TOUCH: PLATFORM FOR LOCATION AND NOTIFICATION IN AN ENVIRONMENT WITH HIGH FLOW OF PEOPLE. TOUCH: PLATAFORMA PARA LOCALIZAÇÃO E NOTIFICAÇÃO EM UM AMBIENTE COM ALTO FLUXO DE PESSOAS.

Abstract: With the emergence of the "mobile society", the exponential use of cell phones due to the possibility of maximized communication, has become a powerful tool for tracking and improving data, including whether or not an individual remains in a given location through geolocation. Considering an IES (Higher Education Institution) as a scenario with a high flow of people, it is necessary to create mechanisms capable of assisting students in their search for students, especially in the teachers' room. In this way, an integrated platform called Touch was developed to meet this need. This consists of two applications working in an integrated manner using the same database using the Firebase tool (version 2.3) which provides the possibility of using a cloud database with the Realtime Database (RD) feature and a Virtual Private Server (VPS) to host the WEB application, with the backend as a communication layer between the Mobile applications and the database written in GoLang (version 1.10), responsible for handling all business rules, necessary validations and with the ability to communicate via HTTP protocol. The Touch was made with the intention of facilitating the meeting of teachers by the employees through its geographical location, thus reducing the disturbances that occurred in the need to contact teachers. Although it has some limitations related to human use and technical implementation, in initial tests Touch proved to be effective when requested, promoting real-time updates on the teacher's status, demonstrating the importance of implementing technologies to assist manual tasks to enable better organization and use of human labor, making it more productive.
Keywords: Mobile Devices. Geolocation. Feedback.

Resumo: Com a emergência da "sociedade móvel", o uso exponencial dos celulares por conta da possibilidade de comunicação maximizada, se tornou uma poderosa ferramenta para o rastreio e aprimoramento de dados, incluindo a permanência ou não de um indivíduo em determinado local por meio da geolocalização. Considerando uma IES (Instituição de Ensino Superior), como um cenário com alto fluxo de pessoas, faz-se necessária a criação de mecanismos capazes de auxiliar na procura de discentes por parte de alunos, principalmente na sala dos professores. Desta forma, real desenvolveu-se uma plataforma integrada denominada *Touch* para atender essa necessidade. Esta é composta por duas aplicações trabalhando de forma integrada fazendo uso da mesma base de dados por meio da ferramenta *Firebase* (versão 2.3) que provê a possibilidade de uso de *cloud database* com a característica de *Realtime Database* (RD) e um *Virtual Private Server* (VPS) para hospedar a aplicação WEB, tendo o *backend* como uma camada de comunicação entre as aplicações *Mobile* e a base de dados escrita em *GoLang* (versão 1.10), responsável por tratar todas as regras de negócios, validações necessárias e com a capacidade de se comunicar através de protocolo HTTP. O *Touch* foi feito com intuito de facilitar o encontro de docentes por parte dos funcionários através da sua localização geográfica diminuindo desta forma os transtornos ocorridos na necessidade de entrar em contato com professores. Ainda que possua algumas limitações relacionadas ao uso humano e implementação técnica, em testes iniciais o *Touch* se mostrou eficaz quando requisitado, promovendo atualizações em tempo real quanto ao estado do professor, demonstrando a importância da implementação de tecnologias para auxiliar tarefas manuais para possibilitar uma melhor organização e aproveitamento da mão de obra humana, tornando-a mais produtiva.

Palavras-chave: Dispositivos móveis; Geolocalização; *Feedback.*

1 INTRODUÇÃO

Com o advento dos Dispositivos Móveis (DMs), a facilidade de acesso às informações e à comunicação tornou-se cada vez mais evidente, permitindo a qualquer momento que os indivíduos sejam contactados, sendo a conectividade tida como uma maneira efetiva deste manter-se acompanhado, de aprender e de interagir sem que haja necessariamente uma teoria para tanto. Nesse sentido, os DMs acabaram por se tornar objetos inseparáveis, fazendo cada vez mais parte do dia-a-dia, permitindo assim uma maior facilidade do consumo de conteúdo, interação com as capacidades suas computacionais, de comunicação e da facilidade de captar facilmente um contexto através das mídias por áudio, vídeo, tempo e localização (Carvalho, 2015).

Novos recursos relacionados a tecnologia foram criados com o intuito de simplificar as experiências da vida cotidiana e das tarefas rotineiras, têm sido o foco de pesquisadores de indústrias, no objetivo de solucionar problemas do usuário, tornando as tecnologias das quais estes operam o mais transparentes possíveis, necessitando assim de uma mínima interação e, em decorrência disso, de ações simples como um toque, gesto e comando de voz, por exemplo (Cruz, 2013). Segundo Carvalho (2017), graças aos sistemas de geolocalização em tempo real ocorreu o crescimento da tecnologia móvel e a maior interação entre os espaços físico e digital, a exemplo do Sistema de Posicionamento Global (GPS), desenvolvido em 1972, a facilidade de se utilizarem diversos sistemas de informação e de aplicações ou serviços *mobile* tornou-se ainda mais palpável. Comumente, aplicativos de geolocalização informam a localização de um usuário para outros, dependendo de um conjunto de coordenadas de latitude e longitude, e fazem a associação de locais reais, sendo executados por meio de DMs em uma experiência bem mais real se comparado a *desktops* por conta do recebimento e envio de informações mudarem de acordo com a mudança da localização do indivíduo (Ionesco, 2018).

Nesse sentido, levando-se em conta o cenário de uma IES (Instituição de Ensino Superior), foi desenvolvida uma plataforma integrada – denominada Touch, que facilite o encontro de docentes por parte dos funcionários através da localização geográfica destes, em um ambiente com alto fluxo de pessoas, dando um retorno para o discente de forma mais rápida (professor presente ou ausente) diminuindo desta forma os transtornos ocorridos no processo de entrar em contato com os professores, seja na demora do atual processo em que os funcionários se locomovem dentro do ambiente e, ou frequentemente a falta de informações necessárias acerca da pessoa que está sendo solicitada, dentre outros.

A plataforma proposta é composta por dois aplicativos integrados construída tendo como base a arquitetura Cliente / Servidor. Os dois aplicativos (atendente e professor) serão construídos com o auxílio do *framework Ionic*, e compilados com o *Cordova Apps*.

Este trabalho foi dividido entre 5 seções principais. Sendo que a seção 1 apresenta uma breve introdução que apresentando os aspectos relevantes deste trabalho; na seção 2 serão abordados os referenciais teóricos que darão embasamento à solução proposta; a seção 3 será descrita a metodologia utilizada para o desenvolvimento da solução; a seção 4 descreverá o processo de desenvolvimento da plataforma proposta, considerando as regras de negócio e os requisitos do sistema e gerando um protótipo da plataforma *Touch*; bem como apresentará os resultados do processo de testes e avaliação dessa proposta; e, finalmente na seção 5 apresentar-se-á a conclusão deste trabalho.

2 REFERENCIAL TEÓRICO

2.1 A EVOLUÇÃO DOS DISPOSITIVOS MÓVEIS

Nas últimas décadas, o significativo aumento do uso de Dispositivos Móveis (DMs) para as mais diversas finalidades, tornou-se cada vez mais evidente. Dispositivos como o *Personal Digital Assistants* (PDAs) e celulares, se fizeram cada vez mais comuns no cotidiano dos indivíduos, mesmo que no início fossem fabricados com finalidades bem específicas e pré-programadas em seu *hardware* e que restringiam sua utilização. Por exemplo, os primeiros dispositivos MP3 que tinham por finalidade única a reprodução de arquivos de áudio e mesmo os primeiros celulares, que à época, apenas realizavam chamadas de voz. Porém, o crescimento exponencial das funcionalidades relacionadas ao poder computacional, trouxeram à tona novos ambientes de programação relacionados aos DM, possibilitando o desenvolvimento de aplicações de forma mais independente e diversificada, demonstrando uma "generalização" da utilização deles. Assim, aplicações feitas antes apenas para *desktop* passaram a ser disponibilizadas para outras plataformas, permitindo acesso à internet e tornando o alcance a informações mais fácil e rápido (Oliveira; Medina, 2007).

A emergência da "sociedade móvel" foi se tornando mais notória, em especial pela crescente conectividade e da variabilidade das fontes de comunicação e informação disponibilizadas em casa, no trabalho, na escola e nas ruas. Desta forma, conectividade e mobilidade estão intrinsicamente ligadas, especialmente a partir do desenvolvimento das redes sem fio e da ocorrência de tecnologias nômades relacionadas a celulares, *notebooks* e *tablets*. Dispositivos estes, que não apenas instrumentos para comunicação entre pessoas, mas também potentes sistemas de informação para produzir, receber e disseminar diversos conteúdos das mais variadas fontes. Os *smartphones* que antes incorporavam de forma prioritária os serviços de voz, passaram então a evidenciar um leque maior de informações textuais e a conexão com a web, tornando mais fácil a publicação de informações sem que estas necessitem de uma edição dos meios de comunicação em massa e sendo disseminadas por qualquer pessoa, a qualquer hora e lugar (Mantovani; Dantas, 2010).

Tendo em vista a necessidade da computação móvel, especialmente no meio acadêmico, o Instituto de Pesquisa de Stanford (*Stanford Research Institute*) performou uma pesquisa relacionando o uso de DMs nesses ambientes, obtendo como resultado que seu uso proporcionou maior motivação de aprendizado, além de apontar também uma maior colaboração e comunicação entre estudantes e professores, pois a tecnologia atua como uma ferramenta facilitadora no acesso a obtenção de conteúdo sem hora ou local previamente estabelecidos (Bartholo; Amaral; Cagnin, 2009).

No que se refere aos DMs e sua relação com locais reais, surge o contexto da Geolocalização que é definida como uma das ramificações de pesquisa relacionadas à tecnologia, tida como onipresente na computação ciente do contexto (*context-aware computing*), propondo uma confecção de dados de forma automática para dispositivos capazes de mostrar condições reais do usuário no ambiente ao qual está inserido por meio do DM que este utiliza. Nesse sentido, levando em conta que os celulares são os dispositivos mais utilizados, e da possibilidade de fornecerem uma comunicação maximizada pelas tecnologias sem fio, é possível que o rastreio de quem o possui seja feito de forma mais eficiente, com possibilidade e aprimoramento de dados relacionados à permanência de um

indivíduo em determinado local, controlando sua busca e localização, relacionando-a à pontos geográficos (Ferreira; Moreira; Mozzaquatro, 2011).

Atualmente, existem diversas opções de ferramentas para geolocalização como o *Mappy*, *ViaMichelin* e *Google Maps*, por exemplo, que utilizam grande quantidade de imagens para mapeamento eficiente por meio de endereços de roteamento de rede ou GPS (Sistema de Posicionamento Global) para um determinado local por meio de dados relacionados a latitude e longitude (Lemes; Dias, 2014).

2.2 DESENVOLVIMENTO MOBILE

Segundo Oliveira (2017), a parcela de consumidores brasileiros que utilizam *smartphone* cresceu mais de quatro vezes nos últimos 5 anos, enquanto em 2012 apenas 14% da população brasileira utilizava celular. Já no ano de 2017, a taxa de uso dos aparelhos atingiu 62%, segundo o estudo *Google Consumer Barometer*. Essa evolução também impactou os desenvolvedores, que precisaram desenvolver ferramentas e metodologias para criar aplicativos de forma mais rápida e menos onerosa. De acordo com a CronApp (2018), existem duas diretivas para desenvolvimento de aplicativos *mobile*: Híbrido e Nativo.

O desenvolvimento nativo consiste na técnica de se utilizar os recursos disponibilizados pelas próprias empresas detentoras do Sistema Operacional (SO) em questão. Como exemplo, o Google que disponibiliza o *Android SDK (Software Development Kit)* e a Apple que disponibiliza o *IOS (Iphone Operational system) SDK*. Estes *SDK`s* fornecem todas as funcionalidades para desenvolver com base em uma plataforma específica (Cordeiro, 2017). Já o desenvolvimento híbrido nasceu da necessidade de se desenvolver apenas uma base de código que é executável em diversos ambientes diferentes (cenário este que é impossível de se realizar com o desenvolvimento nativo, pois as SDK's são específicas para a plataforma em questão), a princípio sem se preocupar com as especificidades de cada uma, é uma forma de desenvolvimento menos custosa para se criar e manter, pois não se faz necessário mão de obra para uma plataforma específica (Madureira, 2017).

Com o decorrer da evolução tecnológica, as formas de desenvolvimento de *software* tenderam a seguir princípios para garantir alguns aspectos básicos e ter alguma chance de competitividade no mercado. Hoje, dependendo do nível de complexidade do *software*, se faz necessária a estruturação do mesmo em uma arquitetura bem definida para proporcionar algumas características benéficas para o processo de desenvolvimento (Aladdin, 2018).

Para desenvolvimento da plataforma *Touch* a arquitetura de *software* escolhida foi o SOA (*Service-Oriented Architecture)* que se baseia nos princípios da comunicação distribuída, servindo como um modelo de se planejarem estratégias na área da Tecnologia da Informação por meio do paradigma de solicitação e resposta, para comunicar sistemas clientes e sistemas implementadores de serviços. Este é um padrão de arquitetura onde componentes presentes em aplicativos promovem serviços a outros componentes usando como "ponte" um protocolo para comunicação, afirmando que as funções implementadas pelas aplicações devam estar disponíveis em forma de serviços que são frequentemente ligados por meio de um barramento de serviços (*enterprise service bus*) para permitir que interfaces sejam acessadas por meio de *web services* (Barry, 2012).

Essa arquitetura tem os serviços como sua característica central e, juntamente destes, devem ser delimitados os modelos de projetos de modo a assegurar a integração de todos os

processos, a reutilização e a interoperabilidade entre os negócios e a plataforma de tecnologia, que deve estar disponível no intuito de permitir que os serviços sejam feitos de forma confiável e segura além de também permitir que a arquitetura de TI (Tecnologia da Informação) existente possa evoluir, a exemplo dos sistemas legados. Em outras palavras, o SOA torna-se uma estratégia que afeta as organizações de um modo geral, necessitando de uma visão ampla e de que as "metas" sejam comunicadas a todos os envolvidos, desde usuários do negócio até os arquitetos de TI (Furtado C. , Pereira, Azevedo, Baiao, & Santoro, 2009).

A plataforma *Touch*, por ser distribuída e utilizar o modelo Cliente/Servidor, faz uso do protocolo HTTP (*HyperText Transfer Protocol*) para realizar troca de informações entre as partes. Este protocolo foi idealizado no início de 1990, e conta com um sistema de comunicação em rede que detém diversos protocolos, utilizado para a promoção da transferência de dados entre computadores de cunho mundial no intuito de fornecer serviços, tendo um documento, quando completo, composto por diferentes sub-documentos adquiridos, a exemplo de textos, imagens, vídeos, *scripts* e descrição de *layout* (Mosharraf; Forouzan, 2012).

Para que os dados possam ser enviados, é necessário que estes sejam transferidos pela *Web* com auxílio de protocolos TCP (*Transmition Control Protocol*) e de IP (*Internet Protocol, Protocolo de Internet*) para a conexão cliente-servidor por meio de *sockets* TCP/IP. Essa comunicação ocorre pela troca de mensagens que vai usualmente de um navegador *Web*, chamada de solicitação/requisição, quando enviada pelo cliente e o *feedback* (retorno) por meio do servidor, de resposta (Pillou, 2008).

O HTTP utiliza usualmente a porta 80 para interação entre sites por meio de uma linguagem HTML (*Hipertext Markup Language*), mas não restringindo-se somente a esta. Com relação ao acesso a determinado tipo de documento, é necessário que este seja pesquisado num site a partir de *links*, onde o endereço (URI - *Universal Resource Indentifier*) deve ser digitado. É importante ressaltar que URI e URL (*Universal Resource Locator*) são diferentes, visto que o segundo é um tipo de URI mas que pode ser localizado de forma direta (Copes, 2018).

Por se tratar de uma plataforma distribuída, o *Touch* conta com diversos componentes heterogêneos que precisam estabelecer uma comunicação uns com os outros. Um destes componentes é uma API *(Application Programming Interface)* responsável por executar todas as regras de negócio da plataforma. Esta API foi implementada utilizando REST *(Representational State Transfer)* que é um modelo de arquitetura de *software* distribuído, designado para extrair vantagens de protocolos já existentes, pode ser utilizado sob qualquer protocolo e, em aplicações *WEB* geralmente é implementado com base no HTTP. Este modelo foi definido pelo *Dr. Roy Fielding* na sua dissertação de mestrado em 2000 e ganhou notoriedade pela camada de flexibilidade de tratar múltiplos tipos de requisições além retornar diferentes informações em diferentes formatos. Esta flexibilidade proposta permite que o desenvolvedor construa APIs que seja capaz de atender a necessidade de diversos clientes ao mesmo tempo, pois possui a possibilidade de enviar os dados requisitados da maneira que o cliente solicitar, além de abstrair algumas informações inerentes a implementação como nome de parâmetros, métodos, entre outros (Mulesoft, 2017).

Uma das arquiteturas de *softwares* bastante difundida na comunidade de desenvolvimento *mobile* é o MVP *(Model-View-Presenter)* que foi criado no início dos anos 1990 na Taligent, uma *joint venture* da Apple, IBM (*International Business Machines*) e HP (*Hewlett Packard*). Esta arquitetura dá suporte para a criação de interfaces gráficas e a atualização destas sem precisar misturar as camadas de visualização e negócio do *software*, por conta de tais características a utilização desta arquitetura foi escolhida para o desenvolvimento da plataforma *Touch* (Thiengo, 2017).

Paralelamente, novas tecnologias visando otimizar o processo de desenvolvimento passaram a ser cada vez mais criadas e difundidas, a exemplo do *GoLang* que, segundo Souza (2018), é uma linguagem *Open Source* concisa, clara e eficiente apresentada pela Google *Inc.* no ano de 2009 e mantida pela comunidade desde então. É notável a semelhança sintática com a linguagem C, porém se destaca pela biblioteca *built-in* que oferece um ambiente confortável para o desenvolvimento de *software* moderno e pelo fato de possuir apenas 25 *keywords*. O projeto inicial da linguagem foi feito em setembro de 2007 por Robert Griesemer, Rob Pike e Ken Thompson e nasceu com a premissa de oferecer um tempo de compilação e *deploy* reduzido para aplicações com um grande código fonte (Lauden, 2018). Desta forma, estas características citadas foram primordiais e suficientes para a determinação da escolha desta API para a plataforma.

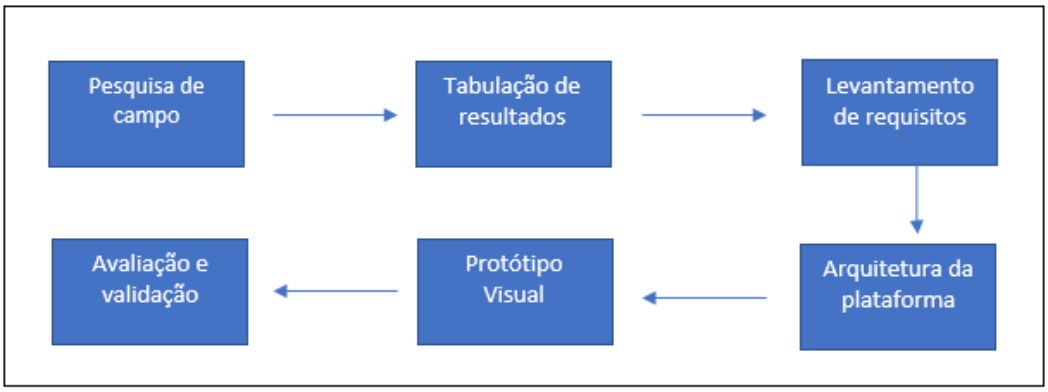
Como alternativa de armazenamento de dados, utilizou-se o *Firebase*, que é uma plataforma de desenvolvimento de aplicativos para dispositivos móveis e *WebApps* desenvolvida pela *Firebase* Inc. em 2011 e adquirida pela Google em 2014. Durante o desenvolvimento deste projeto, foi feito uso da funcionalidade de *Realtime Database*, que é uma base de dados orientada a documentos na nuvem que mantém uma conexão *WebSocket* aberta com o *Client*, através disto consegue disparar eventos que são tratados e recuperados pelo *Client*, possibilitando uma renderização assíncrona e orientada a eventos. Além também da funcionalidade *Push Notification*, que possibilita enviar uma notificação personalizada para o dispositivo de um usuário específico (Adriano, 2018).

Em aplicações com uma complexibilidade elevada, requerimentos específicos são necessários para executar e testar estas aplicações. Com o intuito de encapsular todos esses requerimentos e que possa ser executado em qualquer ambiente, criou-se o conceito de *container* que, segundo a *RedHat* (2017), possui o objetivo de criar independência e garantir a execução de aplicações separadamente, visando otimizar o uso da infraestrutura e segurança ao manter estas em ambientes isolados. A implementação de *container* utilizada para compor o ecossistema da arquitetura do *Touch* é o *Docker* que, é uma plataforma *Open Source* de alto desempenho escrita em *Go* que facilita a criação e administração de ambientes isolados. O *Docker* se baseia em um arquivo chamado *Dockerfile* para criar as imagens, além de oferecer um ambiente capaz de gerencia-las e disponibilizadas para qualquer pessoa ao redor do globo (Diedrich, 2015).

3 METODOLOGIA

Este trabalho é uma pesquisa-ação de cunho quali-quantitativo cujo desenvolvimento do protótipo, teste e validação, esta organizada conforme representado na Figura 1.

Figura 1. Representação do processo metodológico.



Fonte: do autor

a) Etapa 1 - Pesquisa de campo e validação da problemática: foram elaborados dois questionários eletrônicos, utilizando a plataforma *Typeform*. Um questionário para alunos e o outro para os funcionários, cada um com 5 questões fechadas, e aplicados entre os dias vinte e trinta do mês de agosto de 2019, sendo: 4 funcionários responsáveis pelo atendimento de alunos na sala dos professores e 50 alunos de uma universidade localizada em São Luís – Maranhão dos diversos cursos, no intuito de quantificar o problema relacionado com as dificuldades de entrar em contato com professores na IES em horários extraclasse;

b) Etapa 2 - Tabulação dos resultados iniciais: foi utilizado o programa *MS Excel* (2016) para organizar os dados obtidos por meio do questionário utilizado na pesquisa de campo com alunos e funcionários. Tais dados foram tabulados de forma a identificar os problemas mais recorrentes em relação à localização e notificação dos professores. Ou seja, os dados coletados apresentaram as dificuldades que os atendentes possuem em encontrar um professor quando solicitado e a confiança do aluno em relação a informação passada pelo atendente. A tabulação de dados foi utilizada posteriormente, em um estudo comparativo, para mensurar a efetividade do protótipo desenvolvido;

c) Etapa 3 - Levantamento e validação dos requisitos de *software*: com base nas informações coletadas na Etapa 2, foram identificados os requisitos funcionais, não funcionais e regras de negócios, que foram modelados utilizando os diagramas da UML (*Unified Modeling Language*). Tais diagramas, desenvolvidos com auxílio da plataforma *LucidChart*, serviram de base para o desenvolvimento e concepção da plataforma, documentando e delimitando o escopo implementado;

d) Etapa 4 - Definição e concepção da arquitetura da plataforma: a plataforma é composta por duas aplicações que trabalharam de forma integrada fazendo uso da mesma base de dados. Para esta integração, foi utilizada a ferramenta *Firebase* (versão 2.3) uma base de dados na nuvem que possui a característica de *Realtime Database* (RD) e um *Virtual Private Server* (VPS) para hospedar a aplicação WEB. O *backend* foi uma camada de comunicação entre as aplicações WEB/*Mobile* e a base de dados que foi escrita em *GoLang* (versão 1.10), responsável por tratar todas as regras de negócios, validações necessárias e com a capacidade de se comunicar através de protocolo HTTP.

O aplicativo para o atendente da sala dos professores foi desenvolvido utilizando o *framework Ionic* (versão 4.0) e portado para o S.O Android através do Cordova. Esta aplicação comunicou-se diretamente com o *Backend* através do protocolo HTTP. Já o aplicativo para o professor foi desenvolvido utilizando o *framework Ionic* (versão 3.0) e o *Ionic Material Design* para auxiliar na padronização do design dos componentes. Ambos aplicativos foram testados unitariamente através dos *frameworks* Karma (versão 2.0) e Jasmine (versão 3.0);

e) Etapa 5 – Desenvolvimento do protótipo visual: em relação aos requisitos de *software* e modelagem definidos anteriormente, o *software Adobe XD* (versão 10.0.12.9) foi utilizado para elaborar o desenho da plataforma, de forma a possibilitar uma noção da distribuição gráfica em relação às necessidades atendidas pelo *Touch* e das funcionalidades que o mesmo teria que disponibilizar;

f) Etapa 6 - Avaliação e validação da plataforma pelos usuários: ao final do processo a proposta de solução da plataforma *Touch* foi avaliada por 1 professor e quatro funcionários que tiveram contato mais direto com a plataforma. Para tanto, suas opiniões foram coletadas a partir de uma ferramenta inclusa no *Touch* para tal finalidade. Houve também uma pesquisa com os alunos para verificar se a qualidade e a precisão dos *feedbacks* apresentaram melhorias. Os aspectos avaliados foram referentes tanto à usabilidade da plataforma quanto às funcionalidades implementadas para a resolução dos problemas relacionados à localização e notificação descritos por esses atores no questionário situacional descrito na primeira alínea deste.

Com o intuito de facilitar o entendimento do processo a ser otimizado com o uso desta plataforma *Touch* será aqui apresentado o cenário (caso) quanto a disponibilidade/ disponibilidade da interação (atendimento) de docentes e discentes em uma sala de reuniões localizada dentro da sala dos professores de uma IES.

Segundo o NTI (Núcleo de Tecnologia da Informação) de uma Universidade localizada na cidade de São Luís - MA, com pouco mais de 700 professores ativos no Estado do Maranhão. Devido à grande quantidade destes profissionais, faz-se necessária a desenvolvimento/criação de mecanismos que auxiliem os atendentes na verificação de presença ou ausência dos docentes na sala dos professores, tendo em vista a alta movimentação de pessoas neste setor. Com isso, verificou-se a grande dificuldade de se localizar um determinado professor de forma eficaz e verificar sua disponibilidade de atendimento do discente ou convidado. Esta tarefa gera grande esforço por parte dos colaboradores responsáveis pelo atendimento, esforço este que poderia evitado e direcionado a realização das outras atividades que estes atendentes também executam.

Segundo relatos de alguns atendentes neste cenário ocorrem normalmente: a demora na localização do professor, ou a incerteza por parte do atendente sobre o professor solicitado estar ou não naquele ambiente, a demora do professor no atendimento causando um retorno de aviso do atendentes, isto acaba acarretando um *feedback* lento e impreciso para quem necessita comunicar-se com o professor. Para comprovação da gravidade do problema contextualizado acima, foram feitos dois questionários eletrônicos. O primeiro voltado para os quatro atendentes locados na sala dos professores com o intuito de capturar a opinião destes em relação ao processo de procurar um professor. E um segundo com cinquenta alunos desta Universidade, que frequentemente procuram os professores.

Os questionários foram aplicados entre os dias 20 e 30 de agosto de 2019. Os dados obtidos serviram de insumo para entendimento do panorama geral em relação a necessidade de procura de um determinado professor e o nível de confiança das informações fornecidas pelos atendentes.

Com base nas respostas dos alunos é possível observar que, quando os alunos foram questionados sobre a necessidade de contatar um professor em regime extraclasse, toda a amostra entrevistada respondeu que sim. 88% da amostra afirmou que já teve que esperar mais tempo do que o programado por um *feedback* relativamente simples como "o professor não se encontra". Em relação às informações erradas passadas pelos atendentes aos alunos 76% dos alunos afirmaram ter recebido uma informação equivocada, enquanto apenas 24% afirmaram que as respostas recebidas estariam corretas.

Quando perguntado que nota você daria para a qualidade da informação passada pelos atendentes, em uma escala de 0 a 10, em relação a prestação de seus serviços. Assim, apenas 4,8% atribuíram nota 10 para a confiança do *feedback* repassado, enquanto que ninguém atribuiu nota zero; 7,1 % dos entrevistados atribuíram nota 1; 2,4% atribuíram igualmente as notas 2, 3 e 4 cada; 16,7% a nota 5; 19% a nota 6; a maior parte da amostra acabou por dar nota 7 correspondendo a 28,6% dos entrevistados; 11,9% deram a nota 8 e por fim, 4,8% disseram-se mais satisfeitos, dando nota 9 em relação a sua confiança na resposta dos atendentes.

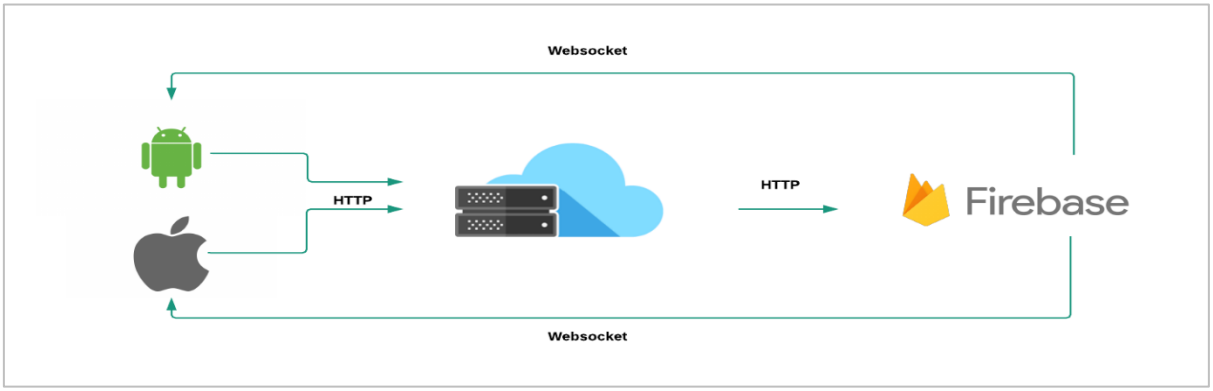
Agora analisando as informações oriunda dos atendentes, quando perguntados a respeito de possíveis problemas passados por conta do fornecimento de uma informação equivocada relacionada a ausência/presença de algum professor, verificou-se que bem mais da metade (70%) responderam que "SIM" tiveram vários problemas em decorrência de informações equivocadas ou tardias. Apenas 30% relatou que não há qualquer problema. Diante disso verifica-se a notória necessidade de meios informacionais mais alinhados para a diminuição desse tipo de erro. Quando questionados sobre o tempo para procurar pelos professores na sua respectiva sala, 80% dos atendentes admitiram que o procedimento adotado até então é demorado. Os 20% restante afirmaram que este procedimento é um pouco demorado. Diante do exposto, verificou-se a necessidade de se otimizar o sistema de atendimento docente/discente realizado na sala dos professores pelos atendentes.

4 DESENVOLVIMENTO DA PLATAFORMA *TOUCH*

Após a identificação do problema, iniciou-se o processo de desenvolvimento da plataforma *Touch* com o estudo dos requisitos, as regras de negócio e finalmente a prototipação visual, desenvolvimento do *software* e testes funcionais.

A plataforma *Touch* foi pensada em dois aplicativos mobile, um para o atendente e outro para o professor), que trabalharão de maneira integrada e complementar. Os aplicativos se comunicam diretamente com um *Virtual Private Server* (VPS) através de protocolo HTTP e, este VPS, é um servidor *Ubuntu* responsável por hospedar a API do *Touch*, se comunica também via HTTP com o *Firebase*, e para realizar a atualização dos componentes nos aplicativos o *Touch* mantém uma conexão *WebSocket* aberta com o *Firebase*, como exemplificado na Figura 2.

Figura 2. Representação gráfica da arquitetura da plataforma.

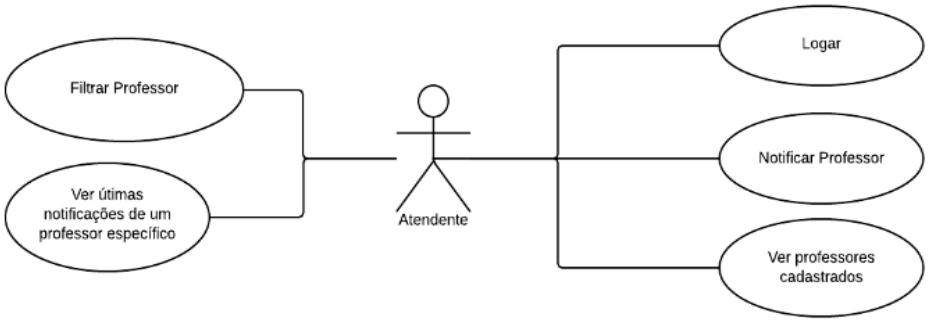


Fonte: do autor

A plataforma *Touch* se propõe a resolver a dificuldade de localizar um indivíduo específico em um local com alto fluxo de pessoas. Com base nos dados obtidos foi possível definir os requisitos funcionais e não funcionais, regras de negócio e atores com seus respectivos casos de uso.

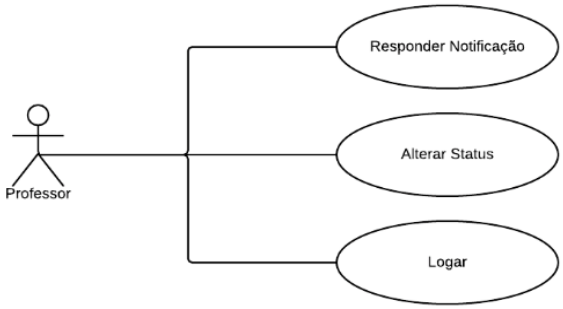
A utilização da plataforma ocorrerá em etapas sucessivas de modo a permitir o *login* no sistema, quando necessário, abrir uma notificação para algum professor de acordo com a necessidade de um aluno, verificar se o professor em questão já se encontra cadastrado na plataforma e assim, por fim verificar se suas últimas notificações para o professor foram ou não aceitas para então dar um *feedback* ao aluno, Figura 3. O outro papel identificado no processo se refere ao professor, com papel de receber as notificações abertas pelo atendente aceitando-a ou não, alterar sua disponibilidade, realizar *login* no sistema, Figura 4.

Figura 3. Caso de uso do Atendente e do Professor.



Fonte: do autor

Figura 4. Caso de uso do Professor.



Fonte: do autor

Por meio dos diagramas contidos nas Figuras 3 e 4 foi desenvolvido um conjunto de requisitos funcionais, não funcionais e regras de negócios que compõe a plataforma, estes requisitos possuem o objetivo de oferecer uma visão macro das especificidades e regras de negócio que regeram os *workflows*. Os requisitos não funcionais foram relacionados ao uso da aplicação em termos de desempenho, disponibilidade, confiabilidade, manutenibilidade e tecnologias envolvidas, as principais considerações são apresentadas no Quadro 1.

Quadro 1. Lista de requisitos não funcionais.

Código	Objetivo	Descrição
RNF01	Infraestrutura	O servidor deve estar hospedado na nuvem da *DigitalOcean* com o Sistema operacional *UbuntuServer* 16.04
RNF02	Infraestrutura	As aplicações da plataforma deverão rodar dentro de containers
RNF03	Aplicativos	Ambos os aplicativos devem ser escritos em plataformas híbridas de desenvolvimento
RNF04	Base de Dados	A base de dados deverá ser orientada a documentos
RNF05	Experiência do Usuário	As mensagens de *feedback* deverão ser exibidas em *box* com estilo nativo.

Fonte: do autor

Os requisitos funcionais foram relacionados com todas as funcionalidades que fazem parte da plataforma como um todo e as regras de negócios regeram as condições para estas funcionalidades serem executadas, como exemplificado no Quadro 2.

Quadro 2. Lista de Requisitos Funcionais e suas regras de negócio.

Código	Módulo	Ator	Funcionalidade
RF01	Atendente	Atendente	Notificar Professor
Regras de negócio			
O atendente deve estar logado; O professor deve estar com `{avaliable:true, inHome:true}` na base de dados; A notificação aberta ficará com o status de “pending” e terá a validade de 10 minutos, caso o professor não responda esta até o fim da validade, ela será excluída.			
Código	Módulo	Ator	Funcionalidade
RF02	Atendente	Atendente	Ver professores cadastrados
Regras de negócio			
A lista de professores será apresentada em ordem alfabética; Para evitar *overloading* e economizar dados, os carregamento dos professores será *lazy*.			
Código	Módulo	Ator	Funcionalidade
RF03	Atendente	Atendente	Logar no sistema
Regras de negócio			
O atendente deverá informar a matrícula e a senha, ambos os campos são obrigatórios; Os usuários deverão estar armazenados no *Firebase*.			
Código	Módulo	Ator	Funcionalidade
RF04	Professor	Professor	Responder Notificação
Regras de negócio			

As notificações poderão ser respondidas de duas maneiras: Recusar e Aceitar. Caso a opção seja "Recusar" a notificação mudará para o `{status:"rejected"}`.Caso a opção seja "Aceitar", a notificação mudará para o `{status:"accepted"}`.			
Código	Módulo	Ator	Funcionalidade
RF05	Professor	Professor	Logar no Sistema
Regras de negócio			
Após efetuar o *login.* A localização deve está ativada, caso contrário será exibido a mensagem: "Você precisa habilitar a Localização para continuar usando o Touch".			
Código	Módulo	Ator	Funcionalidade
RF06	Professor	Professor	Alterar Status
Regras de negócio			
O status pode ser alterado entre Disponível e Indisponível, caso o professor esteja indisponível, não será possível abrir notificações para ele.			

Fonte: do autor

4.1 PROTOTIPAGEM FUNCIONAL

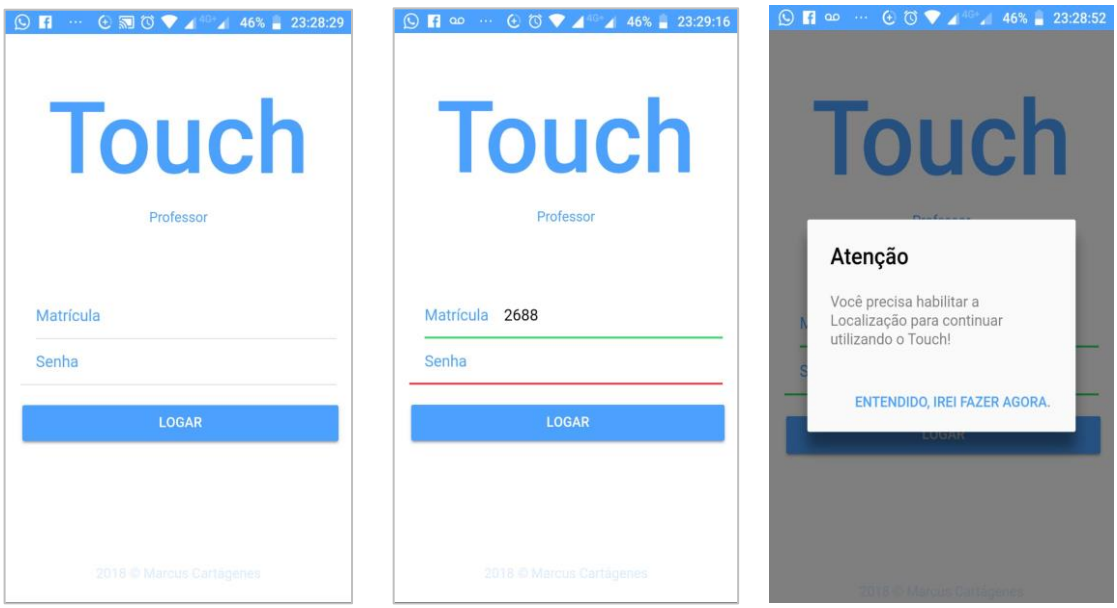
A prototipagem funcional é o processo de construção da plataforma tendo como base os requisitos funcionais, não funcionais e regras de negócios, seguindo fielmente os diagramas construídos. Como foi dito nos capítulos anteriores, a plataforma *Touch* conta com dois aplicativos mobile, uma *Rest* API hospedada em um VPS, e uma base de dados na nuvem. Pelo fato de a base de dados ser orientada a documentos, não se fez necessária a criação de um diagrama E-R (Entidade Relacional).

No total, foram construídas 7 telas para o módulo do Professor e 5 telas para o módulo do atendente, por conta da conexão WebSocket aberta com o *Firebase*, os componentes nestas telas são atualizados dinamicamente sem a necessidade de recarregar a página. As telas foram construídas com auxílio do *Ionic Framework* e os diagramas construídos com a ferramenta *LucidChart*.

4.1.1 Módulo Professor

No módulo do Professor, são apresentadas as principais telas dos cenários *login* e de localização. Na Figura 5, ao lado esquerdo está a representação o estado padrão, quando o formulário ainda não foi submetido e os campos ainda não foram preenchidos e, o lado direito a representação do cenário onde houve algum erro de validação dos campos necessários. É possível verificar que em do destaque da linha na cor vermelha que o erro de validação foi na senha, nota-se que o a matrícula (linha verde foi estava cadastrada e foi validada). Para o usuário efetuar o *login* é necessário estar com a “Localização do dispositivo” habilitada, Figura 5.

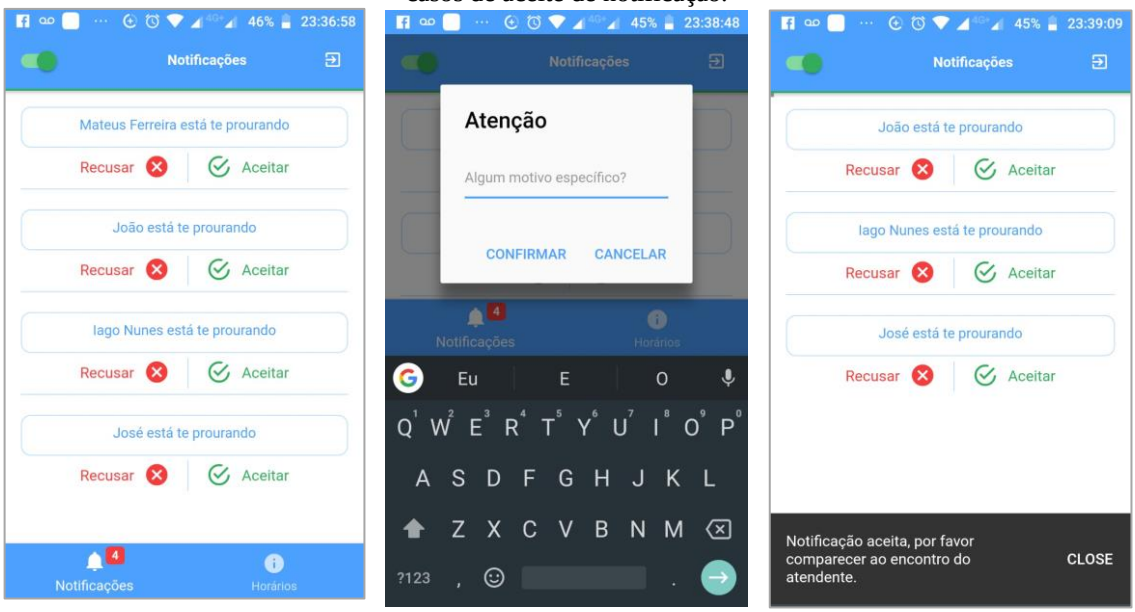
Figura 5. Tela de login no estado padrão à esquerda e tela de login quando ocorre validação do formulário à direita. Cenário em que a localização do dispositivo está desativada.



Fonte: do autor

Após a realização do *login* do usuário, ele deverá ser redirecionado para a tela de "Notificações". A tela de notificações é onde o professor poderá interagir com o atendente, através desta tela é possível responder a uma notificação (negativa ou positivamente) e alterar a disponibilidade.

Figura 6. Página de notificações. Formulário para rejeição de notificação a esquerda e a direita para casos de aceite de notificação.

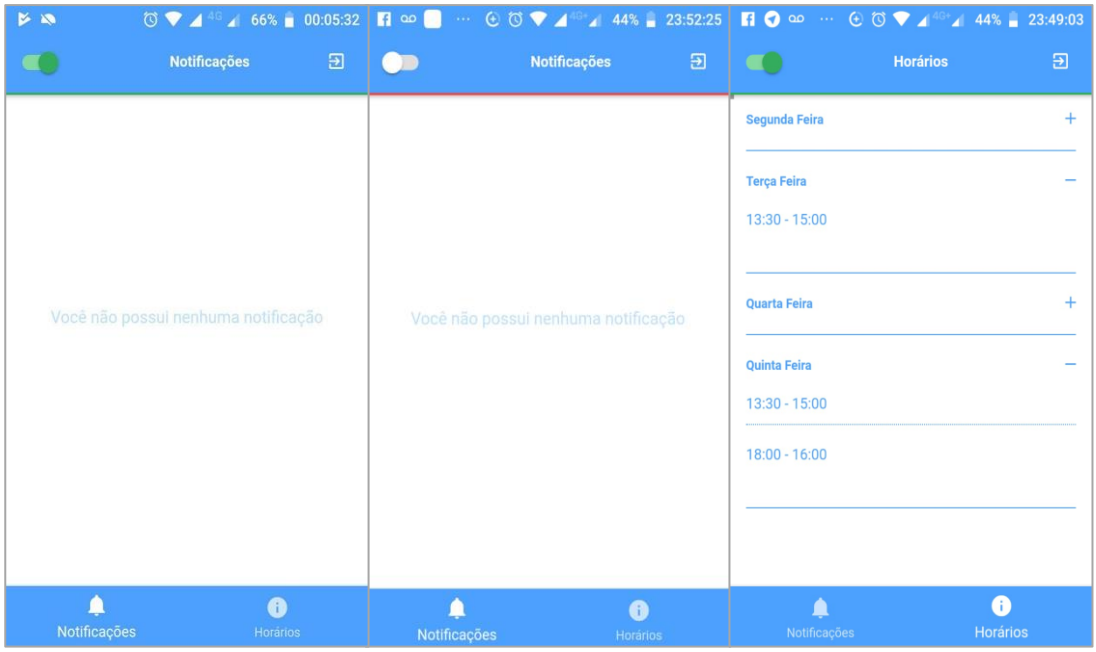


Fonte: do autor

Caso o professor responda a notificação negativamente, será exibido um formulário onde pode ser enviada uma justificativa para complementar a resposta que o atendente passará para o aluno, logo após a confirmação, a notificação é excluída automaticamente da

fila do professor, conforme demonstrado na Figura 6 ao lado esquerdo; com relação a uma resposta positiva.

Figura 7. Cenário onde o professor está na sala dos professores e disponível para notificação a esquerda, a direita, cenário onde o professor não está na sala dos professores. Lista de horários extraclasse do professor.



Fonte: do Autor

O *status* do professor pode ser controlado pelo mesmo através do *Toggle Button* no cabeçalho, este botão possui dois estados: Ativo (Verde) e Inativo (Cinza). Caso o *Toggle* esteja cinza, significa que o professor em questão optou por não receber notificações, caso esteja verde, o professor está apto a receber notificações no seu dispositivo. No que se refere a barra horizontal abaixo do *header*, esta pode variar entre as cores verde e vermelho caso o professor esteja na instituição e fora desta respectivamente (Figura 7). Por meio da aba "Horários", o professor terá acesso aos seus horários extraclasse para atendimento, esta informação será oferecida por uma API da própria Universidade, possuindo apenas característica de ser somente leitura.

4.1.2 Módulo Atendente

Assim como o módulo do Professor, o usuário Atendente também precisa ser autenticado. O fluxo de *login* do atendente se inicia com a validação do formulário, caso o formulário submetido esteja válido, a plataforma verificará no módulo de autenticação do *Firebase* se as credenciais informadas estão devidamente cadastradas, caso esteja, o usuário será redirecionado para a página de professores; caso as credenciais não existam, a plataforma informará que não existe nenhum usuário vinculado a aquelas credenciais e, se o formulário de submissão estiver inválido, a plataforma mostrará uma mensagem sobre o evento. Após a efetividade da autenticação do atendente, o mesmo será redirecionado para a página de "Professores", que, na sua inicialização será responsável por registrar um *WebSocket* com o *Firebase* para listar todos os professores, garantido assim a atualização do *status* dos professores em tempo real no aplicativo.

Uma das funcionalidades que esta página oferece é a listagem dos professores que pode assumir três estados:

- Professor disponível: Ocorre quando o professor está na instituição e está disponível para receber notificações, conforme a Figura 8.
- Professor indisponível: Ocorre quando o professor está na instituição e optou por não receber notificações (símbolo do sino desabilitado), conforme a Figura 9.
- Professor ausente: Ocorre quando o professor não se encontra na instituição (símbolo da casa vermelha), conforme a Figura 10.

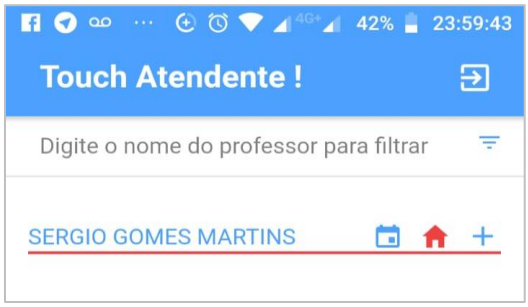
Figura 8. Cenário onde o professor se encontra na instituição e optou por receber notificações.

Fonte: Do Autor.

Figura 9. Cenário onde o professor está na instituição, porém optou por não receber notificações.

Fonte: Do Autor.

Figura 10. Cenário onde o professor não está presente na instituição.



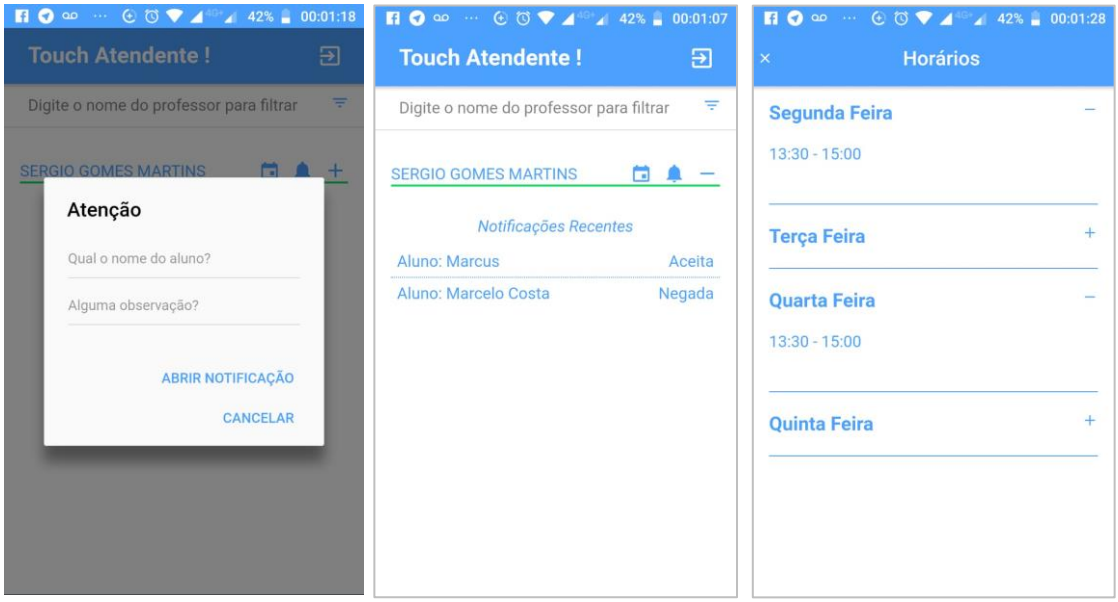
Fonte: Do Autor.

Para que uma notificação possa ser enviada a um professor específico, é imperativo que o mesmo esteja na instituição e tenha optado por receber notificações, como explicado no RF (Requisito Funcional) 01 (Notificar Professor). O processo de abertura de notificação começa com o preenchimento do formulário que contêm dois campos: Nome do aluno

requerente (obrigatório) e Descrição (opcional), após o preenchimento das informações, a plataforma realizará uma validação nos campos digitados, caso a validação seja sucesso, é feita uma requisição HTTP PUT para a API do Touch enviando os dados do formulário, a API será responsável por configurar uma nova notificação (Inserir data de abertura, validade, status inicial, uid...) e enviar esta notificação para um nó na árvore de notificações do *Firebase*.

O atende disporá de um modal para parametrizar a notificação que deseja enviar, será obrigatório informar o nome do aluno e opcional alguma informação para complementar a notificação, como mostra a Figura 11. Com o objetivo de complementar as informações sobre o professor, o atendente terá acesso a lista de horários extraclasse e as notificações recentes do professor; estas informações são cacheadas no dispositivo tendo em vista economizar a banda do dispositivo do usuário, como mostrado na Figura 11.

Figura 11. Modal para abertura de notificação. Notificações recentes abertas para o professor e o *status* de cada uma na esquerda, e na direita, a lista de horários extraclasse.



Fonte: Do Autor.

4.2 RESULTADOS DA AVALIAÇÃO

Para medir a efetividade das funcionalidades oferecidas pela plataforma *Touch*, foi realizado uma bateria de testes entre 20 e 26 de outubro de 2019 com 1 professor e 4 atendentes que são responsáveis por localizar professores dentro da sala dos professores. Esta bateria de testes teve como ator central o atendente, e possuiu o objetivo de comprovar que com as informações promovidas pela plataforma são o suficiente para dar um *feedback* mais rápido e preciso sobre a localização de uma pessoa (representado pelo professor) em um ambiente com alto fluxo de pessoas. Foram aplicados 3 cenários diferentes, como descrito abaixo:

1. Professor dentro da sala dos professores e disponível para receber notificações;

2. Professor dentro da sala dos professores e indisponível para receber notificações;

3. Professor fora da sala dos professores.

Segundo os atendentes que participaram, estes são os cenários mais comuns do seu contexto de trabalho. Levando em consideração sobre a veracidade trazida por meio da Plataforma *Touch* sobre o status do professor, todos os atendentes afirmaram confiar nos dados provenientes da mesma. Com relação a eficiência do aplicativo em dar um feedback aos alunos de forma mais rápida e precisa, todos os atendentes continuaram afirmando a efetividade da plataforma levando em consideração as informações providas pela mesma.

Levando em consideração a simplicidade do processo de abertura de notificação ao professor, novamente todos os atendentes demonstraram achar tal processo simples. Finalmente quando perguntados se a a plataforma ajudaria no processo de localização e notificação de um professor, todos os atendentes responderam que sim. Quanto ao quesito usabilidade da plataforma, quando perguntados sobre o conforto de operacionalizar o aplicativo esta foi avaliada pelos atendentes, notas de 0 para péssimo e nota 10 para excelente. Verificou-se que 50% doas atendentes deram nota 7 e que os outros 50% deram nota 8, e obteve uma média de 7.5.

Apesar dos resultados obtidos pelo uso do atendente serem satisfatórios, o professor em questão levantou alguns problemas capazes de impactar o uso da plataforma como um todo. O fato de que a plataforma utiliza unicamente o celular para o referenciamento geográfico traz a possibilidade de o professor não estar atento as notificações ou esquecer o celular. Por ser uma solução aplicada em um ambiente cujo a faixa etária é bastante variada, pode haver uma resistência de uso quando levamos em consideração a cultura organizacional e a forma de trabalhar dos professores. Tais aspectos podem acabar impactando negativamente na veracidade das informações repassadas pelos atendentes.

5 CONCLUSÃO

A construção da Plataforma *Touch* para auxiliar alunos e atendentes no processo de contatar professores dentro de um ambiente com alto fluxo de pessoas demonstrou grande importância. A partir da pesquisa pela opinião de alunos e funcionários foi possível detectar fatores primordiais no que se refere a possíveis melhorias que a mesma poderá acarretar, caso integrada, como a diminuição do tempo em que alunos comumente costumam esperar para terem acesso a informação se o professor se encontra ou não dentro de sua sala ou mesmo na instituição e a demora do atendente no processo de procura pelo professor, por exemplo.

Ainda que possua algumas limitações relacionadas ao uso humano (não visualizar notificação ou esquecer o celular em outro local) e implementação técnica (necessidade de um celular *Android* com versão mínima de 6.3 e IOS 9 ou mesmo a falta de carga), o *Touch* se mostrou eficaz quando requisitado, promovendo atualizações em tempo real quanto ao estado do professor, se disponível ou não para atender alunos fora de classe, além de promover maior conforto dos atendentes pelo fato de não terem que se locomover na sala dos professores sempre que um aluno requisita um professor. Demonstrando, desta forma, a importância da implementação de tecnologias para auxiliar tarefas manuais, possibilitando

assim uma melhor organização e aproveitamento da mão de obra humana, tornando-a mais produtiva.

REFERÊNCIAS

Adriano, T. S. (2018). Introdução ao Firebase. Acesso em 01 de 10 de 2018, disponível em Medium: https://medium.com/@programadriano/introdu%C3%A7%C3%A3o-ao-firebase-bd59bfd03f29

Aladdin, M. (2018). Software Architecture - The Difference Between Architecture and Design. Acesso em 04 de 10 de 2018, disponível em Codeburst: https://codeburst.io/software-architecture-the-difference-between-architecture-and-design-7936abdd5830

Barry, D. K. (2012). Service-Oriented Architecture (SOA) Definition. Acesso em 20 de 09 de 2018, disponível em Service Architecture: https://www.service-architecture.com/articles/web-services/service-oriented_architecture_soa_definition.html

Bartholo, V. F., Amaral, M. A., & Cagnin, M. I. (2009). Uma Contribuição para a Adaptabilidade de Ambientes Virtuais de Aprendizagem para Dispositivos Móveis. Revista Brasileira de Informática na Educação, 17(2).

Carvalho, A. A. (2015). Apps para dispositivos móveis: manual para professores, formadores e bibliotecários.

Carvalho, L. (2017). O potencial exploratório da Geolocalização em games. Periodicos UFP.

Copes, F. (2018). The HTTP Protocol. Acesso em 04 de 10 de 2018, disponível em Flavio Copes: https://flaviocopes.com/http/

Cordeiro, F. (2017). Android SDK: O que é? Para que Serve? Como Usar? Acesso em 11 de 10 de 2018, disponível em AndroidPro: https://www.androidpro.com.br/blog/android-studio/android-sdk/

CronAPP. (2018). Compreenda as diferenças entre o desenvolvimento nativo e híbrido . Acesso em 11 de 10 de 2018, disponível em CronApp: https://www.cronapp.io/compreenda-as-diferencas-entre-o-desenvolvimento-nativo-e-hibrido/

Cruz, K. C. (2013). Estudo sobre o Near Field Communications e seu papel em pagamentos via dispositivos móveis.

Diedrich, C. (2015). O que é docker? Acesso em 08 de 10 de 2018, disponível em Mundo Docker: https://www.mundodocker.com.br/o-que-e-docker/

Ferreira, T. A., Moreira, R. C., & Mozzaquatro, P. M. (2011). Um estudo sobre computação sensível ao contexto baseado em geolocalização.

Fielding, R. T. (2000). Representational State Transfer (REST) . Acesso em 07 de 10 de 2018, disponível em ics: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm

Furtado, C. C., Pereira, V., Azevedo, L., Baião, F., & Santoro., F. (2009). Arquitetura Orientada a Serviço. Relatórios Técnicos do Departamento de Informática Aplicada da UNIRIO.

Furtado, C., Pereira, V., Azevedo, L., Baiao, F., & Santoro, F. (2009). Arquitetura Orientada a Serviço - Conceituação. Rio de Janeiro: Universidade Federal do Estado do Rio de Janeiro.

Ionesco, D. (2018). Acesso em 13 de setembro de 2018, disponível em The Geolocation Revolution: https://www.entrepreneur.com/article/205796

Lauden, T. (2018). Basic of golang for beginners. Acesso em 11 de 10 de 2018, disponível em hackernoon: https://hackernoon.com/basics-of-golang-for-beginners-6bd9b40d79ae

Lemes, H. f., & dias, J. w. (2014). Aplicações baseadas em geolocalização.

Madureira, D. (2017). Aplicativo nativo, web App ou aplicativo híbrido? Acesso em 08 de 10 de 2018, disponível em usermobile: https://usemobile.com.br/aplicativo-nativo-web-hibrido/

Mantovani, C., & Dantas, G. (2010). Os fluxos informacionais nos dispositivos móveis. Cultura Informacional e Digital.

Mosharraf, F., & Forouzan, B. A. (2012). Computer Networks: A Top-Down Approach (1 ed.). New York: Bookman.

MuleSoft. (2016). What is a REST Api? Acesso em 16 de 10 de 2018, disponível em MuleSoft: https://www.mulesoft.com/resources/api/what-is-rest-api-design

Oliveira, L. R., & Medina, R. D. (2007). Desenvolvimento de objetos de aprendizagem para dispositivos móveis: uma nova abordagem que contribui para a educação. Novas Tecnologias na Educação, 5(1).

Oliveira, F. (2017). Smartphones estão nas mãos de 62% dos brasileiros, diz Google. Acesso em 02 de 10 de 2018, disponível em Folha de S.Paulo: https://www1.folha.uol.com.br/tec/2017/02/1862362-smartphones-estao-nas-maos-de-62-dos-brasileiros-diz-google.shtml

Pillou, J. F. (2008). The HTTP protocol. Acesso em 3 de 10 de 2018, disponível em CCM: https://ccm.net/contents/273-the-http-protocol

Souza, E. F. (2018). GoLang — Simplificando a complexidade . Acesso em 28 de 09 de 2018, disponível em Medium: https://medium.com/trainingcenter/golang-d94e16d4b383

Thiengo, V. (07 de Julho de 2017). MVP Android. Acesso em 10 de 11 de 2018, disponível em Thiengo Calopsita: https://www.thiengo.com.br/mvp-android.