RECA
REVISTA ELETRÔNICA DE COMPUTAÇÃO APLICADA
VOL. 1 NRO. 2 2020

APLICATIVO PONTO DOMÉSTICA

Luccas Passeto
Graduando em Engenharia de Software – Uni-FACEF
luccaspasseto@gmail.com

Guilherme Bertoldi Maia de Souza
Graduando em Engenharia de Software – Uni-FACEF
guilhermebertoldi21384@gmail.com

Prof. Dr. Daniel Facciolo Pires
Docente do Departamento de Computação – Uni-FACEF
daniel@facef.br

RESUMO

São cada vez mais comuns os empregadores domésticos não terem tempo para ficarem em casa ou também os horários de entrada de empregada doméstica não coincidir com o horário de saída dos empregadores domésticos. Ainda, é também comum não se ter um controle adequado das horas trabalhadas dos seus colaboradores, com isso diversos problemas são gerados, e um deles é a dificuldade de comunicação entre empregador e colaborador quando o assunto é a quantidade de horas trabalhadas no dia e na semana. Este artigo teve como objetivo desenvolver um aplicativo móvel para auxiliar no controle de horas registradas pelo colaborador. Desta forma o usuário pode entrar com seu CPF e com a senha que o empregador doméstico passar e poderá efetuar o ponto de entrada, saída, e ele poderá ver o ponto registrado em sua única tela disponível para empregados domésticos que é uma tabela com o status do ponto batido com o tipo do ponto, nome, dia, hora. Este artigo utiliza-se de técnicas de desenvolvimento de aplicações computacionais móveis juntamente com os processos de documentação e modelagem de software.

Palavras-chave: Aplicação móvel. Gestão de ponto eletrônico.

ABSTRACT

It is more and more common that domestic employers do not have time to stay at home or the domestic maid's entry times do not coincide with the domestic employers' departure times, and also do not have adequate control of the hours worked by their employees, with this causes several problems, and the worst of them, poor communication between employer and employee. This article aimed to develop a mobile application to assist in the control of hours recorded by the employee. In this way the user can enter his CPF and the password that the domestic employer gives and he will be able to make the entry, exit point, and he will be able to see the registered point on his only screen available to domestic workers which is a table with the status of the beaten stitch with the stitch type, name, day, time. This article demonstrates techniques for developing mobile computing applications together with the software modeling and documentation processes.

Keywords: Mobile application. Electronic point management.

1 INTRODUÇÃO

Nos dias atuais, o chefe que contrata um empregado doméstico não consegue ter o total controle sobre seus horários pontuais de trabalho devido aos horários divergentes de entrada e saída, principalmente em grandes municípios. A partir disso, nos desafiamos a desenvolver um aplicativo para controle desses pontos de trabalhos domésticos, sendo assim possível para o empregador consultar os horários de entrada e saída de seus empregados domésticos. Além de proporcionar registros dos pontos de trabalho para eventuais obrigações fiscais.

Levando em consideração todas as análises realizadas, este trabalho tem como objetivo desenvolver um aplicativo para que o empregado doméstico tenha uma maior praticidade para o registro do ponto de entrada e saída, e que seja proporcionada rapidez para o empregador na busca de registros de horários de seus funcionários, e com a ajuda da UX/UI (*User Experience / User Interface*) facilitar para ambos na navegação das telas e funcionalidades com fácil entendimento e usabilidade.

Após uma reunião realizada com um contador, nos foi apresentada a dificuldade em possuir dados íntegros e verídicos referentes ao trabalho doméstico, a partir disso, nos propormos a desenvolver um sistema mobile, capaz de registrar esses pontos dos empregados domésticos, gerando um relatório mensal ao empregador dos registros, garantindo maior integridade e segurança nos pontos de trabalho cadastrados.

Para o desenvolvimento foram utilizadas algumas ferramentas tecnológicas. Para gerência do projeto foi aplicada a metodologia Kanban, para designar, dividir e organizar as tarefas, definindo o trabalho realizado e seguindo o prazo para conclusão. Na documentação foram usadas ferramentas para gerar o BPMN (*Business Process Model and Notation*), Diagrama de caso de uso e EAP (Estrutura analítica de projetos).

Levando em consideração a interface e usabilidade, realizamos a prototipação de telas, cuja ferramenta utilizada foi o FIGMA, capaz de realizar as telas finais do projeto. Para a criação do sistema, utilizados o framework React-Native integrado com uma API desenvolvida em NodeJS, onde contém toda a regra de negócio do software e é responsável por inserir, editar e deletar dados do banco de dados da aplicação.

Este artigo está organizado em 7 seções. A seção 1 Apresenta uma introdução, a seção 2 traça um referencial teórico dos temas relacionados. A seção 3 destaca o Modelo Canvas desenvolvido, enquanto que na seção 4 são ilustradas as telas do aplicativo. A seção 5 discute a implementação, e finalmente a seção 6 tece uma conclusão dos trabalhos.

2 REFERÊNCIAL TEÓRICO

Esta seção irá abordar temas relacionados a engenharia de software, desenvolvimento de software, a importância do registro de pontos para empregados domésticos, a biblioteca de desenvolvimento da interface de usuário React-Native, e o gerenciador de banco de dados MySql.

2.1 ENGENHARIA DE SOFTWARE

A engenharia de software é uma tecnologia em camadas. Como ilustra a Figura 1, qualquer tratamento de engenharia (inclusive engenharia de software) deve estar fundamentado em um comprometimento organizacional com a qualidade. (PRESSMAN, 2016) O termo “engenharia de software” ficou conhecido no ano de 1968 após a crise do software, palavra que foi utilizada na época que ocorriam dificuldades no desenvolvimento de sistemas livres de defeitos e que refletissem confiança para o usuário.

Figura 1 – Camadas da engenharia de software

Fonte: (PRESSMAN, 2016, p. 16)

Os métodos da engenharia de software fornecem as informações técnicas para desenvolver software. Os métodos envolvem uma ampla variedade de tarefas, que incluem: comunicação, análise de requisitos, modelagem de projeto, construção de programas, testes e suporte. Os métodos da engenharia de software se baseiam em um conjunto de princípios básicos que governam cada área da tecnologia e incluem atividades de modelagem e outras técnicas descritivas (PRESSMAN, 2016, p. 16). Com base nos resultados obtidos através dos métodos utilizados, garantem a qualidade de entrega do software.

2.1.1 DESENVOLVIMENTO DE SOFTWARE

Levando em consideração a importância cada vez maior do software nas empresas, devemos nos preocupar com a maneira com que irá agregar valor aos negócio, aumentando a produtividade e diminuindo custos. Desse modo, apresentaremos técnicas para desenvolvimento de sistemas com qualidade e menor custo, tanto no desenvolvimento quanto na manutenção.

Metodologias de desenvolvimento são conjuntos de atividades pensadas para coordenar e auxiliar a criação de software. É preciso definir quem faz o quê, quando, como e onde. Não sendo o bastante definir tudo isso em apenas uma reunião, as etapas e as necessidades mudam, se tornando muito dinâmico. Nesse

contexto são bem vindas às metodologias de desenvolvimento de software, assim auxiliando como o trabalho será conduzido e acompanhado.

Numa pesquisa recente a Fundação instituto de administração (2016), independente da metodologia escolhida, o trabalho envolve as seguintes figuras: desenvolvedores (aqueles que receberão os requisitos e construirão o software) e gerente de projetos (responsável por controlar o andamento do trabalho, fazendo o possível para que as entregas sigam dentro dos prazos planejados), na atualidade, podem separar as metodologias em tradicionais e ágeis. Nas tradicionais, como modelo em cascata, metodologia estruturada, o escopo é pouco flexível, por isso, elas têm dado abertura e lugar a metodologias ágeis, uma exigência dos tempos de transformação digital, onde é preciso ter flexibilidade, etapas menores e feedbacks e alinhamentos diários.

Metodologias ágeis têm sido apontadas como uma alternativa às abordagens tradicionais para o desenvolvimento de software. As metodologias tradicionais, conhecidas também como pesadas ou orientadas a planejamentos, devem ser aplicadas apenas em situações em que os requisitos do sistema são estáveis e requisitos futuros são previsíveis. Entretanto, em projetos em que há muitas mudanças, em que os requisitos são passíveis de alterações, onde refazer partes do código não é uma atividade que apresenta alto custo, as equipes são pequenas, as datas de entrega do software são curtas e o desenvolvimento rápido é fundamental, não pode haver requisitos estáticos, necessitando então de metodologias ágeis. Além disso o ambiente das organizações é dinâmico, não permitindo então que os requisitos sejam estáticos (FERREIRA, 2006).

2.2.1 PRINCIPAIS ETAPAS PARA O DESENVOLVIMENTO DE SOFTWARES

Dependendo da metodologia escolhida, pode haver uma diferença na divisão das etapas, porém todas as fases abaixo são essenciais para iniciar e finalizar um projeto.

De acordo com um artigo de qualidade de software publicado pelo DevMedia (2007), o primeiro passo para qualquer projeto de software é começar com o levantamento de requisitos, esse levantamento tem o objetivo de saber quais são as reais necessidades dos clientes. Requisitos são condições, ou seja, exigências que determinam que o software precise seguir determinados padrões.

Após levantar os requisitos é necessário realizar a análise dos requisitos, que consiste em avaliar e verificar a viabilidade do trabalho, então para seguir os próximos passos, é preciso que a equipe seja capaz de desenvolver o software considerando todos os requisitos levantados. Pode acontecer também de os requisitos se tornarem inviáveis seja por ser tecnicamente irrealizável ou por não valer a pena tal esforço. A partir dessa avaliação, é necessário retornar a conversa com o cliente para se necessário, realizar os ajustes na lista de requisitos (LAMOUNIER, 2007).

Assim que já se sabe quais os requisitos para o projeto e sua complexidade, chegou a hora de estimar os custos que estarão envolvidos no serviço. Quanto maior a complexidade do projeto, maior serão os custos para o fornecedor, e o preço cobrado ao contratante, isso, pois demandará mais tempo e mão de obra qualificada. Quaisquer que sejam as condições, elas precisam ser documentadas,

registradas em um contrato que será assinado pelas duas partes (LAMOUNIER, 2007).

A etapa de produção pode ser dividida em várias, levando em consideração a complexidade do projeto e metodologia escolhida para tal desenvolvimento (LAMOUNIER, 2007). O desenvolvimento de grandes sistemas normalmente exerce entregas fragmentadas, assim em vez de concluir tudo de uma vez, são liberadas, testadas e aprovadas diferentes partes do projeto aos poucos.

O projeto nunca vai para o mercado sem antes passar por um período de testes, é preciso ceder o código a utilização de usuários que vão testar todas suas funções, essa é a hora de encontrar erros (LAMOUNIER, 2007). É importante também testar a usabilidade e recomenda-se também, documentar os testes.

A etapa final, seria a implantação, para chegar neste ponto, é necessária que deva ocorrer mais uma etapa de testes para garantir que está tudo funcionando da maneira correta (LAMOUNIER, 2007). A implantação pode também ocorrer em diferentes fases, dependendo da necessidade, pode ser necessário sessões de treinamento para que o cliente aprenda a utilizar a solução que foi criada.

2.3 A IMPORTÃNCIA DO REGISTRO DE PONTOS PARA EMPREGADOS DOMÉSTICOS

Segundo a lei Complementar 150/2015 que rege o emprego doméstico, em seu artigo 12 “É obrigatório o registro do horário de trabalho do empregado doméstico por qualquer meio manual, mecânico ou eletrônico, desde que idôneo”, deixando clara a obrigatoriedade do empregador doméstico em realizar o controle de ponto do trabalhador. É importante ressaltar que, ao final do mês, após o preenchimento da folha de ponto, o empregado preciso assinar o documento. Isso evita o empregador de futuras ações trabalhistas.

2.3.1 PRINCIPAIS VANTAGENS DO PONTO ELETRÔNICO PARA EMPREGADOS DOMÉSTICOS

O ponto eletrônico possui funcionalidades que atendem todas as necessidades surgidas na rotina, seu ponto principal é a flexibilidade, graças a essa facilidade de registrar pontos, ideal para pessoas que desejam que o trabalho seja feito em horários específicos. Outro ponto fundamental é a velocidade no processamento de dados, com o ponto eletrônico, as informações são processadas de forma ágil no momento de contabilizar as horas trabalhadas. No final de cada mês, o programa gera uma folha de ponto completamente preenchida, dessa forma não será necessário digitar planilhas e relatórios no computador, o que reflete em economia de tempo para o empregador.

2.4 REACT-NATIVE

O React Native foi apresentado em uma conferência de React.js em 2015, segundo o Facebook a ideia era revolucionar a maneira como os aplicativos móveis eram desenvolvidos. Em sua versão inicial, havia suporte apenas para o sistema IOS, mas posteriormente foi adicionado o suporte para Android.

Como mencionado pelo Facebook, o objetivo do React Native é facilitar o desenvolvimento dos aplicativos móveis, pois para isso era necessário desenvolver

dois aplicativos, um para a linguagem nativa do Android (Java) e outro para a linguagem nativa do IOS (Swift e Objective-c).

É notável a crescente demanda do uso de aplicativos móveis nos últimos anos. Pensando nisso, diversas empresas desenvolvem suas soluções para facilitar a criação destes aplicativos (ANDRADE, 2020). Com o React Native, o Facebook lançou uma solução incrível para o desenvolvimento multiplataforma utilizando apenas código Java script, o que facilita os desenvolvedores que já possuem conhecimento em desenvolvimento web e até os que não possuem, já que é um framework com alta curva de aprendizado.

2.5 INTEGRAÇÃO DE APLICAÇÕES.

Em tempos de alto valor dos dados e informações que um sistema utiliza para a execução de alguns processos, é necessário garantir a interoperabilidade e um bom fluxo de informações. Dessa forma, a integração via API (Application Programming Interface) do projeto foi desenvolvida em NodeJS. O NodeJS é uma poderosa plataforma para aplicações, para construir fácil e rapidamente aplicações de rede escaláveis, e que utiliza engine JavaScript open source de alta performance do navegador Gooogle CHrome, o morot V8 (https://developers.google.com/v8/), que é escrita em C++ (MORAES, 2018).

Por ter sido projetado para construção de aplicativos de rede, é possível desenvolvermos para qualquer protocolo: DNS, FTP, HTTP, HTTPS, SSH, TCP, UDP ou WebSockets (MORAES, 2018).

Nesse projeto, utilizamos o protocolo HTTP. Vemos o NodeJS ser frequentemente utilizado para construir aplicações web, mas as suas aplicações vão além, como escrever ferramentas de linha de comando, e utilizando o projeto NW.js (https://hgithub.com/nwjs/nw.js), antigamente chamado de Node Webkit, podemos também desenvolver aplicativos nativos para o sistema operacional, compilando um mesmo código-fonte para Windows, Linux ou OS X (MORAES, 2018). Para o projeto em questão, utilizamos o React-Native Framework (https://reactnative.dev/), onde desenvolvemos um aplicativo para Android e IOS, por exemplo.

O NodeJS se trata de uma plataforma server-side, que promete reduzir carga de processamento na máquina do servidor através de uma arquitetura que atende todas as requisições feitas ao servidor de modo que não ocorram bloqueios.

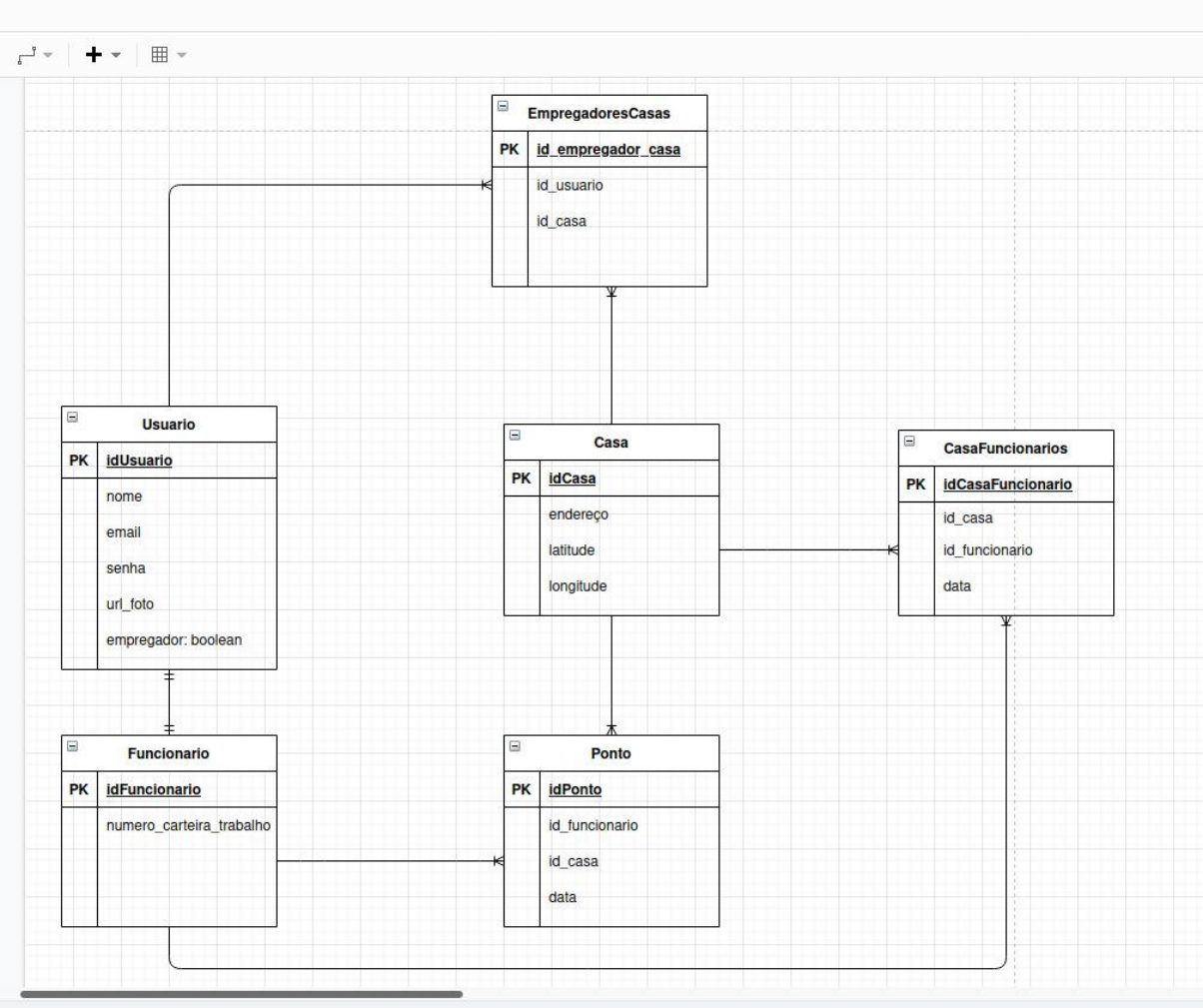
Com o NodeJS é possível criar aplicações Javascript para rodar como uma aplicação *standalone* em uma máquina, não dependendo de uma browser para a execução, como estamos acostumados. Apesar de recente, o NodeJs já é utilizado por grandes empresas no mercado de tecnologia, como Netflix, Uber e Linkedin (LENON, 2018).

O principal motivo de sua adoção pelo mercado é sua alta capacidade de escala, além disso, sua arquitetura e baixo custo, o tornam uma boa opção para implementação.(LENON, 2018)

2.6 BANCO DE DADOS

O banco de dados é uma ótima ferramenta para armazenar dados, arquivos e outros. No nosso projeto planejamos a estrutura inicial pelo MER (Modelo entidade relacionamento). Na figura a seguir podemos analisar da seguinte forma, tem os usuários, que podem ser empregadores ou não e considerando que alguém possa ter mais de uma casa, ou negócio “EmpregadoresCasas” registra as casas do

empregador. Casa guarda as informações do local para usar quando o método for chamado “CasaFuncionarios” registra quem trabalha em qual casa.

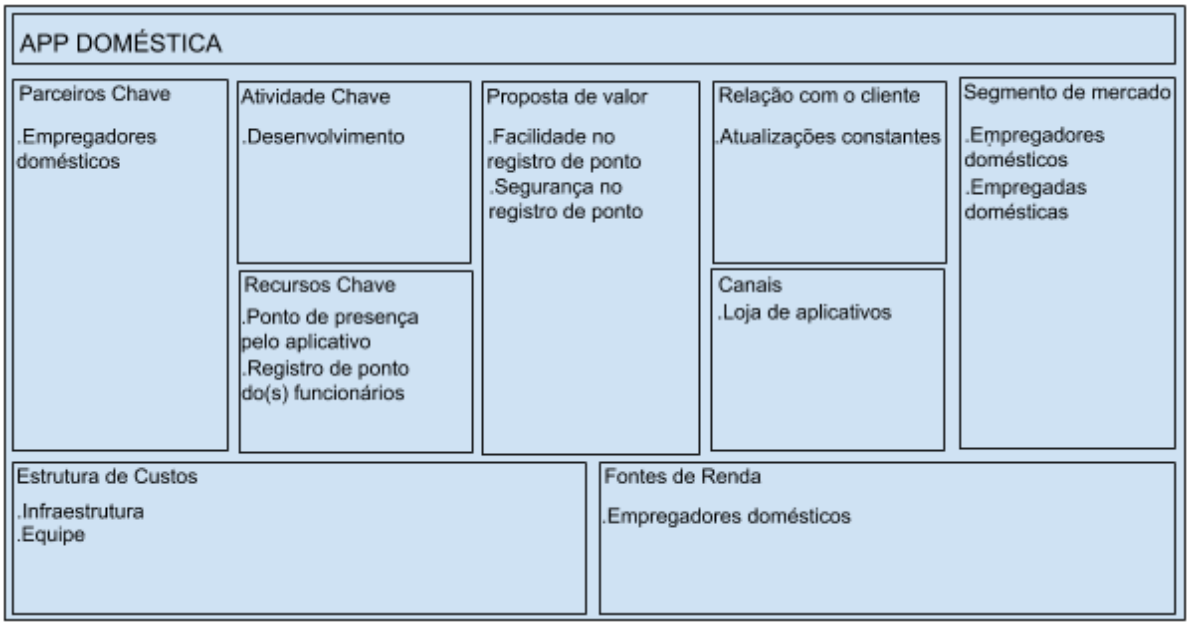


Fonte: Os autores.

3 MODELO CANVAS

O Business Model Canvas é uma ferramenta de gerenciamento estratégico, que permite desenvolver e esboçar modelos de negócios novos ou existentes. É um mapa visual pré-formatado contendo nove blocos do modelo de negócios. O Business Model Canvas foi inicialmente proposto por Alexander Osterwalder baseado no seu traba anterior sobre Business Model Ontology (OSTERWALDER, 2010). O Canvas nos traz a possibilidade de criar modelos de negócios a partir dos nove elementos que toda empresa ou organização possui, descrevendo o modelo de negócio da organização em questão. A Figura 1 exibe o modelo Canvas elaborado para este projeto.

Figura 1 – modelo canvas.



Fonte: Os autores.

3.2 DETALHANDO O MODELO CANVAS DO PROJETO

Abaixo serão listados e explicados os nove tópicos que compõem o modelo canvas apresentado.

(1) Parceiros Chave: Apresenta os recursos principais adquiridos fora da empresa. Em nosso projeto nosso Parceiro chave são os empregadores domésticos e também podemos adicionar futuramente empresas de contratação nesse setor, eles pagaram mensalidade para estar no aplicativo, assim o app se auto sustenta, podendo trazer novos recursos e melhorias.

(2) Atividade Chave: Apresenta as atividades essenciais para conseguir entregar a proposta de valor. Em nosso projeto a atividade chave é o desenvolvimento do app para que ele se torne real e de bons frutos.

(3) Proposta de valor: Apresenta o que a empresa vai oferecer para o mercado e que terá valor para os clientes. Em nosso projeto nossa proposta de valor é a facilidade no registro de ponto para o(a) colaborador(a) para que ele(a) foque apenas no serviço e também na segurança do registro para o empregador(a) que só registrará no local salvo pelo próprio empregador(a).

(4) Relação com o cliente: Apresenta como que se relacionará com cada segmento. Em nosso projeto nossa principal relação será com atualizações constantes informando e ensinando as novas funcionalidades através do app.

(5) Segmento de mercado: Apresenta Quais segmentos de clientes será foco da sua empresa. Em nosso projeto o segmento de mercado vão englobar tanto os empregadores quanto os colaboradores.

(6) Recursos Chave: Apresenta os recursos necessários para realizar as atividades-chave, Em nosso projeto são o ponto de presença pelo aplicativo e todos os registros dos funcionários.

(7) Canais: Apresenta como o cliente compra e recebe seu produto e serviço. Em nosso projeto por ser desenvolvido em plataforma híbrida será adicionado nas lojas de aplicativos do google e na loja de aplicativos da Apple.

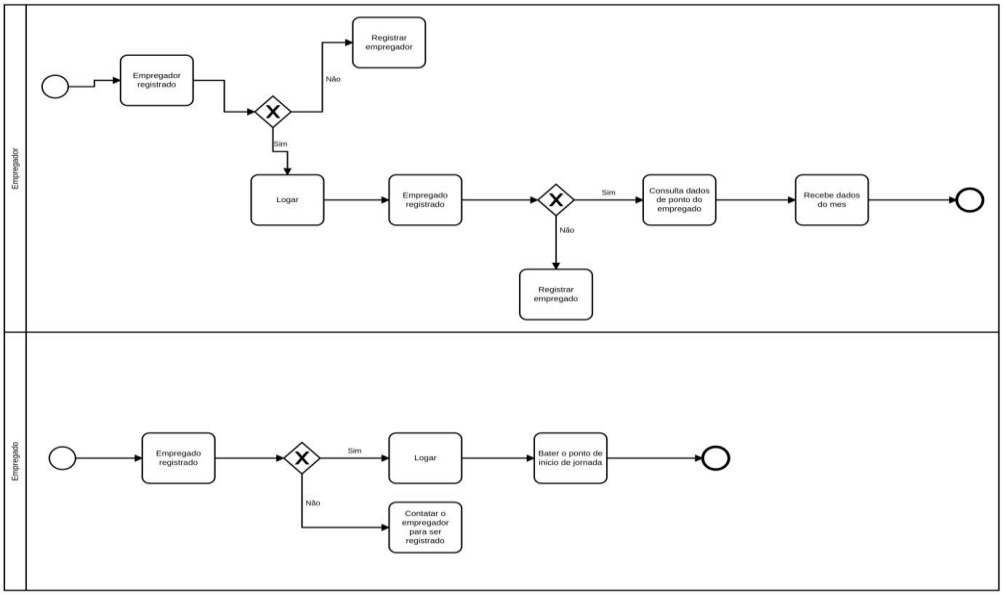
(8) Estrutura de custos: Apresenta os custos relevantes necessários para que a estrutura proposta possa funcionar. Em nosso projeto os custos são formados pela infraestrutura com hospedagem de app nas lojas para serem baixadas online e também com hospedagem do banco de dados para salvar as informações dos registros e afins do app, e também com gastos com o desenvolvimento do projeto conforme ele vai crescendo e tendo que contratar uma equipe.

(9) Fontes de renda: Apresenta as formas de obter receita por meio de propostas de valor. Em nosso projeto a princípio só será cobrado dos empregadores.

4. MODELAGEM DO NEGÓCIO COM BPMN

O *Business Process Modeling Notation* (BPMN) (em português Notação de Modelagem de Processos de Negócio) é uma notação da metodologia de gerenciamento de processos de negócio e trata-se de uma série de ícones padrões para o desenho de processos, o que facilita o entendimento do usuário. A modelagem é uma etapa importante da automação pois é nela que os processos são descobertos e desenhados. É nela também que pode ser feita alguma alteração no percurso do processo visando a sua otimização. A notação também pode ser utilizada para a modelagem de Arquitetura de Processos. O BPMN Foi desenvolvido pelo *Business Process Management Initiative* (BPMI) e atualmente é mantida pelo *Object Management Group* já que as duas organizações se fundiram em 2005. Em março de 2011, a versão atual do BPMN é a 2.0. (PIZZA, 2012**.)**

O BPMN do projeto App doméstica representa todo fluxo de processos da aplicação do empregador desde registrar o empregado ate a consulta de registro dos pontos registrados do(s) empregado(s). Já o do empregado vai desde o seu registro pelo empregador ate a efetuação do ponto de inicio de serviço ou almoço ou saída.

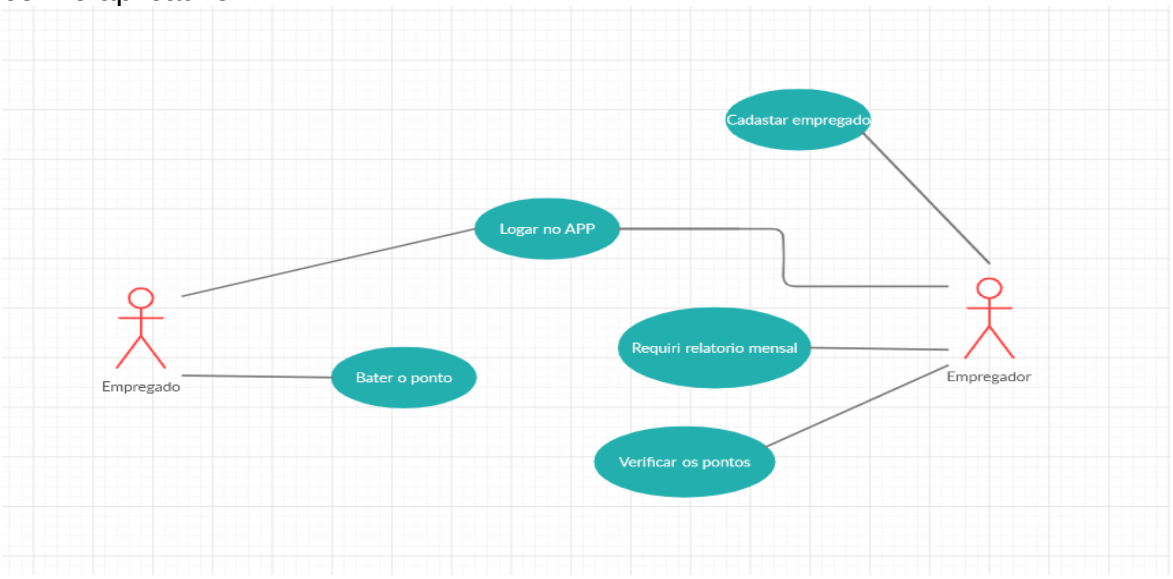


fonte: Os autores.

5 DIAGRAMA DE CASO DE USO

O objetivo dos casos de uso é a identificação das funcionalidades requeridas para o sistema. Assim, os casos de uso incluem-se na fase de ANÁLISE DE REQUISITOS, a fase em que procuramos identificar, da melhor forma possível, o que é que o nosso sistema (frequentemente um ou mais programas) deve realmente ser capaz de fazer. Como seria de esperar, esta fase do desenvolvimento do software tem a sua própria terminologia. (BARROS, 2009.).

O diagrama de caso de uso do projeto App domestica representa os fluxos de interações entre o empregador com o aplicativo e também o empregado com o aplicativo.



Fonte: Os autores.

6. TELAS

Neste tópico falaremos das telas do aplicativo, descrevendo suas funcionalidades.

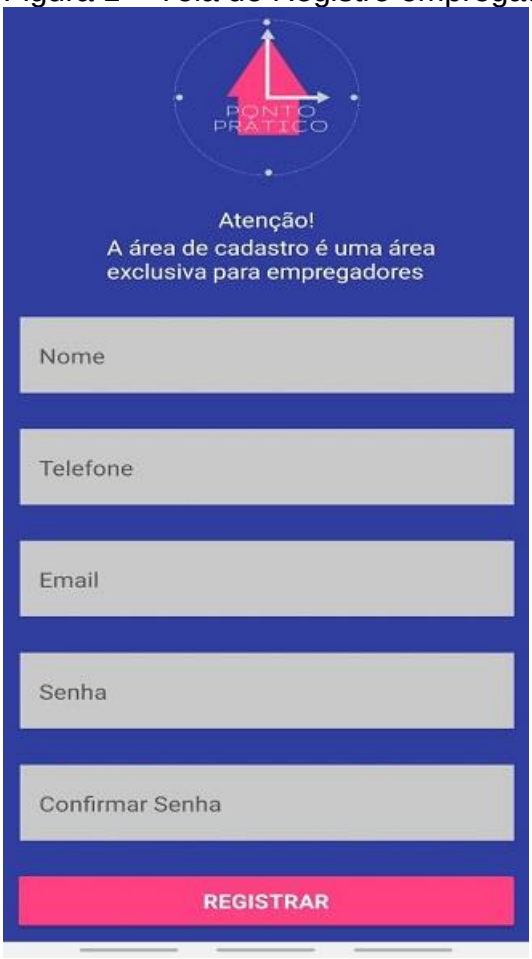
6.1 TELAS DA APLICAÇÃO E SUAS FUNCIONALIDADES

Nesta seção serão exibidas todas as telas da aplicação do App Domestica.

6.2 REGISTRO EMPREGADOR

Para que o empregador tenha acesso ao aplicativo primeiro ele tem que se cadastrar com os seguintes dados: Nome, Telefone, E-mail, cpf/cnpj, Senha. E depois ir para a tela de login e com os dados preenchidos corretamente se logar. Ná Figura 1 a seguir ilustra este cenário.

Figura 1 – Tela de Registro empregador.



6.3 LOGIN EMPREGADOR

Para o empregador logar primeiramente ele tem que escolher o input “Empregador” e depois inserir os seus dados (dados do formulário a preencher) só depois disso feito terá acesso às funcionalidades do APP. A Figura 2 a seguir ilustra este cenário.

Figura 2 – Tela login Empregador.

6.4 REGISTROS DOS PONTOS DO(S) FUNCIONÁRIO(S)

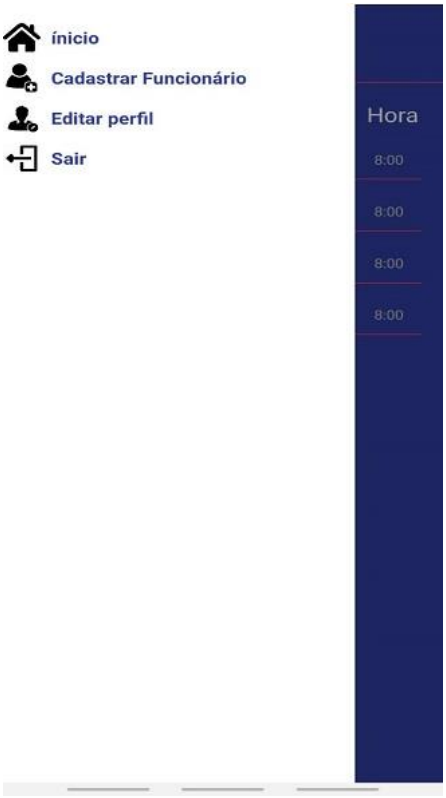
Esta é a tela de inicio para o empregador, ela fornece as principais informações para que ele gerencie os funcionários, nela aparecem as seguintes informações: Cards com todos os seus funcionários com imagem e status do que os mesmos se encontram fazendo, e em seguida uma tabela com os pontos registrados dos funcionários com: O tipo do ponto se é primeira entrada, ou hora do almoço ou segunda entrada, Nome do funcionário, o dia, e a hora. A Figura 3 a seguir ilustra este cenário.

Figura 3 – Tela Registro dos pontos dos funcionários.

6.5 MENU

Esta é a forma que o empregador terá para navegar entre as telas do APP, nela contem os seguintes links, Início, Cadastrar funcionário, Editar perfil, e o link para se deslogar do APP. A Figura 4 a seguir ilustra este cenário.

Figura 4 – Tela do Menu.



6.6 REGISTRAR FUNCIONARIO

Nesta tela o empregador define a foto se quiser do funcionário, e preenche com os dados do empregado o formulário de registro, neste formulário tem os seguintes campos a ser preenchidos: Nome do funcionário, cpf, senha, confirmar senha, registrar a localização em que o empregado irá trabalhar. A Figura 5 a seguir ilustra este cenário.

Figura 5 – Tela Registrar funcionário.

6.7 Editar Perfil

Nesta tela o empregador poderá editar o perfil próprio podendo alterar sua foto de perfil, e-mail e senha. A Figura 6 a seguir ilustra este cenário.

Figura 6 – Tela Editar Perfil.

6.8 LOGIN EMPREGADO

Para o empregado efetuar o login o empregador precisa registra-lo primeiro como já foi dito.

Depois o empregador passará o email e a senha para o empregado se logar no APP. A Figura 7 a seguir ilustra este cenário.

Figura 7 – Tela Login empregado.

6.9 TELA DE PONTO EMPREGADO

O empregado terá acesso apenas á essa tela e o menu só terá o link de sair. Nesta tela o usuário poderá visualizar os seguintes componentes, o botão do menu contendo apenas o link de sair do app, o botão de registrar ponto e uma tabela informando os pontos dele mesmo registradas. A Figura 8 a seguir ilustra este cenário.

Figura 8 – Tela de Ponto Empregado.

7. IMPLEMENTAÇÃO DA INTEGRAÇÃO DO FRONT-END COM O BACK-END

Neste tópico falaremos sobre a implementação do aplicativo APP Domestica.

7.1 FRONT-END.

Para o desenvolvimento do aplicativo foi utilizado o framework React Native. Ná figura 1 é responsável pelo carregamento inicial da aplicação. Esse arquivo foi responsável por importar o arquivo de configuração do Redux, denominado “store”, além de ser carregado através da *tag* “<Provider store={store}>”. Nas linhas 23 ate a 31da *tag* de configuração do redux, é utilizado algumas configurações para o estilo do sistema. A *tag* “<Router />” é zimportada do arquivo de configurações de rotas do aplicativo, esse arquivo possui todas as telas desenvolvidas da aplicação, sendo assim, possível navegação entre as mesmas. A Figura 9 a seguir ilustra este cenário.

Figura 9 - Fragmento da implementação do front-end.

```
import React, {Component} from 'react';
import {Root, StyleProvider, Container} from 'native-base';
import {Provider} from 'react-redux';
import {MenuProvider} from 'react-native-popup-menu';

import Router from './navigation';
import getTheme from './style/theme/components';
import domesticas from './style/theme/variables/domesticas';
import store from './store';

console.disableYellowBox = true;
let db;

class Master extends Component {
  constructor(props) {
    super(props);
  }

  componentDidMount() {}

  render() {
    return (
      <Provider store={store}>
        <StyleProvider style={getTheme(domesticas)}>
          <MenuProvider
            customStyles={{
              backdrop: {
                backgroundColor: '#000',
                opacity: 0.5,
              },
            }}>
            <Root>
              <Container>
                <Router />
              </Container>
            </Root>
          </MenuProvider>
        </StyleProvider>
      </Provider>
    );
  }
}

export default Master;
```

7.2 BACK-END

Para a construção da API da aplicação, foi utilizado o NodeJS, uma tecnologia assíncrona utilizando o Javascript.

Na figura 2 é apresentada a função responsável por realizar o acesso dentro de nossa aplicação, tanto do empregado como do empregador.

Na linha 58, é possível analisar uma verificação chamando a função denominada “*loginValidator*”, essa verificação é responsável por garantir que o usuário nos informou os dados necessários para acesso. Após a verificação é feito uma consulta na tabela “*User*” do banco de dados, para verificar se o *email* ou cpf indicado, realmente exista na base de dados do aplicativo, caso não existir é exibido uma mensagem.

Nas linhas 66 à 70 da função de login, é realizado uma verificação na senha informada pelo usuário, realizando uma comparação entre a senha informada e a senha criptografada que está inserida na base de dados. Caso a senha seja aceita, é retornado os dados do usuário, onde outro trecho do back-end realiza o processo de geração de token JWT, onde é usado no front-end para garantir acesso a rotas restritas da aplicação. A Figura 10 a seguir ilustra este cenário.

Figura 10 - Fragmento da implementação do back-end.

```
47  const makeLogin = async (form) => {
48    if (!loginValidator(form)) {
49      throw new HttpResponseError({
50        status: 401,
51        message: 'usuário e senha são necessários para o login',
52      });
53    }
54
55    const user = await User.query()
56      .where(pick(['email', 'cpf'], form))
57      .first();
58
59    if (!user) {
60      throw new HttpResponseError({
61        status: 401,
62        message: 'Usuário não consta em nossa base',
63      });
64    }
65
66    if (!(await bcrypt.compare(form.password, user.password))) {
67      throw new HttpResponseError({
68        status: 401,
69        message: 'Senha inválida',
70      });
71    }
72    console.log('user', user)
73    return user;
74  };
```

7.3 INTEGRAÇÃO DO FRONT COM ROTA DO BACK-END.

Para que front-end consiga ser integrado com back-end, utilizamos uma biblioteca conhecida na comunidade React, chamada Redux. Na figura 3 é apresentado uma função denominada “action” dentro do Redux, essa função é responsável por manipular os dados de acesso do usuário, posteriormente é passado a url da rota criada no back-end, juntamente com o método HTTP, no caso usamos o POST.

A linha de código 103 onde está descrito o. ”then” presente na figura 11, é acionado caso o back-end nos retorne sucesso. No caso do código em questão, quando a permissão é concedida, o código é responsável por armazenar os dados do usuário dentro da função nativa do React-Native AsyncStorage linha 106, assim podemos acessar esses dados do usuário em qualquer lugar da aplicação.

Figura 11 - Fragmento da implementação de consumo de rotas do front-end no back-end.

```
export function makeLogin(values) {
  return async (dispatch, getState) => {
    let value = '';

    values.cpf.trim();
    values.password.trim();

    if (values.cpf !== '') {
      value = {
        password: values.password,
        cpf: values.cpf.replace(/[^\d]+/g, ''),
      };
    }
    if (values.email !== '') {
      value = {
        password: values.password,
        email: values.email,
      };
    }

    return dispatch({
      type: types.AUTH_REQUEST,
      payload: {
        request: {
          url: 'login',
          method: 'POST',
          data: value,
        },
      },
    })
      .then(async ({payload}) => {
        const {user, token} = payload.data;

        await AsyncStorage.setItem(
          '@domesticas/user',
          JSON.stringify({...user}),
        );
        await AsyncStorage.setItem('@domesticas/token', token);
        setTimeout(() => Navigator.reset('Dashboard'), 700);

        // setTimeout(() => Navigator.reset('ChangePassword'), 700);
        // bugsnag.setUser(user.id.toString(), user.name, user.email);
      })
```

8. Conclusão

Devido à necessidade de reduzir custos com aparelhos físicos de ponto eletrônico que não tem tanto controle pelo empregador, sentimos que tinha a necessidade de desenvolver um aplicativo para facilitar para o empregado bater o ponto e para o empregador ter um controle melhor das horas registradas pelos seus funcionários e também uma redução de gastos para os empregadores, precisando apenas de fazer o download do aplicativo APP Domestica, reduzindo também custos de energia que um aparelho de registrar ponto consome para funcionar.

No decorrer do desenvolvimento do aplicativo encontramos dificuldade em programar a integração com o e-social, e também dificuldade em programar o local por gps.

Como projeto futuro integrará com o e-social para que fique os registros salvos no sistema do governo e também gerar relatórios por este portal, também integrar com o Google maps para que o empregador possa saber a localização do empregado quando o ponto for registrado. Outros projetos futuros é fazer uma tela de vagas de emprego, podendo assim centralizar pessoas que estão contratando e pessoas que querem ser contratadas.

9 - Referências:

DEVMEDIA, Atividades básicas ao processo de desenvolvimento de Software. Disponível em: https://www.devmedia.com.br/atividades-basicas-ao-processo-de-desenvolvimento-de-software/5413. Acesso em: 10 jul. 2020.

DOMÉSTICA LEGAL, Obrigatoriedade do controle de ponto do empregado doméstico. Disponível em: <https://www.domesticalegal.com.br/obrigatoriedade-do-controle-de-ponto-do-empregado-domestico/#:~:text=A%20Lei%20Complementar%20150%2F2015,eletr%C3%B4nico%2C%20desde%20que%20id%C3%B4neo%E2%80%9D.>. Acesso em 08 jun 2020.

FERREIRA, Renata Bastos; LIMA, F. P. A. Metodologias Ágeis: Um Novo Paradigma de Desenvolvimento de Software. In: **II Workshop Um Olhar Sociotécnico sobre a Engenharia de Software–WOSES**. 2006.

FERREIRA, Metodologias Ágeis: Um Novo Paradigma de Desenvolvimento de Software. Disponível em: https://is.cos.ufrj.br/woses/2006/pdfs/10-Artigo10WOSES-2006.pdf. Acesso em 12 ago. 2020.

FIA. Desenvolvimento de softwares: O que é, como funciona e dicas. 2019. Disponível em: <https://fia.com.br/blog/desenvolvimento-de-softwares> Acesso em 23 de abri. De 2020.

GEEKHUNTER, Um guia completo de React Native. Disponível em: <https://blog.geekhunter.com.br/um-guia-completo-de-react-native/>. Acesso em 15 jul 2020.

LENON, Node.js – O que é, como funciona e quais as vantagens. Disponível em : <https://www.opus-software.com.br/node-js/>. Acesso em 02 set. 2020.

MORAES, William Bruno. Construindo aplicações com NodeJS. Novatec Editora, 2018.

OSTERWALDER, Alexander; PIGNEUR, Yves. Business model canvas. Self published. Last, 2010.

PRESSMAN, Roger; MAXIM, Bruce. **Engenharia de Software-8ª Edição**. McGraw Hill Brasil, 2016.