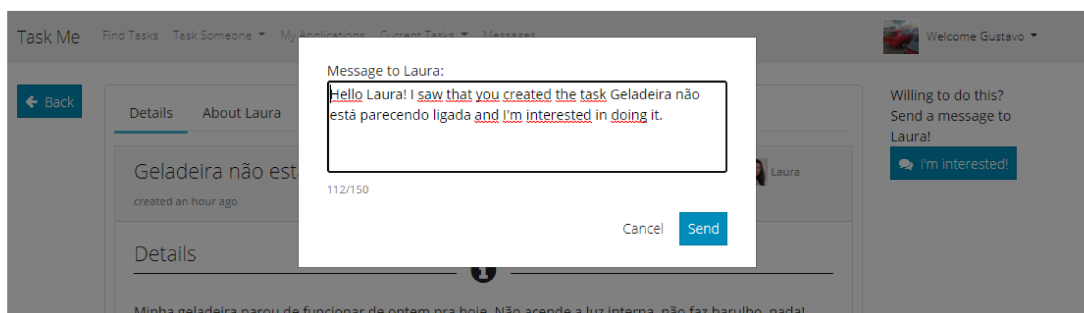
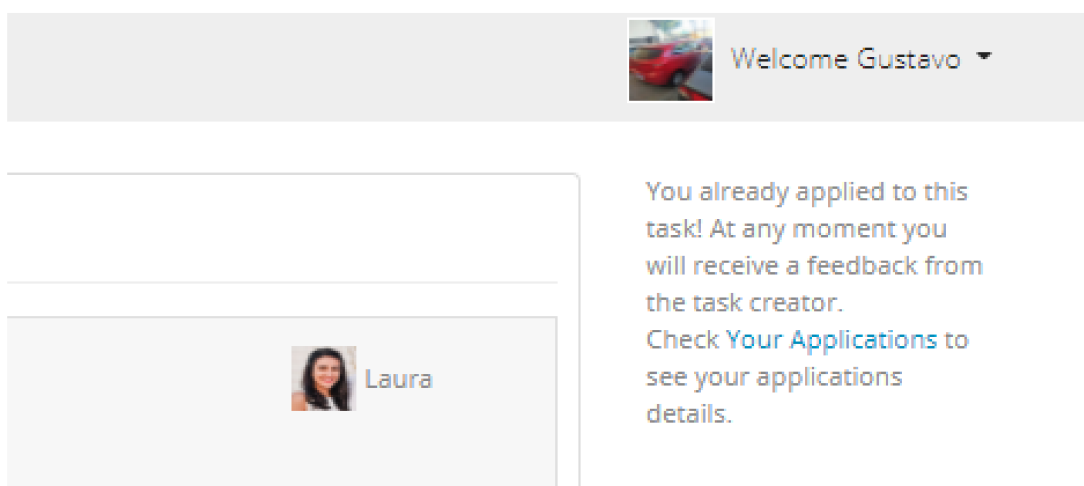
Figura 26 – Nova tarefa sendo exibida na tela *Find Tasks*

Figura 27 – Dica ao usuário - acesse suas aplicações

interesse na tarefa em que ele se aplicou e enviar mais mensagens, se necessário.

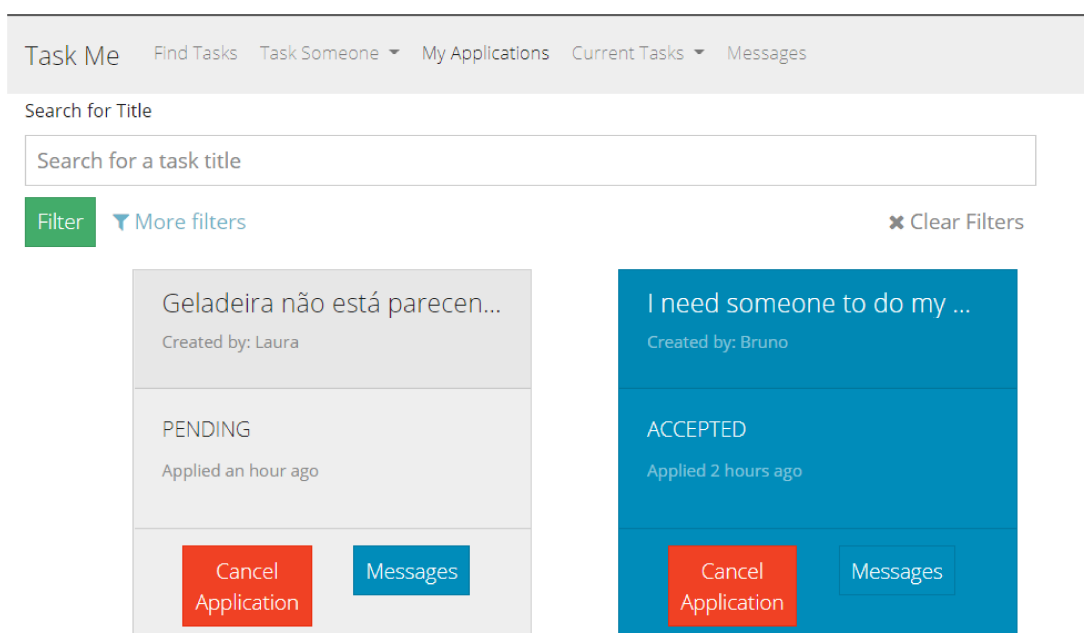


Figura 28 – Tela de aplicações do usuário

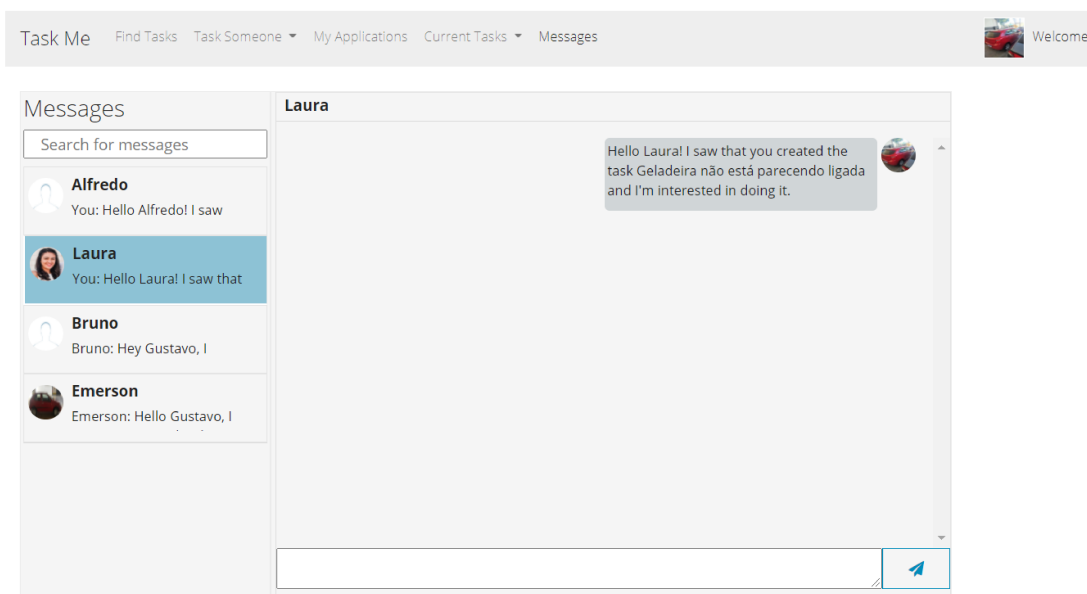


Figura 29 – Tela de mensagens

Na perspectiva da criadora da tarefa, a filtragem e seleção de aplicantes qualificados para tarefa é feita na tela de tarefas criadas pelo usuário (apresentada anteriormente pela Figura 22). Nesta mesma tela é possível observar três botões para cada tarefa. Mas para esse processo de refinamento de candidatos, será focado nos dois botões apontados pela Figura 30.



Figura 30 – Botões para visualização de candidatos (1) e alteração da situação da tarefa (2) na tela de gerenciamento de tarefas criadas

O botão número um, como o nome sugere (*Check applicants*), é utilizado para que a criadora da tarefa possa ver todos os usuários que se candidataram a fazer a tarefa criada por ela. Ao clicar nesse botão é apresentado um diálogo contendo todos os candidatos e suas respectivas classificações (de zero estrelas até 5 estrelas), e ao clicar no nome do candidato, o perfil completo será exibido para que seja feita uma análise mais minuciosa no perfil do candidato (Figura 31).

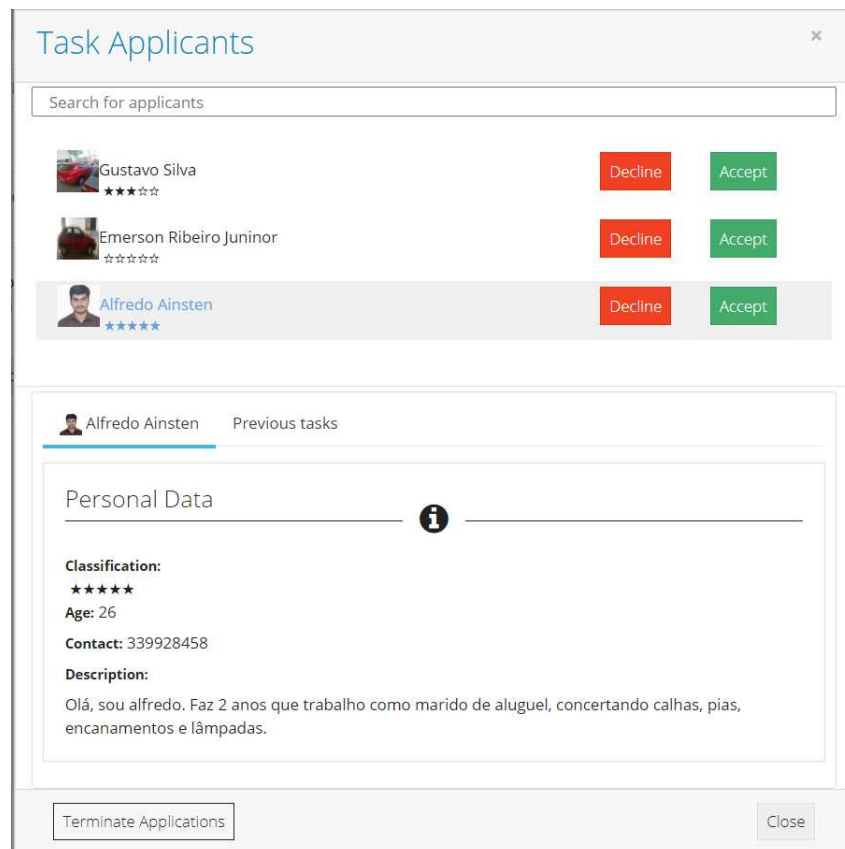


Figura 31 – Diálogo de exibição de candidatos da tarefa

Assim que seja encontrado o melhor candidato para realizar uma tarefa, basta clicar no botão *Accept*, que um diálogo contendo uma mensagem padrão irá ser exibido, para que o usuário possa verificar se a mensagem o satisfaz, e se for preciso, alterá-la (Figura 32). Ao clicar em enviar o usuário deverá confirmar se deseja aprovar aquele candidato para concluir a aprovação.

Caso o candidato não seja o melhor para realizar aquela tarefa, a criadora pode clicar em *Decline* para que um diálogo semelhante ao apresentado na figura acima (porém com uma mensagem de não aprovação) seja apresentado, ou basta apenas não clicar em nenhum botão, mas nesse caso o usuário não receberá nenhum *feedback* de que não será o realizador da tarefa, o que pode gerar expectativas. A mensagem de não aprovação é algo optativo pois pode haver tarefas com 300 participantes, por exemplo, e enviar essas mensagens pode tomar muito tempo do criador da tarefa. Ao final do refinamento de

candidatos, chegamos a um resultado semelhante ao da Figura 33.

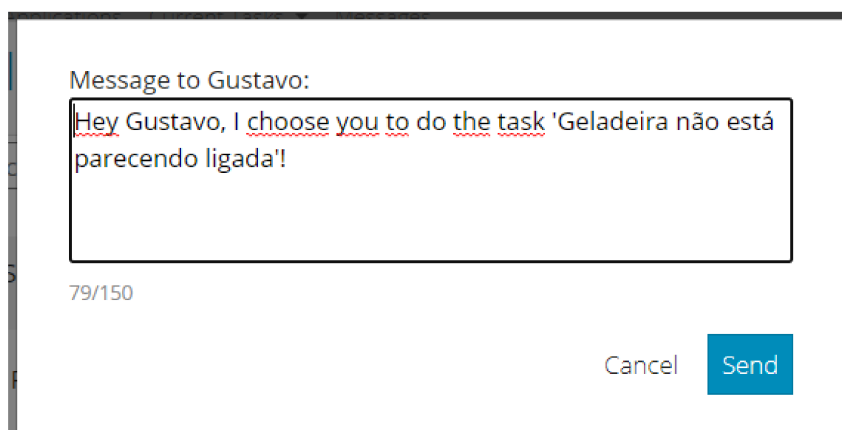


Figura 32 – Diálogo de mensagem de aprovação

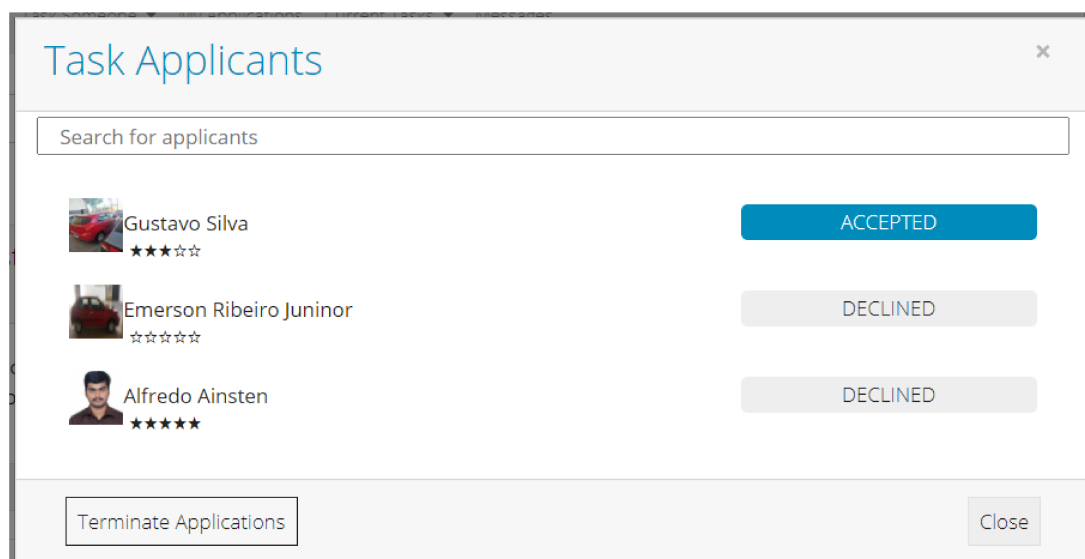


Figura 33 – Diálogo de aplicantes após refinamento de candidatos

Agora o usuário pode encerrar as aplicações de duas maneiras, clicando no botão à esquerda inferior do diálogo de aplicantes (Figura 33), ou através do segundo botão marcado na Figura 30, que será explicado mais a frente. Ao encerrar as aplicações, todas as candidaturas aprovadas (que no nosso exemplo foi apenas uma, mas poderiam ser diversas) irão ter seus aplicantes inseridos nas tarefas, agora como participantes desta. E o botão *Check applicants* da tela gerenciamento de tarefas criadas pelo usuário (Figura 30) passa a ser *Check participants* e o diálogo a mostrar apenas os participantes da tarefa, com um atalho para o *chat* com cada um deles (Figura 34).

Voltando na Figura 30, o botão número dois é capaz de alterar o estado da tarefa para o próximo ou para um estado anterior. Se o estado atual, por exemplo, está como aplicações encerradas (*Applications Closed*), e ao clicar no botão (que na verdade é um *dropdown*), temos que a primeira opção será o próximo estado (Iniciar a tarefa), em

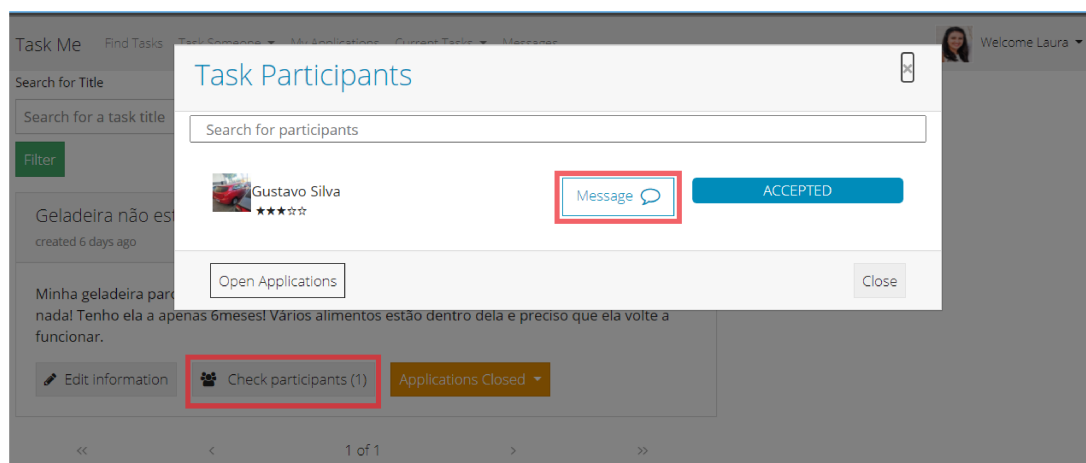


Figura 34 – Diálogo de participantes da tarefa

segundo o estado anterior (abrir para novos aplicantes) e em terceiro a opção de cancelar a tarefa. A Figura 35 ilustra o cenário apresentado.

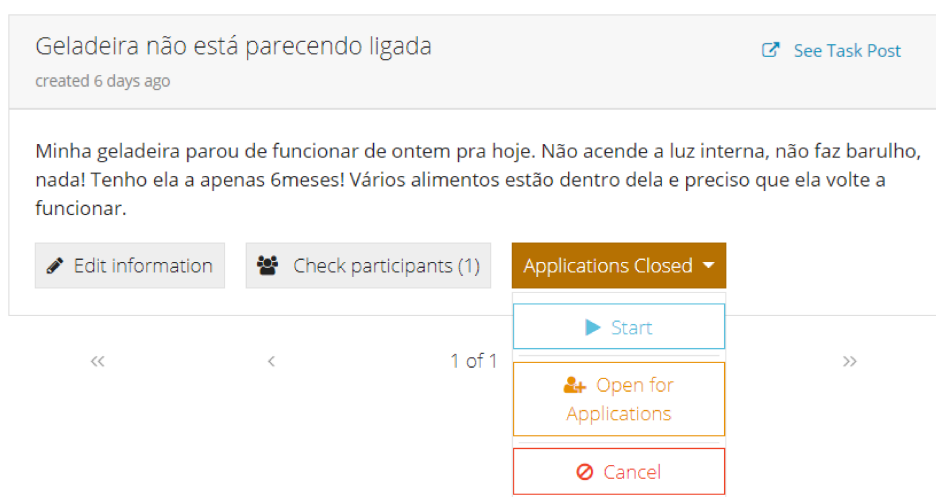


Figura 35 – Opções de mudança de estado da tarefa

Encerrando-se as candidaturas à tarefa, os usuários que foram aprovados poderão verificar a tarefa através do menu *Current Tasks » Punctual Tasks*, devido ao fato de ser uma tarefa de ocorrência única (pontual). Nessa tela é possível ver qual a *deadline*, ou data limite, para o cumprimento dessa tarefa e dependendo de quanto mais perto estejamos da data limite a tarefa sofre uma mudança de cor (verde para um risco leve, amarelo para um risco que requer atenção e vermelho para alto risco) (Figura 36). Ao clicar nas tarefas, o usuário é levado para a tela de detalhes da tarefa.

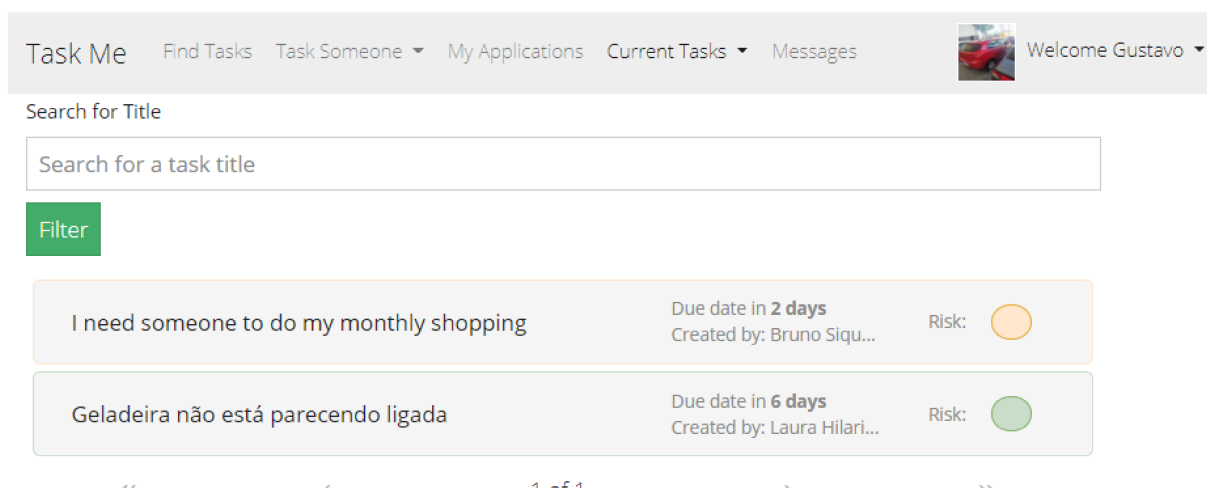


Figura 36 – Tela de tarefas pontuais

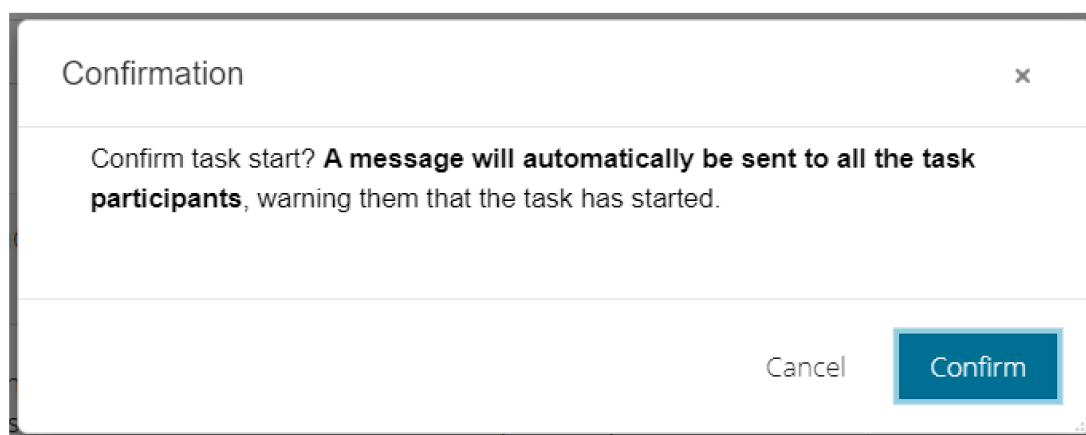


Figura 37 – Confirmação de início de tarefa

Ao iniciar a tarefa por meio da opção *Start* do botão de mudança de estado apresentado na Figura 35, uma mensagem de confirmação é apresentada (Figura 37). A partir deste momento, o participante da tarefa então verifica os detalhes do problema utilizando o *chat* interno da aplicação para se comunicar com a criadora da tarefa, e acordar uma data interessante para que ele possa ir até o local do problema. Após o problema ser resolvido, a criadora da tarefa pode finalizar a tarefa, utilizando ainda do botão de troca de situações, porém por estar em um estágio mais avançado da tarefa, as opções exibidas

são Finalizar (*Done*), Voltar ao estado de Candidaturas Finalizadas (*Applications Closed*) e Cancelar (Figura 38).

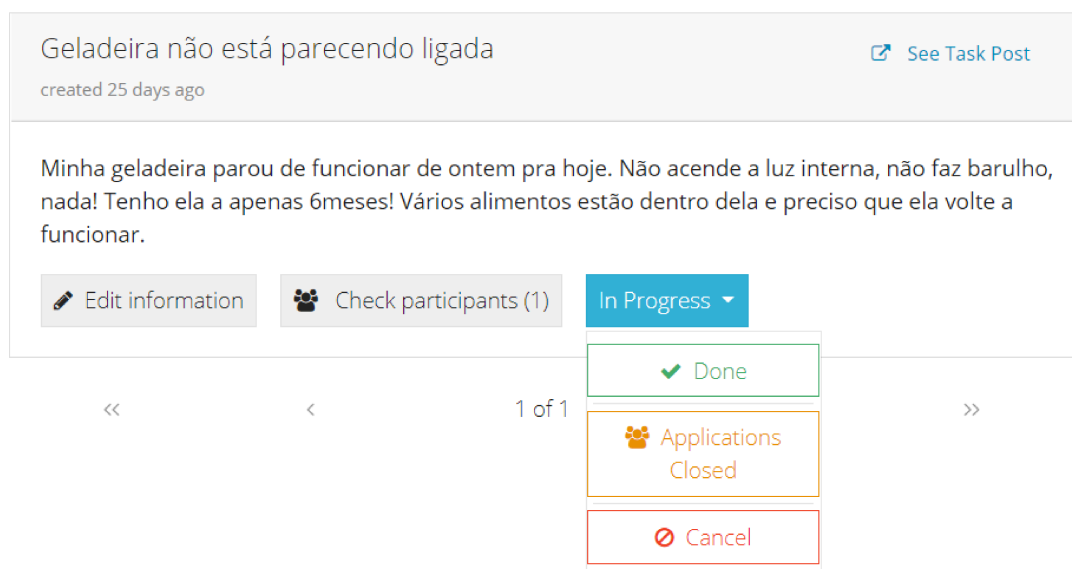


Figura 38 – Opções de mudança de estado quando a tarefa está em estado de execução

A finalização da tarefa irá resultar em um diálogo (Figura 39) contendo todos os participantes da tarefa (que no nosso caso temos apenas um), para que a criadora da tarefa possa definir uma classificação para o serviço daquele prestador. Essa informação de classificação será levada em conta quando o prestador se candidatar para uma próxima tarefa, pois quanto maior for sua classificação, maior será a confiança depositada nele.

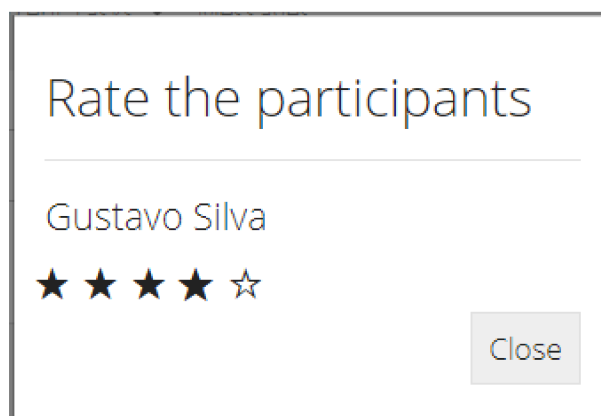


Figura 39 – Diálogo de classificação de participantes da tarefa

3.3.4 Fluxo de criação e completude de tarefas periódicas

O fluxo de criação de tarefas periódicas não se difere muito das pontuais, o acesso a tela é o mesmo, podendo ser acessado através da opção *Task Someone* na tela principal (Figura 14), ou utilizando a barra de navegação, como mostra a Figura 18.

Para essa próxima tarefa, vamos definir que a usuária do sistema precisa revitalizar e manter o jardim de sua casa sempre em ordem, e isso não é uma tarefa de ocorrência única, ou seja, é preciso que mensalmente haja uma manutenção no jardim para garantir que não aparecerão pragas e que as plantas não secarão. Por isso a usuária do sistema irá criar uma nova tarefa **periódica** com os devidos detalhes (e imagens) e marcará a caixa de texto *Repeat* para que possa fornecer a periodicidade com que o jardim deve ser revisitado (que para esse exemplo será mensalmente), a data de início e data de término das visitas (pois a partir de certa data, a usuária tomará conta do jardim ela mesma), como demonstram as Figuras 40 e 41.

Task Someone

Title

Preciso de pessoas para revitalizar e manter meu jardim

Repeat ☒

Frequency

Monthly

Start date

26/12/2020

End date

27/03/2021

Description

Preciso de pessoas dispostas a:
Irrigue as suas plantas regularmente;
Corte a grama regularmente;

254/1000

Address

Rua Francisco Sales, 155, Uberlândia, MG

Attachments





Figura 40 – Informações da tarefa periódica sendo criada

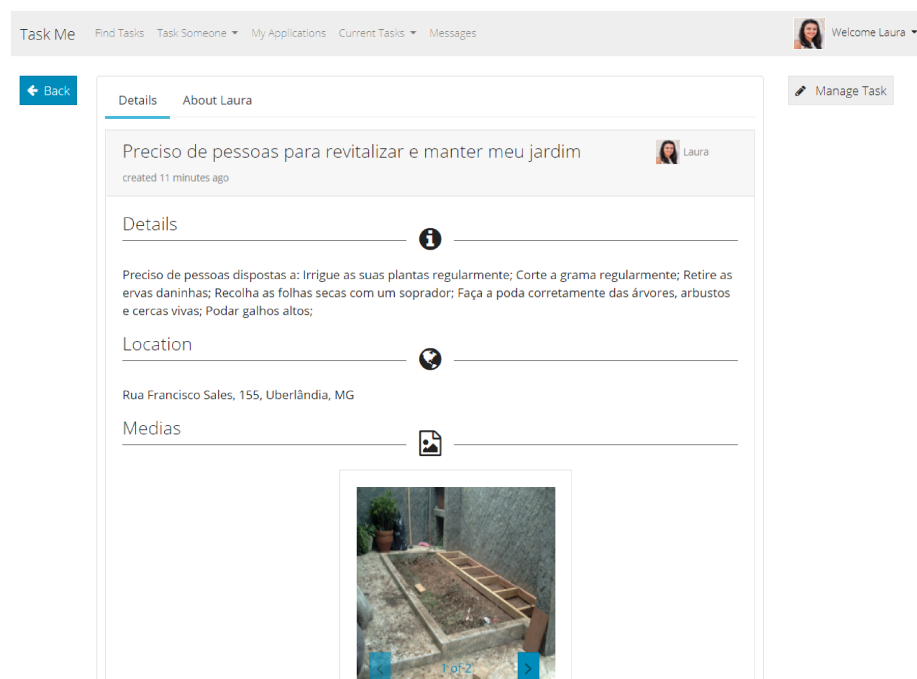


Figura 41 – Informações na tela de detalhes da tarefa

Como dito anteriormente, após a criação da tarefa, esta já se encontra disponível para que os usuários do sistema possam se candidatar e então é possível que a criadora da tarefa possa, na tela Gerenciar Tarefas Criadas (*Manage Created Tasks*), verificar os candidatos à tarefa (Figura 42) .

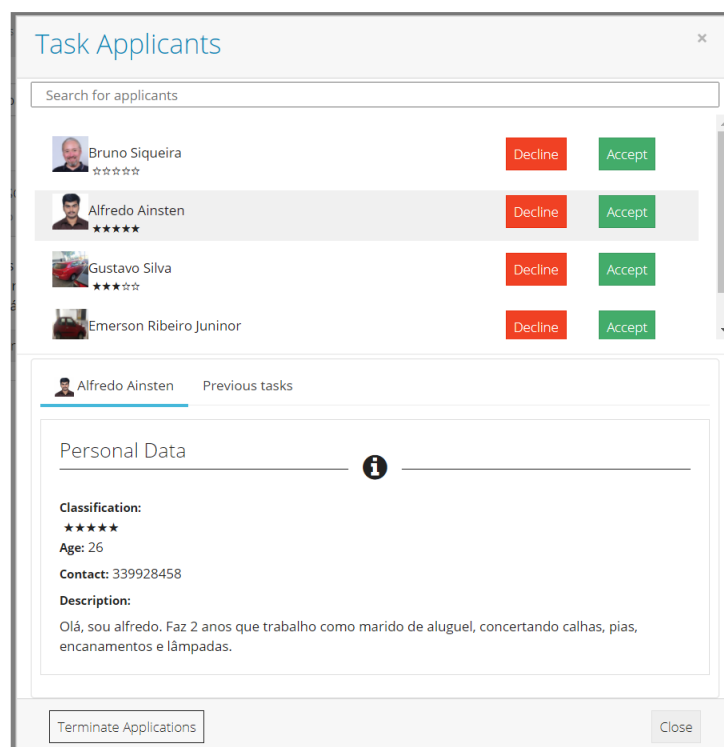


Figura 42 – Candidatos à tarefa

Com base nas descrições de perfil e nas classificações dos usuários, vamos considerar no nosso cenário hipotético que a usuária optou pelos candidatos Alfredo e Gustavo (Figura 43). Diferentemente do que ocorreu na tarefa pontual apresentada na Seção 3.3.3, neste novo cenário dois profissionais foram escolhidos. A quantidade de profissionais escolhidos para determinada tarefa é de livre escolha do usuário e pode levar em consideração critérios como a intensidade do trabalho e a rapidez de sua conclusão, como por exemplo a limpeza de um escritório de três andares, que necessitaria de diversos participantes. De acordo com o serviço solicitado pela usuária, ela imagina que o trabalho será mais eficiente se dividido por dois participantes.

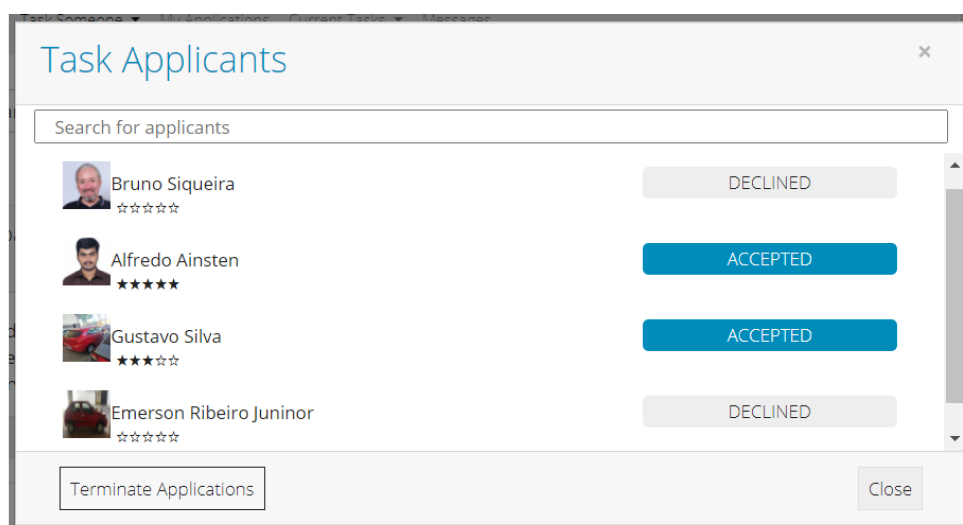


Figura 43 – Conclusão da escolha de candidatos

Após a escolha dos candidatos, podemos alterar a situação da tarefa para Candidaturas Fechadas (*Applications Closed*). Isso pode ser feito pelo botão "*Terminate Applications*", mostrado na Figura 43, e é importante para que a tarefa não seja mais exibida na tela de Encontre uma Tarefa (*Find Task*), e que não haja mais forma de se candidatar para executá-la.

Até o momento, não foram apresentadas grandes mudanças no fluxo da criação e da filtragem por candidatos em comparação ao fluxo de tarefas pontuais. Para o passo seguinte, o início da tarefa, será feito da mesma forma explicada anteriormente, bastando clicar no botão Começar (*Start*) exibido na Figura 35. Mas a mensagem de confirmação não será a mesma, pois para tarefas periódicas, além de avisar que uma mensagem será enviada a todos os participantes da tarefa alertando o início da tarefas, também será avisado que sub-tarefas serão criadas ao iniciar essa tarefa, de acordo com a frequência especificada para a tarefa (Figura 44). Essas sub-tarefas serão explicadas posteriormente.

Após iniciar a tarefa, diferentemente dos botões que podem ser observados na Figura 38, para tarefas periódicas, um botão a mais é exibido na tela de Gerenciar Tarefas Criadas (*Manage Created Tasks*). Trata-se do botão *Sub Tasks* (Figura 45), utilizado para

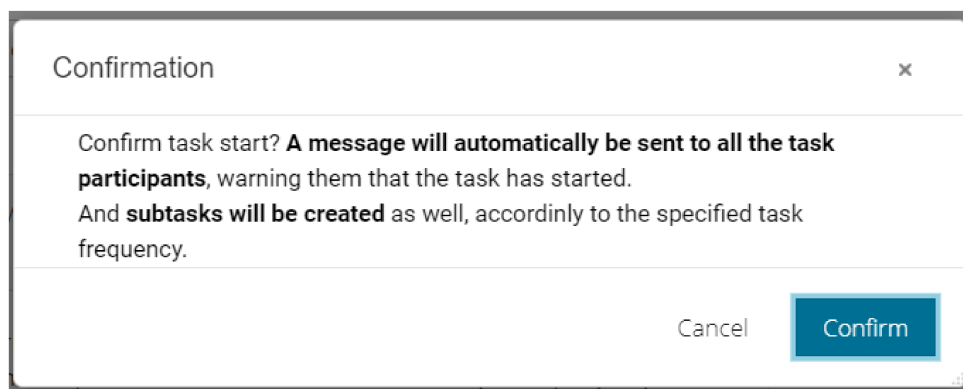


Figura 44 – Confirmação de início de tarefa periódica

que o usuário possa verificar todas as sub-tarefas criadas a partir daquela tarefa periódica iniciada previamente.

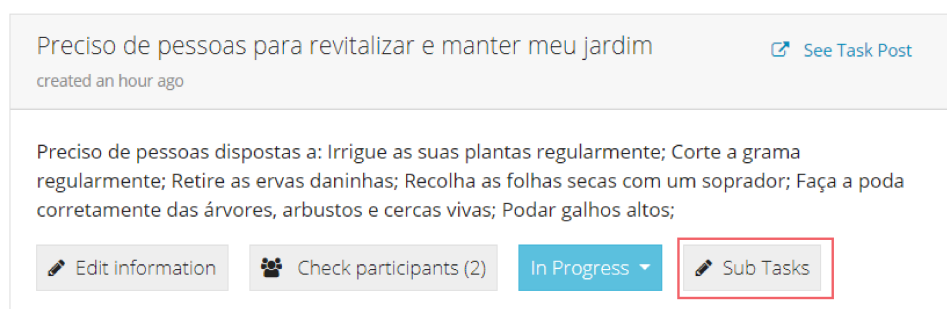


Figura 45 – Botão para exibição das sub-tarefas

Como pode ser observado na Figura 43, a tarefa foi criada com uma frequência mensal, iniciando na data 26/12/2020 e com término em 27/03/2021. Ou seja, automaticamente serão criadas sub-tarefas mensais similares, alterando apenas a data limite de acordo com as datas calculadas. No exemplo em questão a primeira sub-tarefa criada terá data limite 26/12/2020 e a próxima 25/01/2021 (devido a frequência ser mensal) e assim por diante. Para checar as sub-tarefas criadas e seus detalhes, basta que o usuário clique no botão apresentado na Figura 45 para apresentar o diálogo com as sub-tarefas (Figura 46).

Na imagem acima é possível observar que o diálogo das sub-tarefas tem um filtro de datas para que o usuário possa filtrar todas as sub-tarefas de um determinado período. Além disso, cada linha representa uma sub-tarefa onde o primeiro campo é o título da sub-tarefa, o segundo é a data limite, o terceiro campo é a situação daquela sub-tarefa e o quarto são dois botões de ação para visualizar e editar detalhes da sub-tarefa.

As situações das sub-tarefas são reduzidas, pois não existem estados de candidatura aberta ou fechada. Sub-tarefas são criadas na situação Criada (*Created*) com próximo estado sendo Iniciada (*Started*) e então Concluída (*Done*), e em todos esses estados a

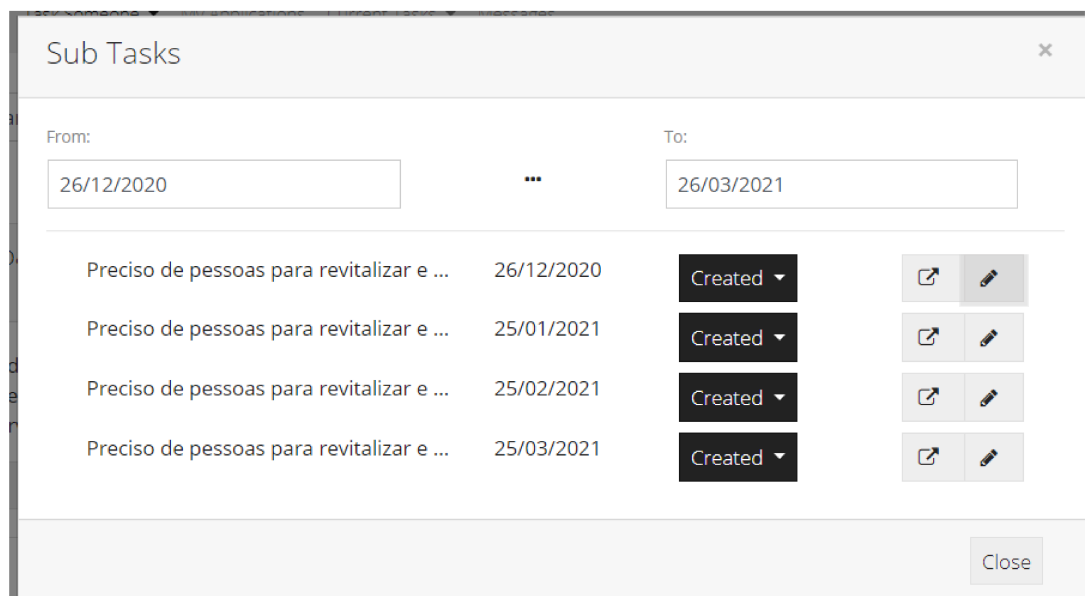


Figura 46 – Diálogo de sub-tarefas

sub-tarefa pode ter sua situação Cancelada (*Cancelled*).

Como dito anteriormente, o título das sub-tarefas é o mesmo da tarefa de origem, bem como a descrição da tarefa e endereço. Mas nada impede que o usuário utilize do botão editar para que possa fazer alterações em uma sub-tarefa em específico. Ao clicar no botão editar, vemos o diálogo de edição das informações (Figura 47).

Digamos que a criadora da tarefa decidiu alterar o título e a descrição da primeira sub-tarefa, para demonstrar mais claramente que a primeira sub-tarefa será focada na revitalização do jardim. E após isso a criadora da tarefa pode mudar a situação de todas as sub-tarefas para Iniciadas (*Started*), ou somente da sub-tarefa em questão (Figura 48). Iniciar uma sub-tarefa não gera envio de mensagens aos participantes, como acontece com as tarefas. Mas alterar o estado das sub-tarefas é importante pois os participantes da tarefa podem visualizar seu status na tela de agenda, que será explicada posteriormente.

Na ótica dos participantes da tarefa, é possível visualizar as sub-tarefas na tela de Agenda de Tarefas (*Task Agenda*), que pode ser acessada através da opção *Task Agenda* na tela principal (Figura 14), ou utilizando a barra de navegação *Current Tasks » Task Agenda*, como mostra a Figura 49. Por meio da utilização da agenda, um participante pode controlar todos seus afazeres, evitando sobrecarregar um dia com diversas sub-tarefas, o que poderia causar a não completude de alguma delas.

Na tela mostrada na Figura 50, vemos que a tela de agenda contém um filtro por período onde a diferença entre as duas datas é a quantidade de dias que os botões de navegação irão crescer ao período atual. Como ilustrado na Figura 51, entre as datas 26/12/2020 e 24/02/2021 existem 60 dias de diferença, logo os botões de navegação ao serem clicados irão navegar 60 dias para o futuro ou para o passado, de acordo com o

Edit Task

Title

Preciso de pessoas para revitalizar e manter meu jardim

State

Created

Due date

26/12/2020

Description

Preciso de pessoas dispostas a:
Irrigue as suas plantas regularmente;
Corte a grama regularmente;

254/1000

Address

Rua Francisco Sales, 155, Uberlândia, MG

Attachments

Add attachments

 Click here or drop your files

Save

Cancel

Figura 47 – Diálogo de edição de sub-tarefas

botão clicado.

A visualização na tela de agenda é feita através do agrupamento de tarefas por data, como mostrado na Figura 50, por exemplo. Para o usuário Alfredo, existem dois afazeres no dia 26/12/2020: “Manutenção de calha” e “Revitalização de jardim”, e graças à tela de agenda o usuário sabe que neste dia suas atribuições já são várias. Dessa forma, ele pode gerenciar as futuras tarefas que irá fazer e impedir que sejam feitas novas atribuições na mesma data, evitando assim o não cumprimento de alguma das tarefas do dia por falta de tempo.

Visto que a situação da sub-tarefa foi alterada para Em Andamento (*In Progress*), subentende-se que as informações dela estão atualizadas de forma que Alfredo não terá dúvidas ao realizar a tarefa. Os participantes da tarefa podem verificar os detalhes clicando nas sub-tarefas (Figura 52).

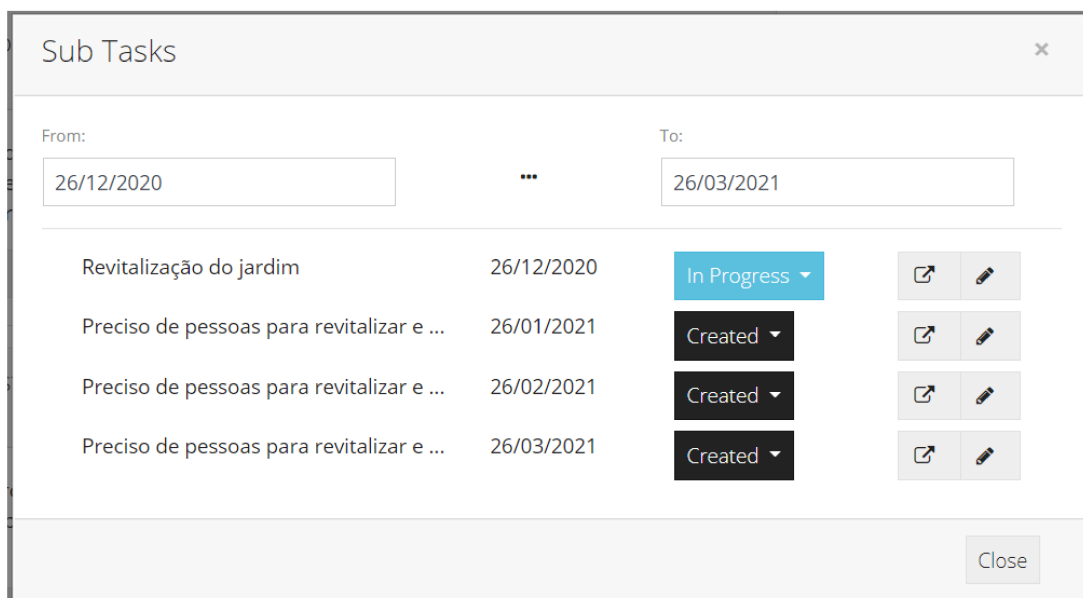


Figura 48 – Sub-tarefas pós edição de título e estado

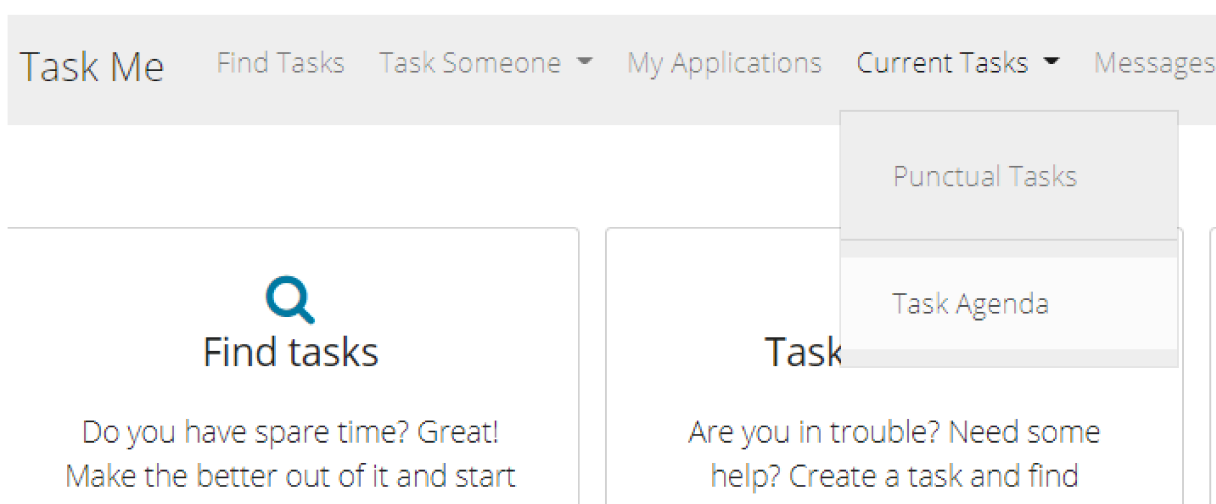


Figura 49 – Opção para acesso à tela de agenda

Durante a execução da tarefa, caso algum participante precise deixar algum tipo de recado para os outros participantes ou para a criadora da tarefa, ou ainda atualizá-la que a tarefa foi finalizada, o participante pode deixar comentários na seção inferior dos detalhes da tarefa (Figura 53).

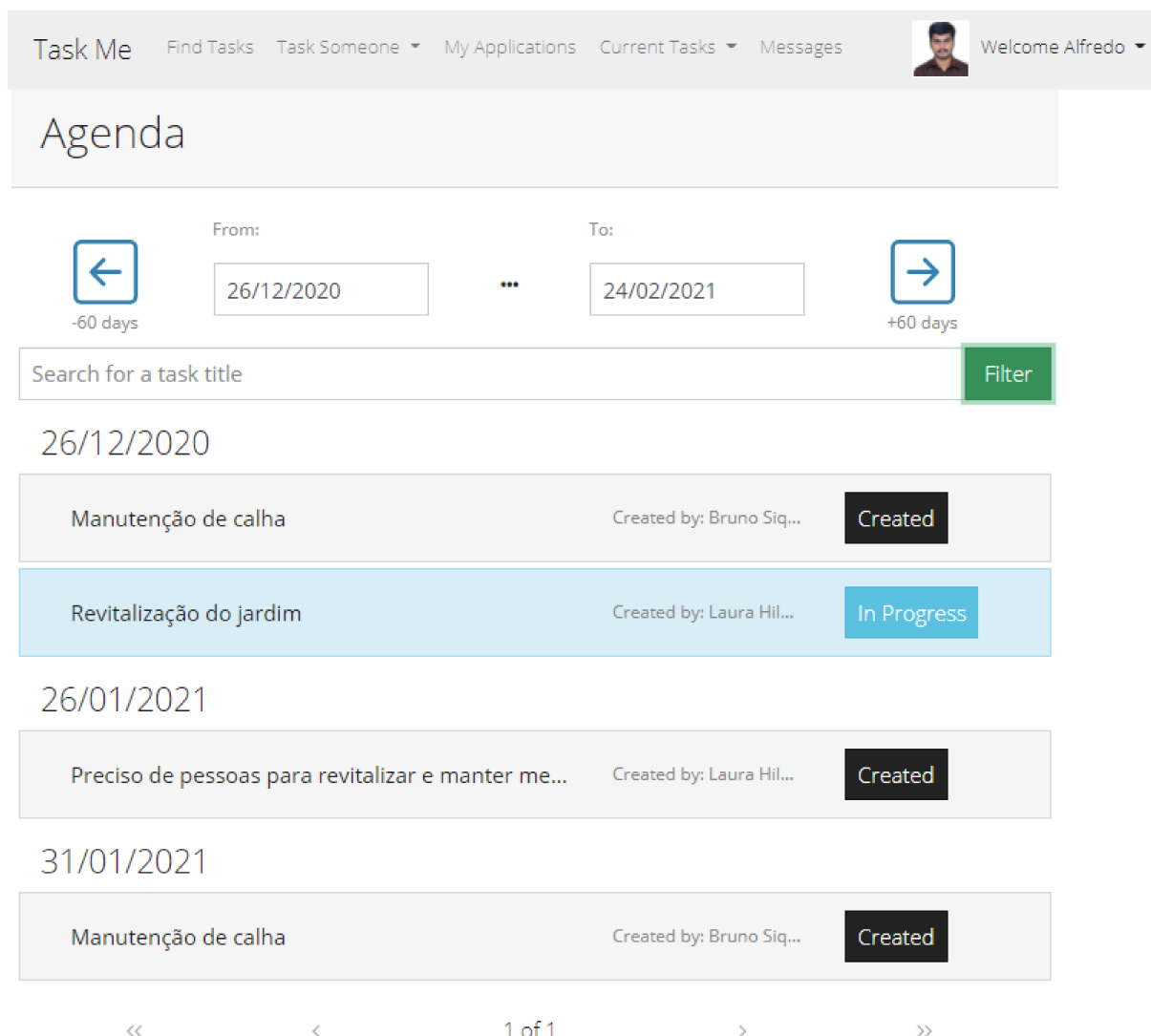


Figura 50 – Tela de agenda

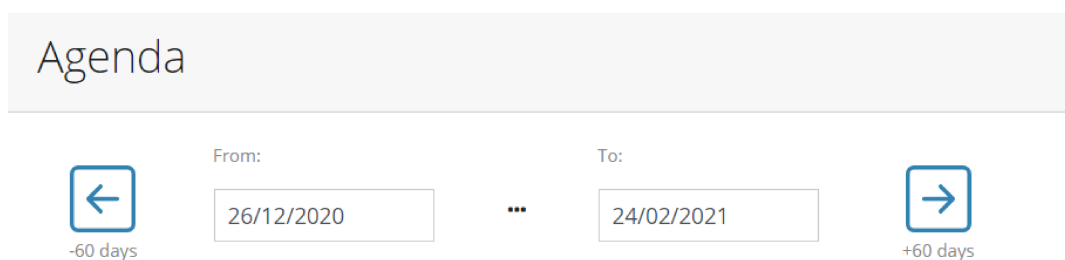


Figura 51 – Navegação e filtro de datas na tela de agenda

Ao verificar o comentário na sub-tarefa e que o serviço do dia foi feito, a criadora pode mudar a situação da sub-tarefa para Encerrada (*Done*), e se necessário fazer algum novo comentário para que os participantes possam vê-lo (Figuras 54 e 55).

No momento atual, caso a criadora da tarefa principal opte por finalizá-la, não será possível, já que apenas uma sub-tarefa está como concluída, como mostra a Figura 55.

The screenshot displays a modal window for a sub-task. At the top, the title 'Revitalização do jardim' is on the left and the date '26/12/2020' is on the right. Below this, there are three fields: 'Creator:' with a profile picture of Laura, 'Location:' with the address 'Rua Francisco Sales, 155, Uberlândia, MG', and 'State:' with a blue button labeled 'In Progress'. A 'Description:' section follows, containing the text: 'Primeiramente vamos: Irrigar as suas plantas, cortar a grama e retirar as ervas daninhas;'. Below the description is a 'Comments:' section with a message 'No comments were added to this task yet.' and 'No results found'. There is a text input field with the placeholder 'Add some comment here...' and a blue 'Add comment' button. A 'Close' button is located at the bottom right.

Figura 52 – Detalhes da sub-tarefa na tela de agenda

This screenshot shows the 'Comments:' section of the sub-task details. It lists three comments: 'Alfredo - 5 days ago' with the text 'Preciso que alguém leve adubo pra brotos!'; 'Gustavo - 4 days ago' with 'Deixa comigo! Tenho 2kg aqui!'; and 'Alfredo - 2 days ago' with 'Tarefa do dia finalizada! Pode encerrar esta, Laura.'. Below the comments is a text input field with the placeholder 'Add some comment here...' and a blue 'Add comment' button. A 'Close' button is at the bottom right. Navigation arrows and '1 of 1' are visible between the comments and the input field.

Figura 53 – Comentários feitos na sub-tarefa

Para a finalização da tarefa principal, suas sub-tarefas devem estar todas concluídas (ou canceladas), caso contrário o usuário receberá uma mensagem de erro alertando-o sobre

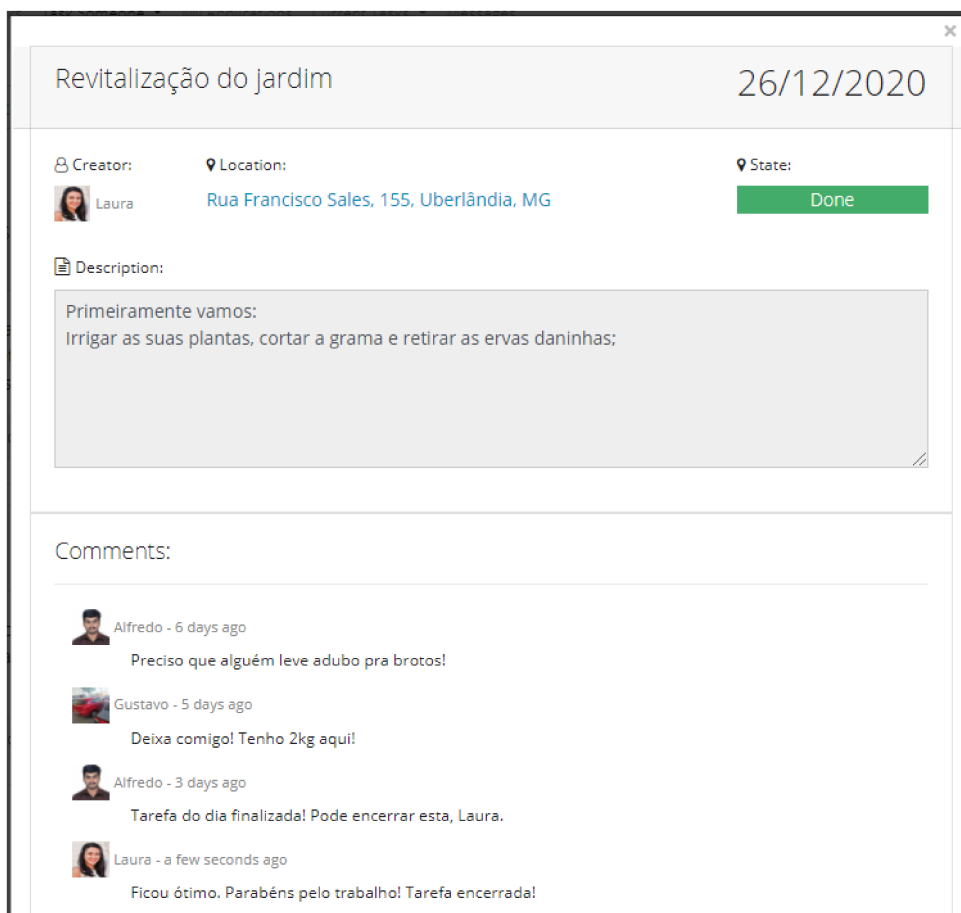


Figura 54 – Tarefa finalizada, com comentário da criadora

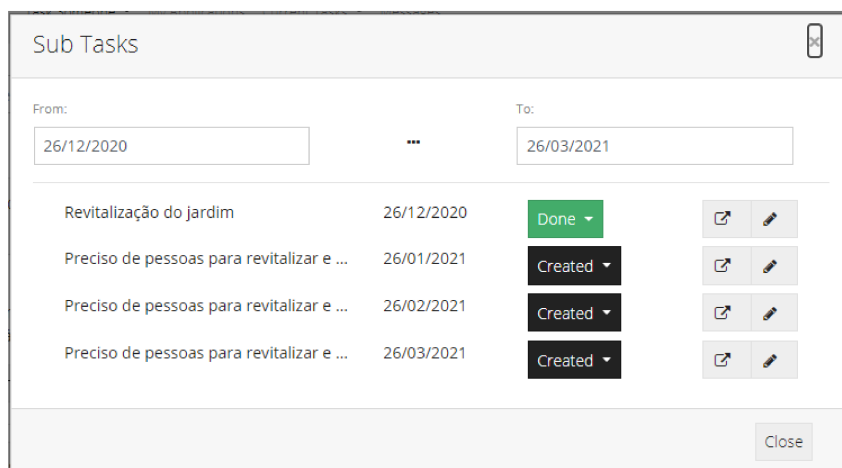


Figura 55 – Diálogo de sub-tarefas contendo somente uma finalizada

isso, como mostrada na Figura 56.

Após alguns meses e a finalização de todas as sub-tarefas (Figura 57), a criadora poderá então finalizar a tarefa principal através da tela de *Manage Created Tasks*, através da opção de mudança de situação (Figura 58), que é o mesmo processo de conclusão feito para tarefas pontuais.

There is unfinished subtasks! In order to conclude this task, all the subtasks must be DONE(or CANCELLED).

Figura 56 – Erro após tentativa de finalização de tarefa periódica sem ter finalizado suas sub-tarefas

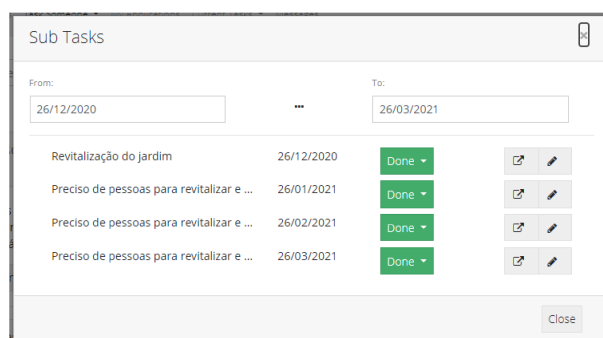


Figura 57 – Todas as sub-tarefas finalizadas



Figura 58 – Mudança de estado da tarefa para finalizada (*Done*)

Na ótica dos participantes, as tarefas concluídas serão exibidas na tela de agenda como mostra a Figura 59. Como a funcionalidade de comentários ainda se encontrará disponível, qualquer participante que deseje fazer alguma consideração, poderá fazê-lo mesmo após a finalização da tarefa principal.

Da mesma forma que ocorre no encerramento de tarefas pontuais, a classificação de participantes também é uma etapa na conclusão de uma tarefa periódica (Figura 60). Dessa maneira os participantes serão mais bem vistos em suas futuras candidaturas. Por fim, nada impede que um prestador se candidate para novos serviços oferecidos por pessoas para as quais ele já trabalhou anteriormente. De igual forma, um prestador de

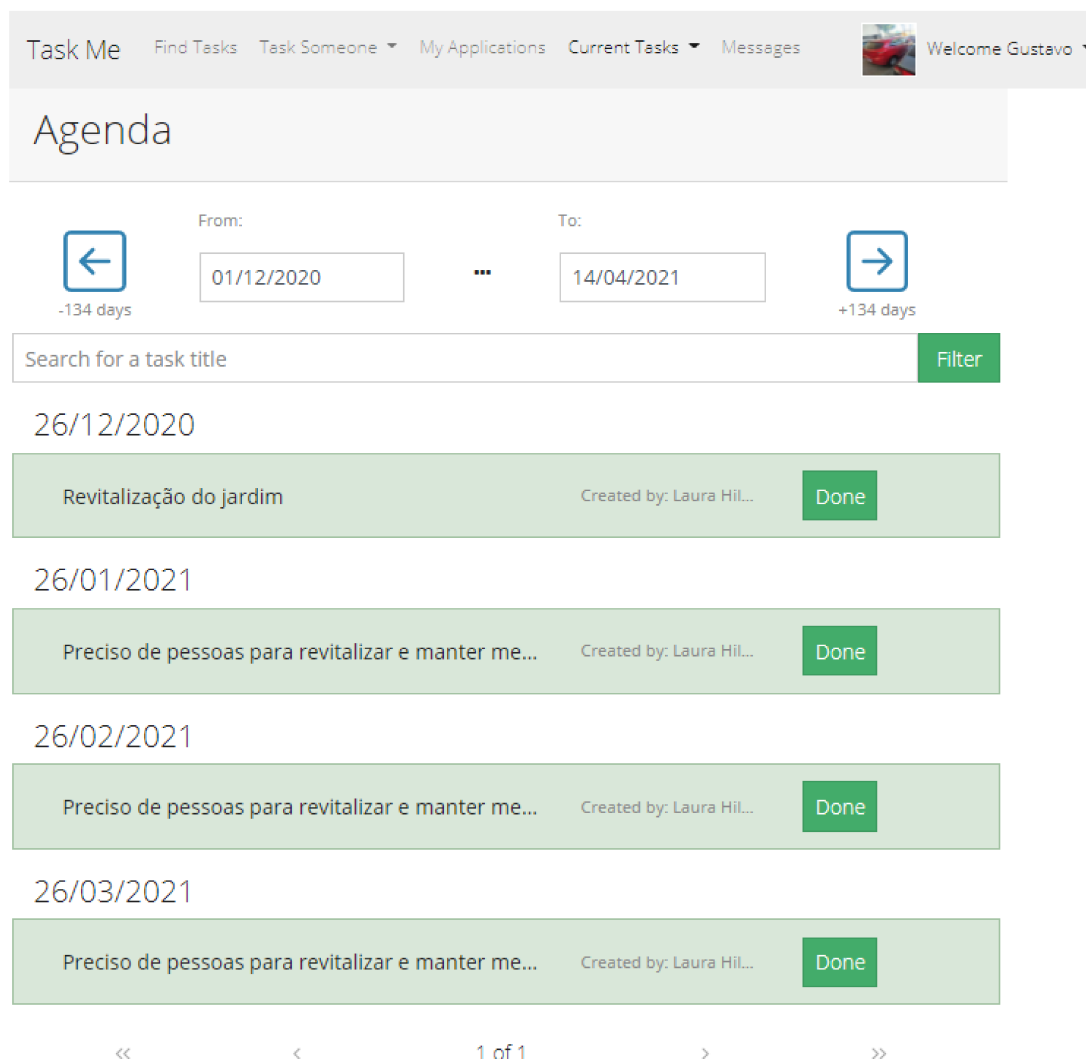


Figura 59 – Sub-tarefas concluídas exibidas na tela de Agenda (*Task Agenda*)

determinado serviço pode atuar como contratante de outro e vice-versa. Essa dinâmica, percebida rotineiramente em nossa sociedade, é prevista e esperada na ferramenta.

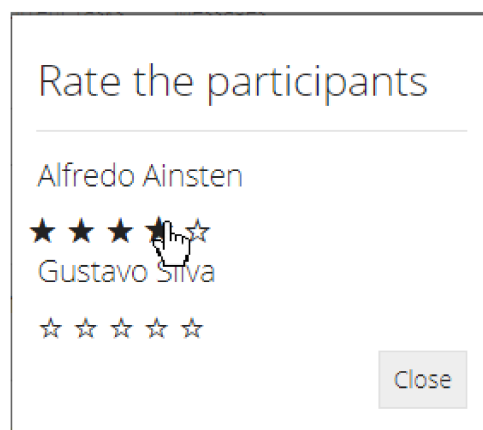


Figura 60 – Classificação dos participantes da tarefa

4 Conclusão

Com todos os avanços tecnológicos, diariamente a sociedade vem informatizando cada vez mais atividades cotidianas, e o processo de oferta de serviços não poderia ser diferente. Diversas aplicações foram criadas com o objetivo de tentar reduzir a necessidade do usuário de ter que fazer orçamento em diversos prestadores, realizando inúmeras ligações e se locomovendo a vários deles.

Centralizar os prestadores em uma plataforma é uma estratégia inteligente utilizada pelas plataformas de prestação de serviço, dessa forma o usuário diz qual sua necessidade e então a aplicação envia aos interessados para que eles façam orçamentos e entrem em contato por e-mail. A confiança que o usuário tem nesse prestador pode não ser a ideal, visto que ele apenas recebeu um retorno com o preço do serviço, então naturalmente o usuário tende a ligar para o prestador para adquirir mais detalhes, e então obter a confiança que o permita entrar em sua casa para realizar algum serviço, por exemplo. Porém, dessa forma o usuário precisa de meios externos à plataforma (por exemplo, ligações telefônicas), e o intuito deveria ser passar o máximo de confiança possível utilizando apenas a plataforma.

A organização das atribuições direcionadas ao prestador de serviços é um ponto que deixa a desejar nas aplicações de mercado. A falta desta faz com que a responsabilidade de saber sua agenda passe a ser completamente do prestador de serviço, deixando de ser uma funcionalidade do sistema e aumentando o risco de erro humano, como por exemplo incumbir-se de diversos afazeres em um mesmo dia e conseqüentemente se esquecer de algum deles.

O desenvolvimento do TaskMe foi realizado com a mesma motivação das aplicações de mercado, porém com foco principal no problema do usuário, na confiança entre usuários e prestadores de serviços, e na organização de afazeres dos prestadores. Na aplicação é possível materializar o problema do usuário em forma de uma tarefa, seja ela pontual ou periódica.

Contando com um chat integrado na aplicação, o diálogo entre um interessado na tarefa e o criador desta se torna muito mais fácil e esclarecedor, não havendo a obrigação de ligar para cada um dos interessados em realizar a tarefa, já que é possível dialogar utilizando o TaskMe. Essa comunicação interna aliada à funcionalidade de classificação de prestadores de serviço (com base em serviços prestados anteriormente) reforça ainda mais a confiança do usuário nos prestadores.

Para os usuários prestadores de serviços o TaskMe conta com organizações de todos os tipos de tarefas. Na tela *Punctual Tasks* é possível que ele veja todas as tarefas

pontuais e quantos dias até a data limite de cada uma delas. Para tarefas periódicas, devido ao fluxo ser maior, a tela *Task Agenda* tem uma organização em forma de agenda para que as atribuições e suas datas sejam exibidas da forma mais clara possível. Ambas essas telas foram feitas com o objetivo de assumir a responsabilidade de organizar as tarefas do usuário em busca de reduzir esquecimentos e sobrecarga de tarefas.

Visando evitar problemas relacionados a hardware obsoleto ou danificado, toda a aplicação foi hospedada na nuvem, tanto a interface quanto o servidor, através do Heroku (uma plataforma na nuvem que oferece serviços de hospedagem). Dessa forma a aplicação se encontra disponível apenas para um ambiente de testes, pois o plano de hospedagem utilizado na plataforma é o gratuito, logo a aplicação não fica *online* 24 horas por dia.

4.1 Trabalhos Futuros

Como próximas etapas deste trabalho para aprimorar/estender a solução proposta, compreendem-se:

- Adicionar novas funcionalidades, como por exemplo:
 - Adicionar interesses no perfil do usuário, para que a aplicação possa mostrar somente tarefas relacionadas com seus interesses.
 - Criar filtros na tela *Find Tasks*, caso o usuário queira filtrar por data, ou outros campos pertinentes.
 - Implementar funcionalidade de envio de notificações por e-mail/SMS aos aplicantes de uma tarefa quando são aprovados e para participantes da tarefa quando a tarefa for iniciada.
- Disponibilizar a aplicação para uso, através de um *upgrade* do pacote de hospedagem na plataforma Heroku, tornando a aplicação funcional 24 horas por dia.
- Desenvolver um aplicativo para *smartphones* com o objetivo de melhorar a experiência dos usuários ao utilizarem o TaskMe através de dispositivos móveis, recebendo notificações diretamente em seus dispositivos ao invés de receberem e-mails.
- Realizar estudo de usabilidade da aplicação.
- Conceber um plano de negócio para ponderar a possibilidade de transformar a aplicação desenvolvida em um produto rentável.

Referências

- ABRAHAMSSON, P. et al. Agile software development methods: Review and analysis. *arXiv preprint arXiv:1709.08439*, p. 22, 2017. Citado na página 21.
- ALMEIDA, R. *HTML: Linguagem de Marcação de Hipertexto*. 2012. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTML/HTML5>>. Acesso em: 12 sep. 2020. Citado na página 23.
- BASTOS, D. F. *O que é Model-view-controller (MVC)?* 2011. Disponível em: <https://www.oficinadanet.com.br/artigo/desenvolvimento/o_que_e_model-view-controller_mvc>. Acesso em: 12 sep. 2020. Citado na página 22.
- Berczuk, S. Back to basics: The role of agile principles in success with an distributed scrum team. In: *Agile 2007 (AGILE 2007)*. [S.l.: s.n.], 2007. p. 382–388. Citado na página 20.
- BERNARDO, K. *Scrum: O guia completo e definitivo!* 2019. Disponível em: <<https://www.culturaagil.com.br/scrum-o-guia-completo-e-definitivo/>>. Acesso em: 12 sep. 2020. Citado na página 19.
- CARVALHO, B. V. d.; MELLO, C. H. P. Aplicação do método ágil scrum no desenvolvimento de produtos de software em uma pequena empresa de base tecnológica. *Gestão e Produção*, sciELO, v. 19, p. 557 – 573, 00 2012. ISSN 0104-530X. Disponível em: <http://www.scielo.br/scielo.php?script=sci_arttext&pid=S0104-530X2012000300009&nrm=iso>. Citado 3 vezes nas páginas 20, 21 e 22.
- DEMETRIO, A. *Site de busca de profissionais ganha prêmio de melhor startup brasileira*. 2012. Disponível em: <<http://g1.globo.com/tecnologia/noticia/2012/07/site-de-busca-de-profissionais-ganha-premio-de-melhor-startup-brasileira.html>>. Acesso em: 22 sep. 2020. Citado na página 15.
- DUARTE, L. *O que é Scrum?* 2020. Disponível em: <<https://escritoriodeprojetos.com.br/o-que-e-scrum>>. Acesso em: 01 oct. 2020. Citado na página 19.
- EDWARDS, M. D. Overhauling a failed project using out of the box scrum. In: *IEEE. Agile 2008 Conference*. [S.l.], 2008. p. 413–416. Citado na página 20.
- ESPÍNDOLA, E. C. *A importância do Modelagem de Objetos no Desenvolvimento de Sistemas*. 2007. Disponível em: <<http://linhadecodigo.com.br/artigo/1293/a-importancia-do-modelagem-de-objetos-no-desenvolvimento-de-sistemas.aspx>>. Acesso em: 08 jan. 2020. Citado na página 29.
- ESTÁGIOS, C. de. *Redes sociais estão consolidadas na busca por vagas: LinkedIn é a mais usada, aponta pesquisa*. 2019. Wiki do abnTeX2. Disponível em: <<https://www.ciadeestagios.com.br/redes-sociais-estao-consolidadas-na-busca-por-vagas-linkedin-e-a-mais-usada-aponta-pesquisa/>>. Acesso em: 03 sep. 2020. Citado na página 12.

GAVIGAN, D. *The History of Angular*. 2018. Disponível em: <<https://medium.com/the-startup-lab-blog/the-history-of-angular-3e36f7e828c7>>. Acesso em: 19 out. 2020. Citado na página 25.

GLOBO, O. *Recebeu SMS do banco? Cuidado, pode ser golpe*. 2017. Wiki do abnTeX2. Disponível em: <<https://oglobo.globo.com/economia/recebeu-sms-do-banco-cuidado-pode-ser-golpe-20904905>>. Acesso em: 01 sep. 2020. Citado na página 12.

IBGE, P. C. do. *Desemprego sobe para 13,3% no 2º trimestre, com redução recorde de ocupados*. 2020. Wiki do abnTeX2. Disponível em: <<https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/28480-desemprego-sobe-para-13-3-no-2-trimestre-com-reducao-recorde-de-ocupados>>. Acesso em: 03 sep. 2020. Citado na página 12.

Kniberg, H.; Farhang, R. Bootstrapping scrum and xp under crisis a story from the trenches. In: *Agile 2008 Conference*. [S.l.: s.n.], 2008. p. 436–444. Citado na página 20.

KRILL, P. *GitHub's top 10 most popular projects*. 2013. Disponível em: <<https://www.computerworld.com/article/2473298/120001-GitHub-s-top-10-rock-star-projects.html#slide1>>. Acesso em: 19 out. 2020. Citado na página 25.

LIMA, G. *HTML: Linguagem de Marcação de Hipertexto*. 2012. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/CSS>>. Acesso em: 12 sep. 2020. Citado na página 24.

Mann, C.; Maurer, F. A case study on the impact of scrum on overtime and customer satisfaction. In: *Agile Development Conference (ADC'05)*. [S.l.: s.n.], 2005. p. 70–79. Citado na página 20.

Marcal, A. S. C. et al. Mapping cmmi project management process areas to scrum practices. In: *31st IEEE Software Engineering Workshop (SEW 2007)*. [S.l.: s.n.], 2007. p. 13–22. Citado na página 20.

MILLS, C. *JavaScript*. 2016. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Aprender/JavaScript>>. Acesso em: 12 sep. 2020. Citado na página 24.

NOAM, I. *JavaScript*. 2019. Disponível em: <https://developer.mozilla.org/pt-BR/docs/DOM/Referencia_do_DOM/Introdu%C3%A7%C3%A3o>. Acesso em: 12 sep. 2020. Citado na página 24.

OLIVEIRA, L. V. de. *UML – Diagramas de Sequência*. 2013. Disponível em: <<http://www.theclub.com.br/Restrito/Revistas/201308/umld1308.aspx>>. Acesso em: 25 dec. 2020. Citado na página 32.

PERRIER, J.-Y. *HTML: Linguagem de Marcação de Hipertexto*. 2013. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/Web/HTML>>. Acesso em: 12 sep. 2020. Citado na página 23.

PONTES, A. *O que é CSS3 ?* 2015. Disponível em: <<https://andrehost.com.br/o-que-e-css3/>>. Acesso em: 12 sep. 2020. Citado na página 24.

- RIBEIRO, A. *Burndown e Burnup no Scrum: como avaliar o desempenho da sua equipe*. 2019. Disponível em: <<https://flowup.me/blog/burndown-e-burnup/>>. Acesso em: 22 sep. 2020. Citado na página 22.
- RISING, L.; JANOFF, N. S. The scrum software development process for small teams. *IEEE software*, IEEE, v. 17, n. 4, p. 26–32, 2000. Citado na página 20.
- ROCHA, B. M. *Desemprego sobe para 13,3% no 2º trimestre, com redução recorde de ocupados*. 2018. Disponível em: <<http://intra.serpro.gov.br/tema/artigos-opinioes/agilizando-o-desenvolvimento-com-spring-framework>>. Acesso em: 01 oct. 2020. Citado na página 26.
- RODRIGUES, J. *Modelo Entidade Relacionamento (MER) e Diagrama Entidade-Relacionamento (DER)*. 2014. Disponível em: <<https://www.devmedia.com.br/modelo-entidade-relacionamento-mer-e-diagrama-entidade-relacionamento-der/14332>>. Acesso em: 25 dec. 2020. Citado na página 30.
- SANDERS, D. Using scrum to manage student projects. *Journal of Computing Sciences in Colleges*, Consortium for Computing Sciences in Colleges, v. 23, n. 1, p. 79–79, 2007. Citado na página 20.
- SCHWABER, K.; BEEDLE, M. *Agile software development with Scrum*. [S.l.]: Prentice Hall Upper Saddle River, 2002. v. 1. Citado na página 20.
- SOCIAL, S. *Social media demographics to inform your brand's strategy in 2020*. 2020. Wiki do abnTeX2. Disponível em: <<https://sproutsocial.com/insights/new-social-media-demographics/>>. Acesso em: 03 sep. 2020. Citado na página 12.
- Sulaiman, T.; Barton, B.; Blackburn, T. Agileevm - earned value management in scrum projects. In: *AGILE 2006 (AGILE'06)*. [S.l.: s.n.], 2006. p. 10 pp.–16. Citado na página 20.
- Sutherland, J. et al. Fully distributed scrum: The secret sauce for hyperproductive offshored development teams. In: *Agile 2008 Conference*. [S.l.: s.n.], 2008. p. 339–344. Citado na página 20.
- Sutherland, J. et al. Distributed scrum: Agile project management with outsourced development teams. In: *2007 40th Annual Hawaii International Conference on System Sciences (HICSS'07)*. [S.l.: s.n.], 2007. p. 274a–274a. Citado na página 20.
- TAKEUCHI, I. N. H. *The New New Product Development Game*. 1986. Disponível em: <<https://hbr.org/1986/01/the-new-new-product-development-game>>. Acesso em: 12 sep. 2020. Citado na página 19.