

NossaVan: Proposta e Implementação de um Aplicativo para Usuários de Transportes Escolares

Ruann Batista Reis¹, Vinícius Rodrigues Duarte¹, Lesandro Ponciano¹

¹PUC Minas em Contagem
Bacharelado em Sistemas de Informação

ruannb.reis@hotmail.com, vinird1993@gmail.com.br, lesandrop@pucminas.br

Resumo. *A urbanização cresce e traz consigo inúmeros problemas sociais que afetam diretamente as pessoas que convivem nas cidades. Todos os cidadãos possuem direito à educação, porém a logística para ir e vir às instituições de ensino nem sempre são adequadas ou de acordo com a preferência do usuário. São oferecidos diversos serviços de transporte escolar, entretanto, em poucos deles o acesso do estudante ao meios dos transporte ocorre via um sistema computacional. Visando reduzir essa dificuldade, propõem-se e implementa-se um aplicativo mobile para a plataforma Android que faça a interação entre motorista de transporte escolar e o estudante. O aplicativo possui cadastro de perfil para motorista e também para aluno, possibilitando o aluno procurar serviços escolares que atendem a instituição e turno em que ele estuda, bem como, endereço de sua moradia. O motorista consegue adicionar e remover alunos em seu respectivo veículo cadastrado tendo total gerência de seus clientes. O aplicativo também possui recursos de cálculo e acompanhamento rotas e funcionalidade do usuário notificar sua ausência na utilização do serviço em determinado dia.*

1. Introdução

A urbanização é um fenômeno ainda em progresso em todo o mundo [Cardoso et al. 2011] este progresso trouxe enormes problemas a serem solucionados, como congestionamentos, consumo de energia e poluição. Felizmente, o avanço da tecnologia tem contribuído para mitigá-los. A computação urbana surge para abordar essas questões usando os dados que são gerados nas cidades (por exemplo, fluxo de tráfego, mobilidade humana e dados geográficos) [Zheng et al. 2014]. Dentro do conceito da computação urbana existem vários serviços tecnológicos que visam melhorar a mobilidade urbana, que é um atributo relacionado aos deslocamentos realizados por indivíduos nas suas atividades de estudo, trabalho, lazer e outras [Magagnin and Silva 2008]. No contexto de melhorar a mobilidade urbana, podemos citar como exemplo o aplicativo Uber ¹, que foi criado com o objetivo de facilitar e inovar a forma pela qual as pessoas se locomovem nas cidades. Dentro deste contexto, podemos aprofundar nas necessidades relacionadas ao transporte de estudantes.

Existem profissionais no ramo de transporte escolar, que oferecem seus serviços a pessoas em diversos bairros e, conseqüentemente, várias instituições de ensino, públicas e privadas. Em paralelo, o número de alunos cresce a cada ano, segundo o Censo

¹Disponível em www.uber.com, último acesso em 11 de novembro de 2018.

da Educação Superior Brasileira [MEC and INEP 2017], divulgado pelo Ministério da Educação (MEC), o ensino superior teve 8,05 milhões de alunos matriculados no ano de 2016, sem considerar alunos matriculados na educação básica. Muitas vezes o meio de transporte é escolhido e custeado pelos estudantes. Há diversas empresas que oferecem tal serviço de transporte. Neste cenário, ainda são encontradas pelo estudantes dificuldades para escolher o serviço que atenda o trajeto de sua residência até a escola a qual está matriculado. Neste trabalho, identifica-se pelo fornecedor do serviço a dificuldade para encontrar estudantes que maximizem a utilização do veículo de transporte. Assim, é possível enunciar o **problema** tratado neste trabalho como: Como um aplicativo pode ajudar estudantes a encontrarem as melhores opções de transporte escolar, considerando fatores como, ponto de partida, escola de destino, valor do serviço prestado, qualidade e segurança do veículo?

O **objetivo principal** deste trabalho é propor, implementar e avaliar um aplicativo para usuários de transportes escolares. O aplicativo visa ser um intermediador entre os proprietários de transportes escolares, tais como vans e ônibus, e alunos ou pais de alunos que estão em busca deste tipo de serviço. Os **objetivos específicos** são: i) investigar as necessidades de estudantes de interação de estudantes e motoristas; e, ii) propor estratégias de interação que se adequem ao contexto e público alvo da aplicativo proposto.

O aplicativo foi desenvolvido para *smartphones* que possuem o sistema operacional Android, devido ao fato do uso desse tipo de dispositivo ter aumentado dramaticamente nos últimos anos [Gandhewar and Sheikh 2010], considerado a plataforma uma das mais comuns no Brasil [ComTech 2018]. Denominado como **NossaVan**, o aplicativo possibilita aos usuários a criação de seu perfil, sendo categorizado como motorista ou como aluno, onde ambos constam com os respectivos formulários de cadastro. No perfil de motorista, é possível gerenciar os alunos inscritos em sua van, previamente cadastrada, adicionar ou remover alunos de sua van, e pode visualizar o trajeto otimizado para as residências dos alunos. No perfil de aluno, é possível encontrar os prestadores de serviços, podendo escolher um para requisitar inscrição. O aluno pode informar ao motorista sobre sua utilização ou não do serviço naquele dia, além de conseguir acompanhar a localização do motorista, para acompanhar o trajeto ou verificar se o prestador de serviços está próximo de sua localização.

2. Referencial Teórico

Nesta seção é descrito o referencial teórico que discorre sobre o conceito da computação urbana e a prática de desenvolvimento de aplicativos utilizando *Material Design* da google. Sendo assim, também é citado as metodologias utilizadas para a interação entre os operadores do projeto e o público alvo.

2.1. Computação Urbana

O trabalho proposto está envolvido com questões como mobilidade urbana, bem como transporte público, se enquadrando então na computação urbana que, segundo Zheng et al (2014), é um campo interdisciplinar no qual as ciências da computação encontram campos relacionados à cidade, como transporte, engenharia civil, meio ambiente, economia, ecologia e sociologia no contexto dos espaços urbanos. A computação urbana auxilia a entender o comportamento humano nas grandes cidades, tornando possível a aquisição,

integração e análise de grandes dados gerados por diversas fontes em espaços urbanos, como sensores, dispositivos, veículos, edifícios e humanos.

A crescente urbanização gera a necessidade de aumento da mobilidade urbana, e ao mesmo tempo, eleva sua complexidade. Segundo Barczak et al [2012], a motorização individual está crescendo cada vez mais, elevando custos sociais, econômicos e ambientais. Como diversos autores apontam, apenas alternativas que incluem o planejamento urbano e de transportes seriam eficazes para reduzir problemas ambientais e aumentar a qualidade e eficiências dos transportes urbanos. Com a computação urbana, é possível mitigar alguns desses problemas, como a logística de transporte, onde através do auxílio da tecnologia é possível traçar a melhor rota para cada indivíduo, reduzindo o tempo gasto no trajeto.

2.2. Material Design

O sistema proposto para o contexto deste trabalho deve ter enfoque nas necessidades dos usuários do produto, buscando usabilidade, acessibilidade e qualidade. Para atender a esses requisitos seguidos neste trabalho, o processo de desenvolvimento utiliza técnicas de material design. O *Material Design*, foi criado pela *Google Design* [Design 2018] e tem como objetivo de atender a usabilidade no intuito de fornecer uma experiência pensada para ser fluída, natural, intuitiva e de simples entendimento, através de diversas particularidades e fundamentos que compõem o *Material Design* [Foundation 2014]. Foram considerados na execução deste trabalho as orientações definidas na documentação do *Material Design*, afim de se obter as melhores práticas de design no trabalho proposto.

Neste trabalho, faz-se a prototipação da interface gráfica aplicando as técnicas definidas pela *Google Design*. De tal forma a buscar que o aplicativo desenvolvido apresente um resultado que fique natural a usabilidade para o público alvo definido.

2.3. Design Centrado no Humano

Dentre as metodologias que consideram o indivíduo em sua abordagem, destaca-se o Design Centrado no Humano (DCH) [Chaves et al. 2013]. O enfoque voltado para o ser humano ocorreu com o início dos anos de 1950 quando os até então produtos produzidos em série e com características funcionalistas, pertencentes à era industrial, passaram a ser considerados bens de consumo, informação e identidade [Chaves et al. 2013]. Os *designs* começaram a abordar os produtos como práticas sociais, preferências e símbolos, tornando o DCH em uma metodologia que aborda a maneira como as pessoa veem e interpretam os objetos.

Neste projeto aborda-se a otimização da experiência do usuário ao utilizar os métodos Ouvir, Criar e Implementar, descritos por [Chaves et al. 2013]. As fases anteriores ao desenvolvimento abordam a discussão de ideias e a interação com usuários, passando por uma criação abstrata antes da concretização da implementação.

3. Trabalhos Relacionados

Nesta seção são apresentados os trabalhos relacionados a mobilidade urbana e transporte escolar. Eles apresentam diversas informações que nortearão a abordagem a ser proposta neste trabalho.

Alves e Costa (2014) propõem a aplicação da metodologia de *Design Thinking*, que tem por finalidade auxiliar processos de inovação e negócios, na elaboração de um projeto de software relacionado a mobilidade urbana [Lima et al. 2014]. O artigo parte do pressuposto que apesar de vários casos de *startups* que estão criando novos produtos de software e negócios nos dias atuais a metodologia de *Design Thinking* ainda é pouco utilizada. Dentro deste contexto o artigo apresenta as fases que compreendem o ciclo de aplicação de *Design Thinking*, bem como eficiência que se é possível ganhar ao aplicar tal metodologia no desenvolvimento de software.

Cardoso et al. (2013) propõem o desenvolvimento de uma aplicativo para dispositivos móveis no intuito de ser um sistema de controle de passageiros para transporte escolar [Cardoso et al. 2016]. Nesse artigo foi utilizado a Interação Humano-Computador (IHC) onde ao se desenvolver sistemas e aplicativos, os mesmos devem levar em consideração os conceitos encontrados em IHC, tais como a disponibilidade, interatividade, usabilidade, experiência dos usuários e acessibilidade.

O aplicativo Uber ² tem como objetivo principal disponibilizar uma tecnologia de transporte sob demanda, e o que torna isso possível é o aplicativo que conecta motoristas parceiros e usuários. Levando em consideração o fator humano gerando valor para o usuário final e gerando emprego para seus parceiros. Em sua página é destacada que além de ajudar as pessoas a se locomoverem, o Uber está sempre trabalhando para trazer o futuro até as pessoas com a tecnologia de carros semiautônomos e com o transporte aéreo urbano, ajudando as pessoas a pedir entrega de comida de forma rápida e econômica, facilitando o acesso à assistência médica, criando novas soluções para o transporte de cargas e possibilitando que as empresas simplifiquem as viagens de colaboradores.

O aplicativo SIU Mobile ³ disponibiliza funcionalidades que permitem rastrear a localização de ônibus públicos. O sistema fornece um serviço de localização, no mapa, das paradas atendidas e o tempo estimado de chegada do transporte público ao ponto de parada desejado.

A proposta do aplicativo NossaVan incorpora conceitos presentes nos trabalhos citados acima, onde o aluno pode encontrar motoristas de transporte escolar, para a contratação dos serviços permitindo que o motorista gerencie os alunos inscritos em sua van. O trajeto de desembarque é exibido em um mapa para o motorista, com a rota otimizada para o menor tempo e distância de percurso. Os alunos podem rastrear a van na qual está inscrito, através de um ícone atualizado em tempo real em um mapa.

4. Análise da Situação Atual

Nesta seção é apresentada a situação atual do problema identificado neste trabalho e as propostas para a resolução deste problema aplicando Engenharia de Software.

4.1. Contexto Atual

A dificuldade das pessoas encontrarem um transporte escolar que atenda as necessidades como horários, preços, rotas, dentre outros motivos é evidente. É bastante comum que, para encontrar um serviço deste tipo, as pessoas interessadas precisam procurar nas escolas algum catálogo que possua dados dos profissionais que oferecem este tipo de serviço

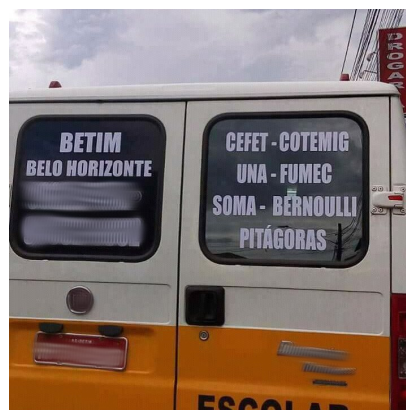
²Disponível em www.uber.com, último acesso em 11 de novembro de 2018.

³Disponível em <http://www.siumobile.com.br>, último acesso em 11 de novembro de 2018.

ou mesmo permanecer na entrada da instituição nos finais das aulas para procurar vans ou ônibus escolares e até mesmo fazer perguntas em redes sociais, como exibido na Figura 1 a fim de encontrar uma solução em suas buscas. A Figura 1 mostra exemplos de como é feita a divulgação de serviços de transporte escolar atualmente. O trabalho proposto visa contribuir de tal forma a vir resolver este tipo de problema, interligando o usuário e o prestador de serviço por meio de um aplicativo.



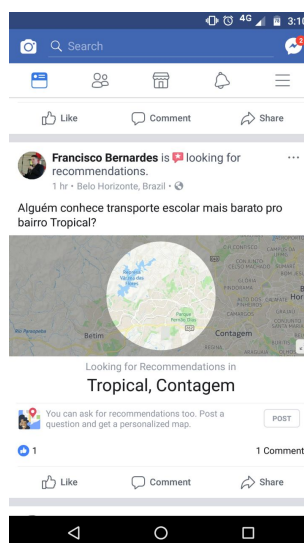
(a) Adesivos colados ao veículo para divulgação do serviço



(b) Adesivos colados ao veículo para divulgação do serviço



(c) Divulgação através de cartões



(d) Busca em redes sociais

Figura 1. Divulgação e procura por serviços de transporte escolar (Imagens de autoria própria)

4.2. Público Alvo

O aplicativo tem o objetivo de solucionar um problema que afeta determinada parcela da população, onde características geográficas e financeiras podem influenciar a necessidade de sanar tal dificuldade. O público alvo se baseia em homens ou mulheres que estão cursando o ensino superior, não possuem veículos para se locomover, pretendem obter um método mais efetivo para sua logística cotidiana e possuem *smartphone*.

Com o objetivo de ajudar a planejar os requisitos do projeto, em seus aspectos visuais ou operacionais, através de um questionário informal e análises do cotidiano de companheiros universitários, identificou-se as personas de usuários alvos do aplicativo, representadas na Figura 2 e na Figura 3.

Pedro Silva


	Idade	25	Descrição Apesar de morar perto da faculdade onde estuda, percorre um longo caminho, diariamente, utilizando o transporte público. Por ser jovem, valoriza bastante o tempo e adoraria possuir um método mais rápido e, possivelmente, mais barato, para se locomover da universidade até sua residência. Vive conectado à internet, gosta de praticidade e, sempre que pode, resolve suas tarefas cotidianas através do seu smartphone.
	Formação	Universitário	
	Residência	Contagem	
	Local de estudo	Contagem	
	Plataforma	<u>Android</u>	

Figura 2. Pedro Silva (Dados fictícios)

Ana Almeida


	Idade	22	Descrição Cursando o ensino superior durante a noite e trabalha durante o dia, utiliza o seu carro como meio de transporte para ir à faculdade, mas prefere outro método de locomoção, caso seja mais econômico, pois sua residência fica na cidade vizinha, gerando muitos gastos com o automóvel. Possui um smartphone pessoal, no qual utiliza, na maior parte do dia, para interagir com outras pessoas ou executar tarefas.
	Formação	Universitária	
	Residência	Belo horizonte	
	Local de estudo	Contagem	
	Plataforma	<u>Android</u>	

Figura 3. Ana Almeida (Dados fictícios)

5. Metodologia

O desenvolvimento do trabalho da intervenção proposta é prática, de modo a disponibilizar produto, software, funcional. As ferramentas e linguagens que são utilizadas na implementação e operação da aplicação são livres para uso, conforme a seguinte lista:

- *Android Studio*, é o ambiente de desenvolvimento integrado (IDE) oficial oferecido pela *Google*, desenvolvido com base no software *IntelliJ IDEA* da *JetBrains* e projetado especificamente para o desenvolvimento do Android.
- *Extensible Markup Language (XML)* é uma linguagem de marcação que define um conjunto de regras para codificação de documentos em um formato legível por humanos e também por máquina, dentre suas várias aplicações neste trabalho foi utilizada para desenvolver o para o *frontend* do aplicativo.
- *Java* é uma linguagem de programação de computador baseada em classe, orientada a objeto sendo que neste trabalho foi-se utilizado no *frontend* do aplicativo.

- *Firebase*, criada pela Google, que auxilia a conexão com o banco de dados e a atualização de dados em tempo real, além da ferramenta de mensageiria.
- *Cloud Messaging*, ferramenta inclusa no *Firebase* tornando a arquitetura de comunicação por notificação sucinta.

O Projeto compartilhado através do repositório Bitbucket está acessível nesta url: <https://bitbucket.org/ViniRd1993/nossavan/src/master/>

O instalador do App NossaVan está acessível nesta url: <https://drive.google.com/drive/folders/1Y-9-pMg32dca2LszLyJ51did7FslqysV?usp=sharing>

5.1. Implementação da Interface Gráfica Com Material Design

A aplicação segue um conjunto de diretrizes que constituem o pacote de *Material Design*. As diretrizes seguidas estão diretamente relacionada com o visual minimalista, com as cores e formato dos componentes como botões e barras de ferramenta, e, por último, o tamanho das fontes. O objetivo dessa implementação é uma interface com poucos detalhes que proporcione fácil interação com o usuário.

5.2. Implementação do *Backend* e de Arquitetura do Aplicativo para Dispositivos Móveis

A aplicação utiliza-se do padrão *Controller* com os respectivos *Data Access Object* (DAO) e classes controladoras. O padrão foi escolhido para otimizar o desenvolvimento modularizado, necessário devido a constantes mudanças de layout que foram implementadas durante o desenvolvimento e pela adoção do *Material Design*, desacoplando a parte operacional e de interface gráfica da aplicação.

5.3. Infraestrutura Remota

O servidor da aplicação, contendo banco de dados e os serviços (API's), foi configurado na plataforma disponibilizada pelo Google, *Firebase*, ideal para aplicações que operam com transações de dados em tempo real e possuem conectividade com a internet. Essas características são essenciais para a sincronia de informações exibidas para o motorista e para o aluno. O aplicativo utilizará a ferramenta de mensageiria, *Cloud Messaging*, para transmitir notificações entre os usuários, possibilitando uma interação entre eles.

5.4. Implementação da Integração com GoogleMaps

A Google é uma organização possui abrangência mundial e, por isso, seus serviços possuem o mesmo alcance. Esta, por sua vez, fornece dados geográficos através da API para desenvolvedores utilizarem em suas próprias soluções. O serviço utilizado é a Google Maps *Geolocation* API, que permite a exibição de um mapa de alcance global e a localização do usuário, através do Sistema de Posicionamento Global (GPS), do inglês *Global Positioning System*. A aplicação foi registrada e utiliza requisições aos serviços através de uma chave RSA (um algoritmo de criptografia de dados, que deve o seu nome a três professores do Instituto de Tecnologia de Massachusetts (MIT), Ronald Rivest, Adi Shamir e Leonard Adleman) que é gerada durante o cadastro, até uma determinada quantidade de consultas o serviço pode ser utilizado gratuitamente e, caso excedido o limite, haverá cobrança monetária proporcionalmente.

5.5. Cálculo de Rota

Para otimizar o percurso do motorista e sugerir rotas, necessitou o estudo da possibilidade do uso de algoritmos específicos para a aplicação. A funcionalidade de rota pode ser utilizada através de aplicativos terceiros, como o GoogleMaps, exibindo detalhes de tráfego, estimativa de tempo para a chegada ao destino e orientação oral.

6. Projeto e Implementação do Aplicativo NossaVan

Nesta seção é apresentada uma modelagem inicial do aplicativo proposto. A implementação do projeto foi realizado em etapas definidas na metodologia. As tarefas foram orientadas em paralelo e em série, em razão da composição da equipe constar com dois desenvolvedores.

6.1. Diagrama de Caso de Uso

A proposta do projeto inicial deve ser composta por funcionalidades básicas para atender as necessidades de motoristas e estudantes. O Diagrama de Casos de Uso demonstrado na Figura 4 exibe a interação entre esses atores do sistema e o conjunto de casos de uso. O diagrama foi desenvolvido com propósito de modelar as funcionalidades que foram implementadas no aplicativo e que estarão acessíveis aos motoristas e estudantes.

6.2. Interface Gráfica do Aplicativo

O aplicativo realizará a interação com o usuário através de um aplicativo para *smartphone*, onde o usuário realizará as operações que necessita para a utilização do serviço. As funcionalidades propostas para o sistema foram modeladas em telas que mostram a arquitetura da informação no aplicativo. As telas estão representadas nas Figuras 5, 6 e 7. Elas encontram-se organizadas em três classes: telas que são usadas por ambos estudantes e motoristas, telas que são usadas apenas por estudantes e telas que são usadas apenas por motoristas.

6.3. Implementação da Interface Gráfica Com Material Design

Ao desenvolver a aplicação utilizando a plataforma para dispositivos móveis Android, foi utilizado as orientações de *design* criada pela empresa mantenedora deste sistema operacional. A adoção do padrão indicado pela Google orientou a o posicionamento de funcionalidades em sua respectiva *Activity* (nome dado às páginas de aplicativos Android) e, combinado com a utilização de cores claras em *background* gera o efeito visual limpo e minimalista, evitando a sobrecarga de detalhes que poderiam, eventualmente, atrapalhar o uso do software. Alguns exemplos podem ser observados na Figura 8.

Os princípios de cores seguidos foram:

Hierarquia: As cores indicam quais elementos são interativos.

Legibilidade: Textos e componentes que contêm informações são legíveis e de fácil compreensão.

Expressivo: Utilização de cores fortes para reforçar a marca.

Os princípios de ambiente seguidos foram:

Elevação: Superfícies que estão empilhadas exibem um efeito visual de nível superior.

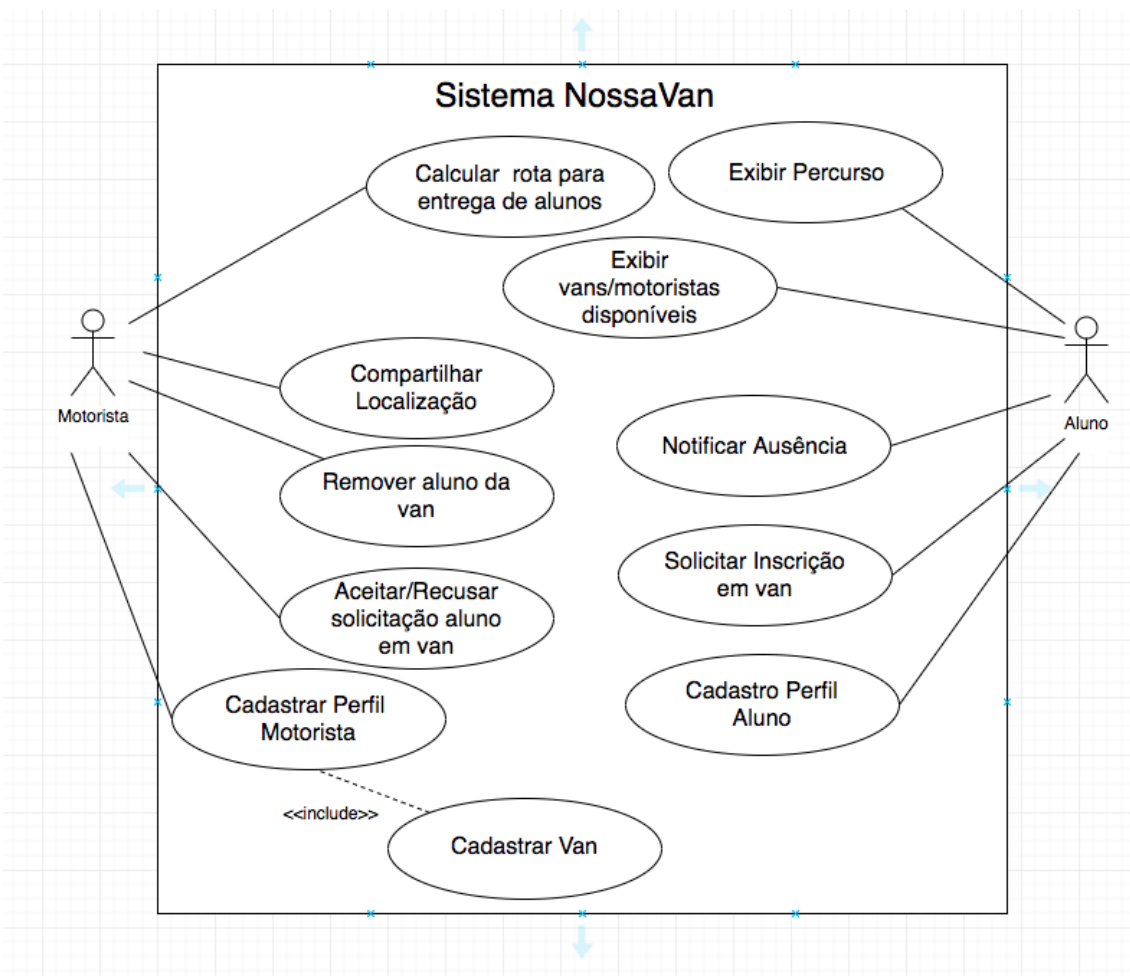


Figura 4. Diagrama de casos de uso referente ao escopo inicial proposto

A aplicação comporta funcionalidades e operações que variam de acordo com o tipo de usuário, sendo "Motorista" ou "Aluno". Apesar das discrepâncias, existem *activities* extremamente parecidas que podem causar dificuldade na identificação, por parte do usuário, de qual tipo de operação que está executando. Para contornar esses problemas, foram utilizados diferentes temas de aplicação para cada tipo de usuário, onde a principal diferença se encontra nas cores dos botões e da *Activity Bar* (barra posicionada na parte superior do aplicativo).

O *login* do tipo aluno possui os seguintes códigos em formato hexadecimal de cores:

- Cor primária: #014893
- Cor primária clara: #4d73c4
- Cor primária escura: #002264
- Cor de realce: #00796B
- Cor de realce escura: #004D40
- Cor de realce clara: #009688

O *login* do tipo motorista possui os seguintes códigos em formato hexadecimal

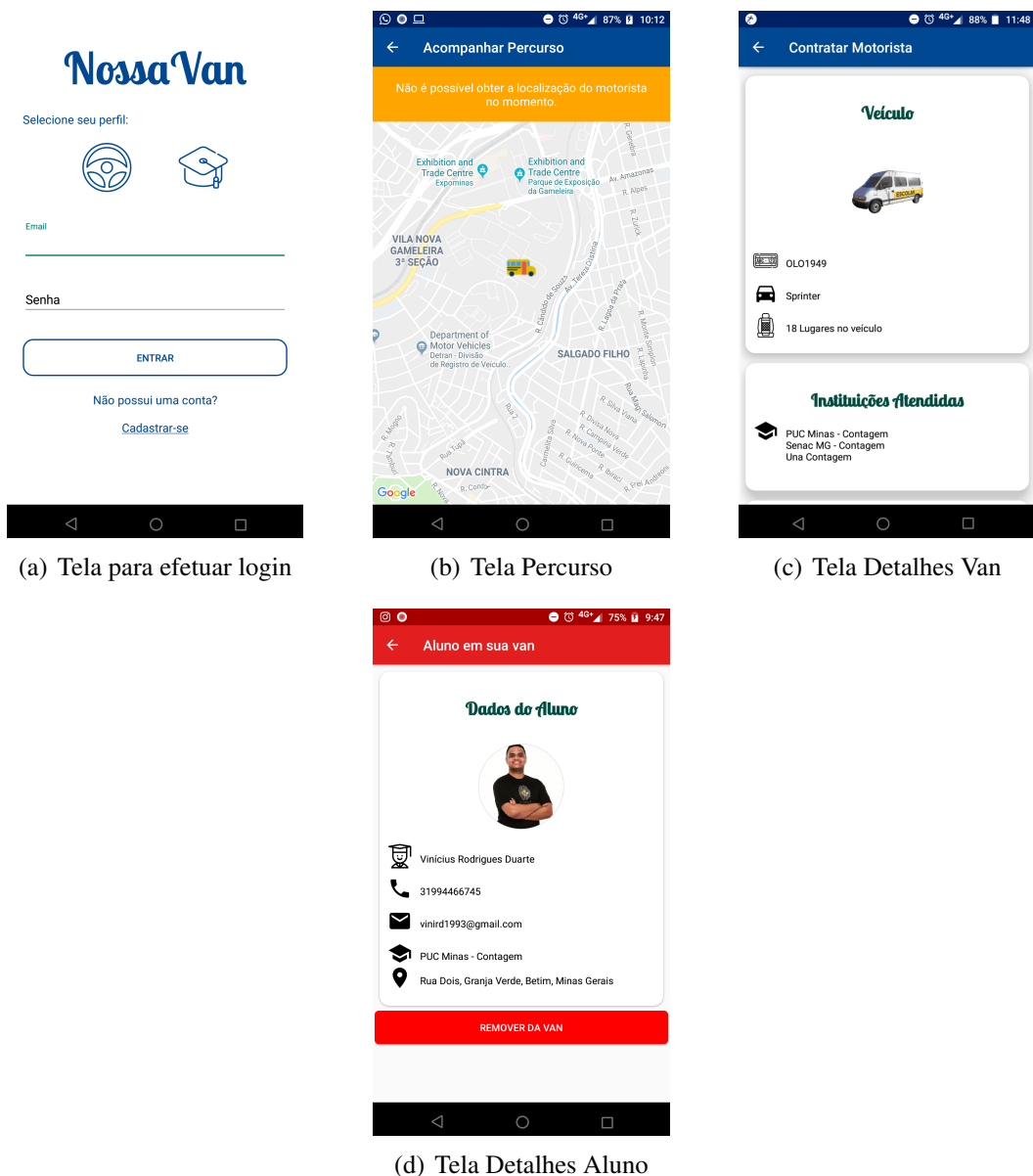


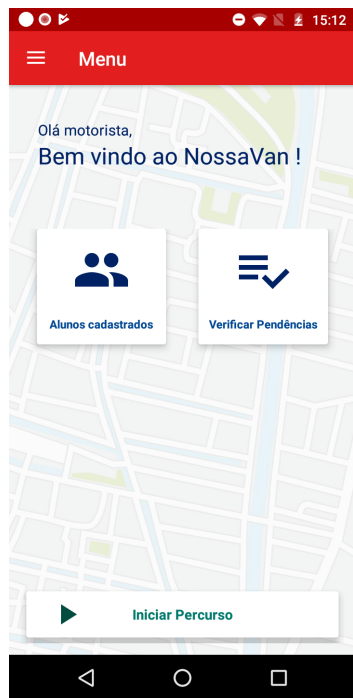
Figura 5. Telas comuns entre motoristas e alunos

de cores:

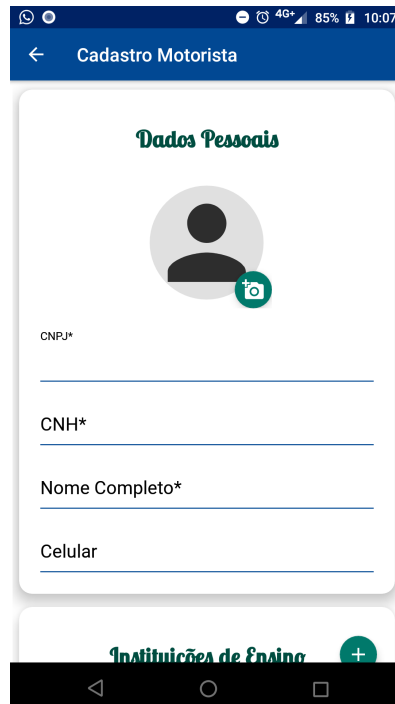
- Cor primária: #e21e1e
- Cor primária clara: #ff5e49
- Cor primária escura: #a70000
- Cor de realce: #ffa726
- Cor de realce escura: #c77800
- Cor de realce clara: #ffd95b

6.4. Arquitetura do aplicativo

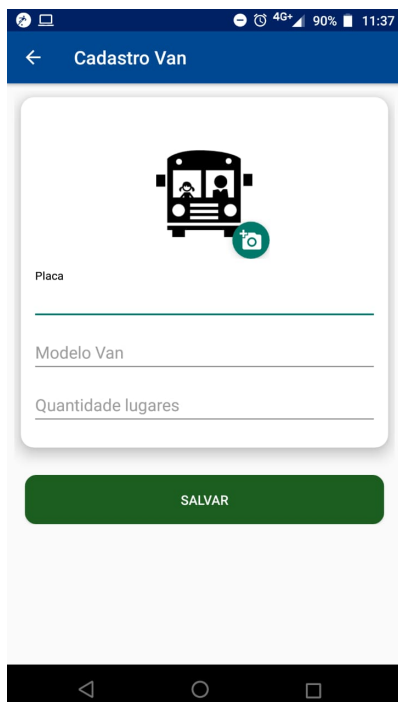
Utilizando dos benefícios da orientação por objeto e tipagem forte, foram implementados padrões controladores adaptados aos recurso nativos da plataforma. As classes do projeto



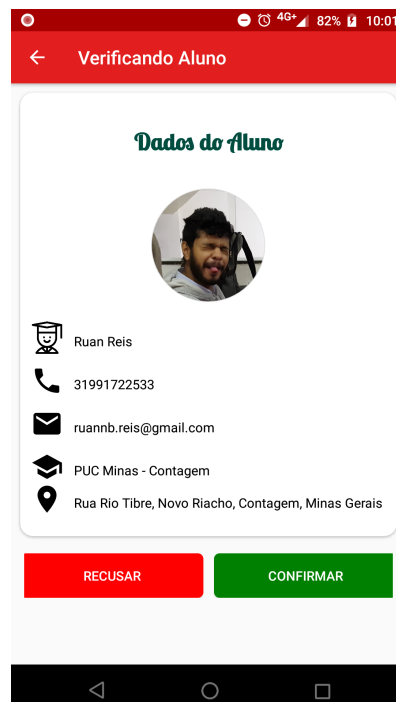
(a) Tela Principal do Motorista



(b) Tela de Cadastro do Motorista



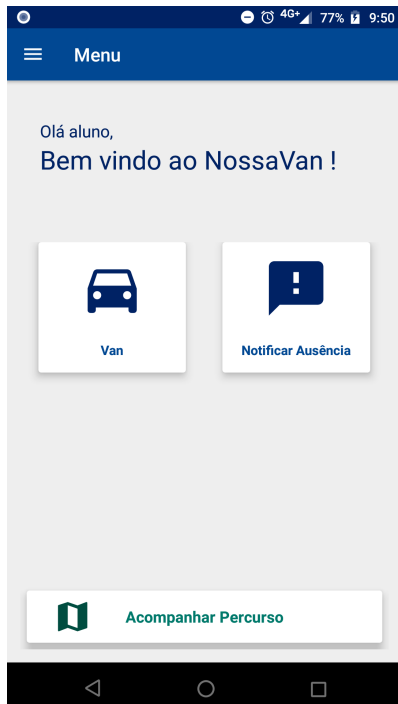
(c) Tela de Cadastro do Veículo



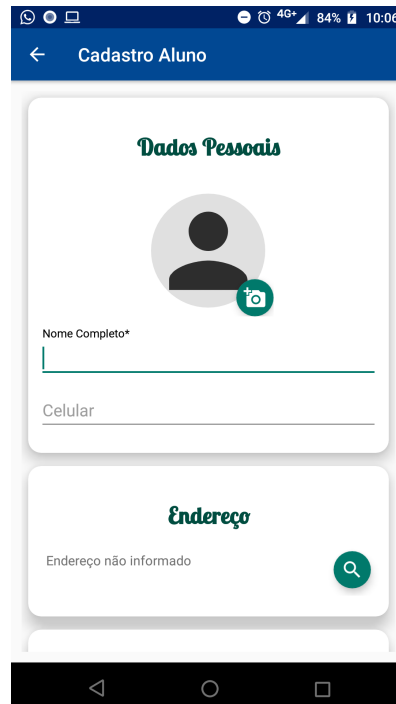
(d) Tela de Adicionar Aluno

Figura 6. Telas específicas para usuários motoristas

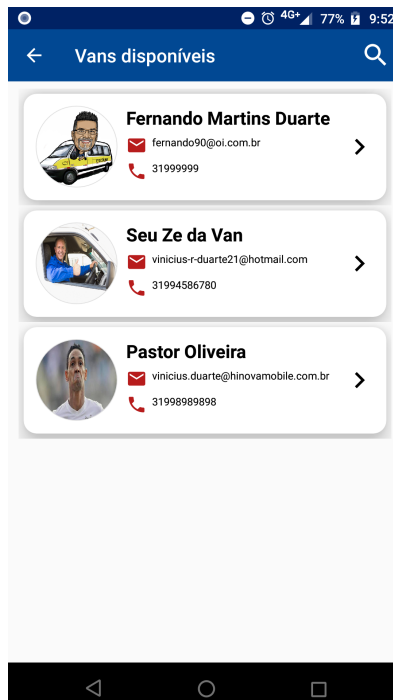
foram divididas em quatro grupos.



(a) Tela Principal do Aluno

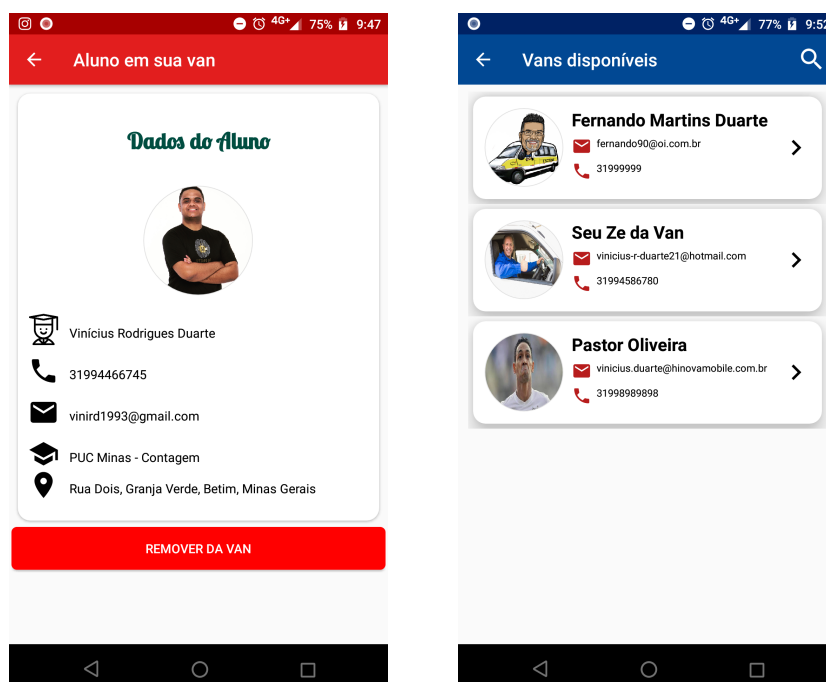


(b) Tela de Cadastro do Aluno



(c) Tela de Procurar por uma Van

Figura 7. Telas específicas para usuários alunos



(a) Mudança de cor do tema da aplicação

(b) Efeito de alto relevo no cartão

Figura 8. Aplicação do Material Design

6.4.1. View

Adaptando o modelo de *view* à arquitetura de programação utilizada no NossaVan, obtemos as classes com comportamento de *activity*. São responsáveis pela execução de tarefas relacionadas a interface gráfica, como a construção de componentes, mudança de cores, preenchimento e validação de campos, além das operações que são executadas na *thread* de UI (*thread* principal do sistema). Conceitualmente deve ser mais simples possível, sem tratamento de lógicas ou regra de negócio.

Classes definidas como *View*:

AcompanharActivity: Interface que exibe a localização atual do motorista, caso o motorista contratado esteja conectado e com o compartilhamento de posição ativo.

AlunosVanActivity: Interface que exibe, para o motorista, os alunos inscritos na van.

CadastroActivity: Exibe para o usuário não cadastrado uma interface para seleção de perfil de cadastro.

CadastroAlunoActivity: Exibe um formulário para o cadastro de um aluno no sistema.

CadastroMotoristaActivity: Exibe um formulário para o cadastro de um aluno no sistema.

DetalhesAlunoActivity: Exibe as informações de cadastro de um determinado aluno.

DetalhesMotoristaActivity: Exibe as informações de cadastro de um determinado motorista.

LoginActivity: Exibe campos para inserção de credenciais e seleção de modo de *login*, para o usuário acessar sua respectiva conta.

MainActivity: Exibe a tela inicial com acesso às principais funcionalidades do aplica-

tivo.

PendenciaMotoristaActivity: Exibe pendências de contatos de alunos.

ListaMotoristaActivity: Exibe os motoristas disponíveis para cadastro.

ProcurarInstituicaoActivity: Exibe as instituições para serem adicionadas aos pontos atendidos, durante o cadastro de motorista.

RotaMotoristaActivity: Exibe a rota otimizada para o desembarque dos alunos.

ProcurarInstituicaoActivity: Exibe as instituições para serem adicionadas aos pontos atendidos, durante o cadastro de motorista.

6.4.2. *Controller*

Responsável pela regra de negócio, tratamento de lógicas operacionais e tratamento de queries efetuadas pela *view*. Na arquitetura do NossaVan, o contexto da aplicação, obtido na *activity* é necessário para a maioria das operações, ao ser criado, toda controladora possui uma referência para o contexto da sua *view* progenitora. Responsável pela execução de requisições na *web* e operações assíncronas.

Classes definidas como *Controller* e que executam operações de *background* relativas às suas respectivas *views*:

AcompanharController, AlunosVanController, CadastroController, CadastroAlunoController, CadastroMotoristaController, DetalhesAlunoController, DetalhesMotoristaController, LoginController, MainController, PendenciaMotoristaController, ListaMotoristaController, ProcurarInstituicaoController, RotaMotoristaController e ProcurarInstituicaoController.

6.4.3. *Model*

São classes representativas dos objetos que são salvos no banco de dados não relacional. Possuem apenas atributos e seus respectivos *getters* e *setters* com o construtor vazio, podendo ser facilmente convertidos em outros modelos de dados como *JavaScript Notation Object* (JSON) ou *Plain Old Java Object* (POJO). São herdeiros de uma classe serializadora, que possibilita o transporte do objeto sem a necessidade da criação de um *Data Transfer Object* (DTO). Classes definidas como *Models*:

Endereco, InstituicaoEnsino, Van, Aluno, AlunoEmVan, Mensagem, Motorista, PendenciaMotorista, Percurso.

6.4.4. *Singleton*

São as classes de utilidade genéricas, utilizadas como ferramentas para facilitar a execução do código. Incluem, também, os *Data Access Objects*, que realizam as operações entre os objetos e o banco de dados. Classes definidas como *Singletons*:

DialogUtil, HttpConnection, Mask, Notification, PathJSONParser, RecyclerItemClickListener, Tools, AsyncWork, BroadcastRegister.

6.4.5. Interação entre classes

O processo de comunicação entre as classes é baseado na Figura 9. A *view* recebe os eventos de interação do usuário, caso seja necessário alguma operação que foge do escopo visual, realiza uma chamada para sua respectiva controladora executar o serviço. Ao receber alguma demanda, a controladora executa o serviço utilizando o contexto recebido durante a criação de sua instância, quando há alteração nos dados persistentes, atualiza o banco de dados local e na nuvem. Ao receber um evento de modificação, o banco de dados dispara uma notificação para todas as *views* que estão utilizando os dados alterados, para consistência entre a visibilidade do usuário e o repositório interno.

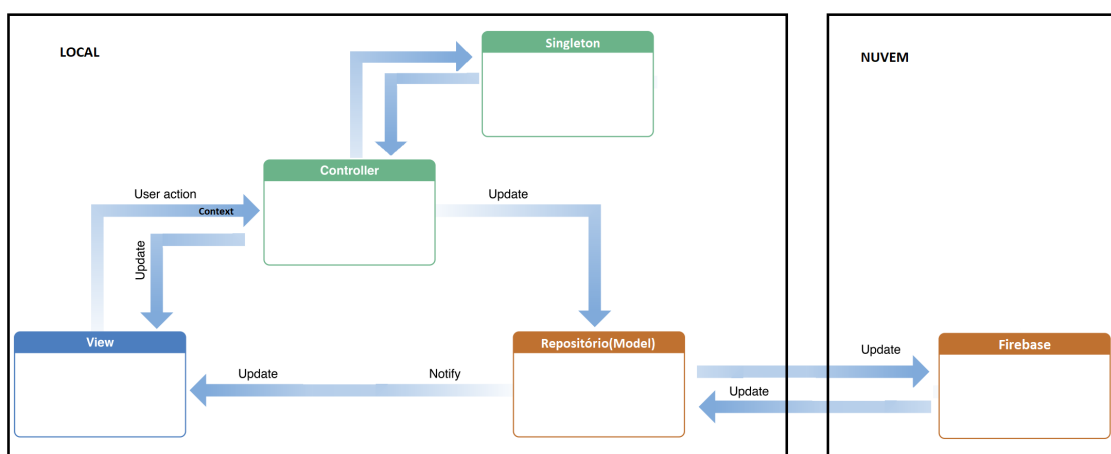


Figura 9. Arquitetura de comunicação entre tipos de classes

6.5. Infraestrutura

Para o desenvolvimento de qualquer aplicação escalável é necessário uma infraestrutura de *backend* que consiste em processar, armazenar e gerenciar os dados de negócios do utilizados pelo aplicativo. Com o objetivo de possibilitar a expansão da solução proposta no presente artigo, uma solução de plataformas cruzadas, tanto *web* quanto *mobile*, foi utilizada uma ferramenta de suporte com arquitetura baseada em *Mobile Backend as a Service* (MBaaS) um *backend* completo com operações para autenticação de usuário, persistência de dados, armazenamento de arquivos, mensagens e lógica de negócios personalizada [Nguyen 2016].

Existem inúmeras ferramentas baseadas na arquitetura MbaaS, para este trabalho foi adotado o *Firebase* que usa a infraestrutura do Google e é redimensionado automaticamente de acordo com a demanda do serviço, tornando possível criar aplicativos sem a necessidade de gerenciamento de recursos de hardware para o servidor. As APIs do *Firebase* estão incluídas em um único *Software Development Kit* (SDK), sendo possível expandir sua aplicação para mais plataformas e linguagens, inclusive Java que é a linguagem adotada na aplicação desenvolvida, tendo o *Firebase* como seu *backend* unificado [Google]. O *Firebase* disponibiliza diversas funcionalidades que vão desde armazenamento em tempo real, o *Realtime Database*, até relatórios com *dashboards* como demonstrado na Figura 10 através de um painel como demonstrado na Figura 11. É possível gerenciar todas as funcionalidades disponíveis, serão citadas funcionalidades utilizadas

no desenvolvimento deste projeto, como *Authentication*, *Cloud Messaging* e *Realtime Database*.

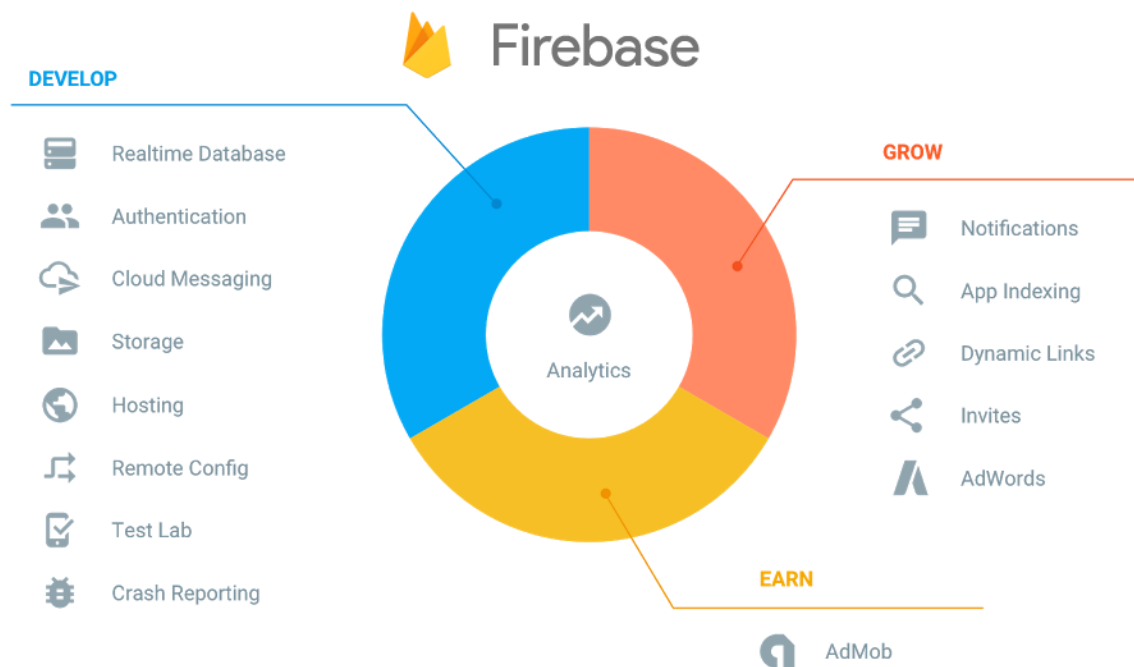


Figura 10. Funcionalidades disponíveis no Firebase

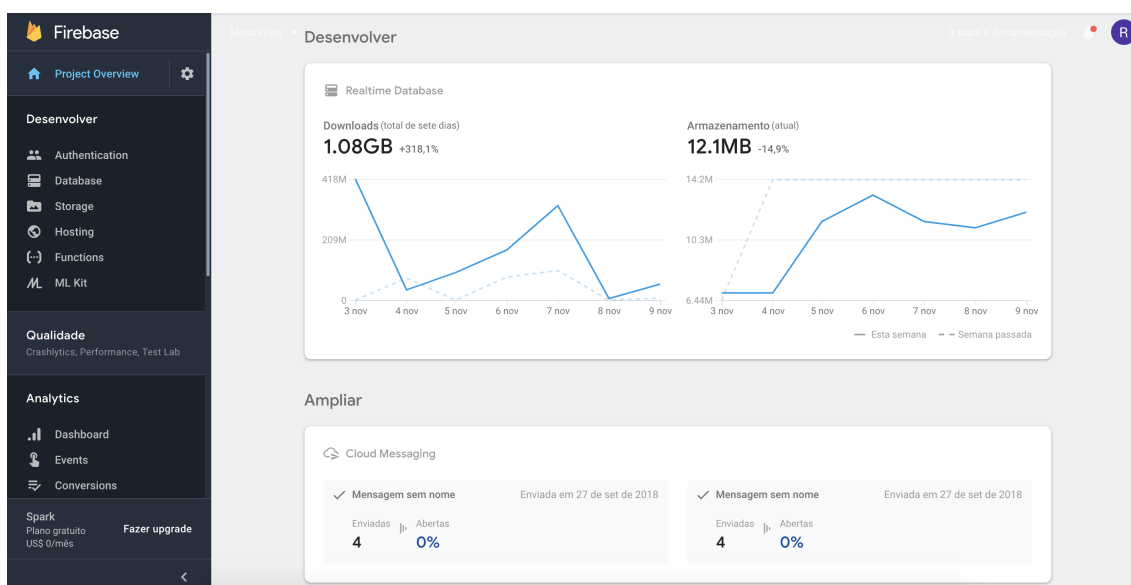
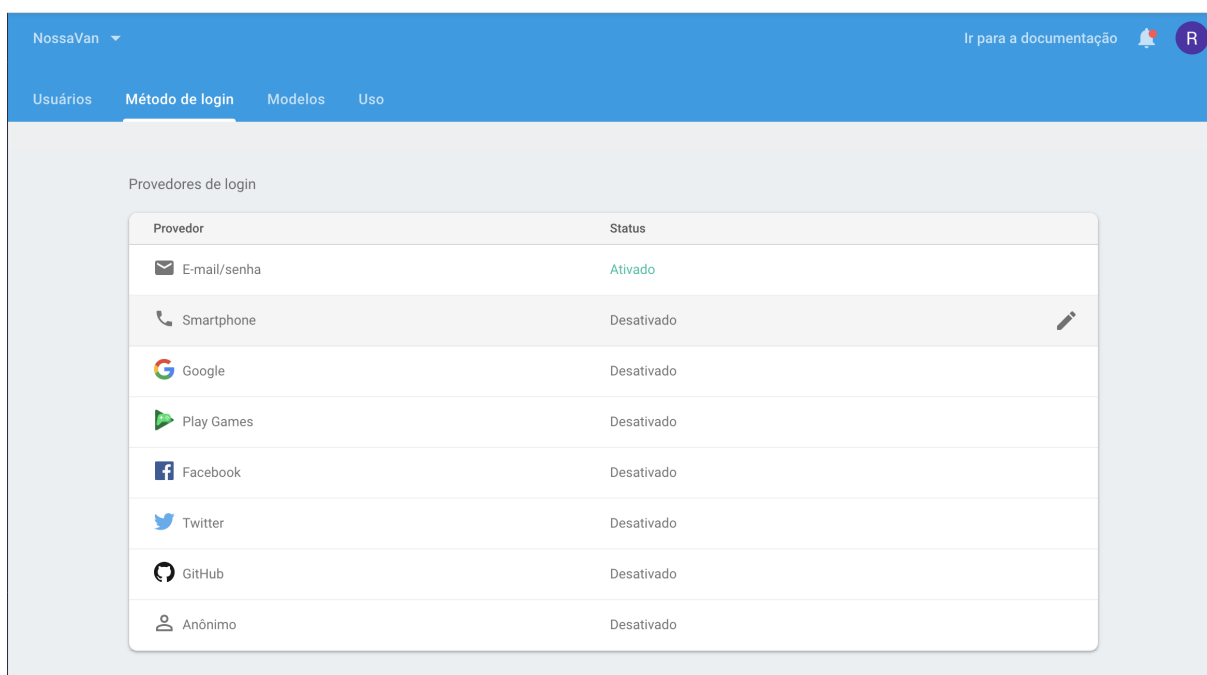


Figura 11. Painel de Configuração do Firebase

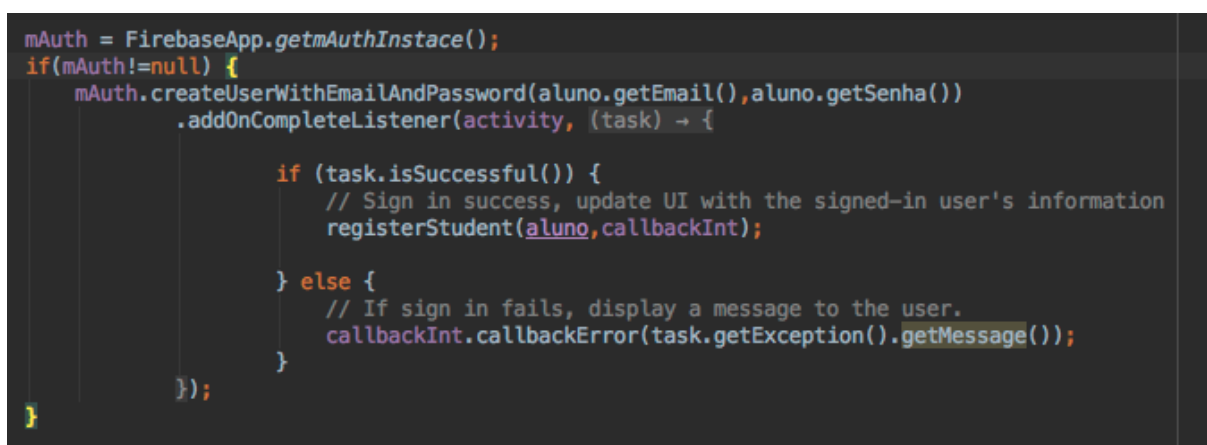
6.5.1. Autenticação

O *Firebase* permite personalização no processo de autenticação de usuários, essa funcionalidade é chamada por *Firebase Authentication*, onde no painel *web* pode se es-

colher quais serão os meios de autenticação disponíveis. Existem vários métodos de autenticação, como por exemplo a autenticação utilizando *Facebook*, *Gmail*, *Twitter* entre outros. No desenvolvimento da aplicação foi configurado no servidor como meio de autenticação o *email* agregado de uma senha, já na aplicação desenvolvida foi necessário importar o SDK denominado como *firebase-auth* e codificado com a linguagem Java, função demonstrada na Figura 12, onde são selecionados quais os artifícios utilizados para autenticar o usuário.



(a) Configuração Autenticação no Firebase



(b) Configuração Autenticação no Aplicativo

Figura 12. Configuração de Autenticação

6.5.2. Realtime DataBase

O *Firestore Realtime Database* é um banco de dados hospedado na nuvem. Os dados são armazenados como *JavaScript Object Notation* (JSON) e sincronizados em tempo real com todos os clientes conectados. Ao contrário de solicitações HTTP típicas, o *Firestore Realtime Database* usa observadores de mudança no banco de dados para gerar eventos de sincronização. Sempre que os dados são alterados, todos os dispositivos conectados recebem a atualização.

O aplicativo permanece responsivo mesmo *off-line*, pois o SDK do *Firestore Realtime Database* mantém seus dados em disco. Quando a conectividade é restabelecida, o dispositivo cliente recebe as alterações perdidas e faz a sincronização com o estado atual do servidor. A segurança e a validação de dados estão disponíveis por meio de regras de segurança baseadas em expressão do *Firestore Realtime Database*, executadas quando os dados são lidos ou gravados. Assim como mostrado na Figura 13 temos um exemplo de como é armazenado, neste caso, os dados da instituição de ensino no formato que o *Firestore Realtime Database* trabalha.



Figura 13. Armazenamento de dados no Firestore Realtime Database

6.5.3. Firebase Cloud Messaging

O *Firestore Cloud Messaging* (FCM) é uma solução de mensagens entre plataformas que permite o envio confiável de notificações sem custo. Usando o FCM, no aplicativo desenvolvido os usuários, independentemente da visão se motorista ou aluno, tem a sua disposição a funcionalidade de notificar a ausência na utilização ou prestação do serviço. Conforme demonstrado na Figura 14, as mensagens de notificação são exibidas para o usuário em forma de notificação, neste caso em questão um aluno inscrito na van do motorista logado, enviou o aviso a sua não utilização do serviço.

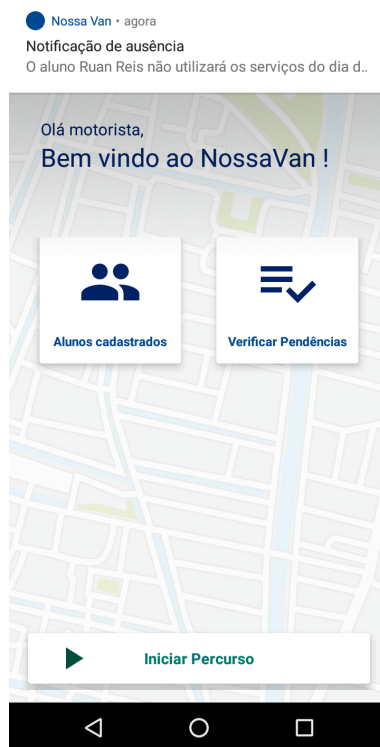


Figura 14. Cloud Messaging - Recebendo notificação de ausência

7. Conclusão

Este trabalho apresentou o projeto e desenvolvimento do aplicativo de conexão entre universitários, clientes de serviços de transporte escolar, e motoristas de vans, prestadores de serviços de transporte escolar. Os resultados foram satisfatórios para a solução proposta, de acordo com os casos de uso propostos, o princípio central foi atendido. O aluno cadastrado consegue identificar e contatar motoristas de transporte escolar. O sistema informa, baseado nos alunos inscritos na van, o trajeto otimizado, que deve ser realizado para o desembarque de alunos em suas respectivas residências cadastradas, considerando a localização atual do veículo.

Neste trabalho, não foram realizados testes de funcionamento completo em situações reais. Os testes foram subsidiados por simulações de operações do usuário em funcionalidades principais do sistema, como o compartilhamento de localização, inscrição em van e cálculo de rota otimizada. As funções do sistema oferecem respostas imediatas, desde que existe conexão com a internet ativa. A performance do sistema está relacionada diretamente com a conexão do dispositivo do usuário com a internet, não oferecendo suporte a funcionalidades *offline*.

A principal contribuição do projeto, é a materialização da solução proposta, contando com a integração com API's pagas possibilitam suportar grandes cargas de serviço. A infraestrutura é escalável e projetada para suportar cerca de mil usuários diários, característica essencial para a definição de um produto.

Como trabalhos futuros pode-se citar a avaliação da solução proposta com usuários reais e a implementação de novas funcionalidades, com a finalidade de tornar o

produto mais atrativo para o usuário e, conseqüentemente, aumentar o número de clientes. Funções que podem tornar o produto rentável com o modelo *freemium* são o compartilhamento de rota e localização para parentes, recomendações de prestadores de serviços com preços menores, inclusão de guia de rota através de áudio, mudança de trajeto de acordo com informações de trânsito, afim de otimizar o tempo, opções para recuperação e troca de senhas pelos usuários do aplicativo.

Apêndices

A. Análise de Viabilidade do Projeto Proposto

Após a verificação da situação do problema, que justifica o desenvolvimento deste trabalho, torna-se necessário a realização de uma análise de viabilidade do projeto. Neste caso deve se avaliar as condições do desenvolvimento do aplicativo, para analisar a viabilidade de sua continuidade.

A.1. Viabilidade Técnica

No quesito técnico, existe uma grande variedade de *frameworks* e bibliotecas disponíveis que se adequam as necessidades operacionais do projeto, como, por exemplo, o GoogleMaps API (do inglês, *Application Programming Interface*) . O grande número de dispositivos ativos que utilizam o sistema operacional [ComTech 2018] em que o aplicativo será instalado significa que grande parte dos usuários possuem o requisito principal para utilização do serviço. A variedade de dispositivos com configurações de *hardware* diferentes viabiliza com que pessoas de diversas classes sociais possam ter acesso a um *smartphone* para a instalação do aplicativo. A equipe responsável pelo desenvolvimento possui conhecimento das ferramentas e tecnologias que foram utilizadas, minimizando os riscos relacionados a aprendizagem de novos conteúdos e má integração do sistema.

A.2. Viabilidade Operacional

Na análise da viabilidade operacional, há uma adequação da proposta para a resolução do problema identificado, visto que existe, citado anteriormente, a necessidade de um intermediador entre motorista e passageiro. Além disso, as duas personas, que foram definidas na Figura 2 e na Figura 3, se enquadram no perfil de usuário com alto nível de aceitação da proposta de solução.

A.3. Viabilidade Econômica

Na análise de viabilidade econômica levando em consideração os custos de desenvolvimento, foram utilizadas ferramentas gratuitas, com limite de acesso, no processo de desenvolvimento, que serão citadas na seção 5, enquadrando a solução em um custo-benefício que torna propício a continuidade deste trabalho. A equipe de desenvolvimento já possui os recursos materiais, que incluem *smartphones* e computadores, para a execução do projeto, mantendo o curso de operação baixo.

A.4. Viabilidade de Cronograma

Na análise de viabilidade de cronograma, a materialização da proposta deve ser feita em no máximo quatro meses, período entre agosto e novembro de 2018. A equipe é composta por dois desenvolvedores com experiência na área, o que reduz o tempo de entrega. O produto final, assim como proposto, é composto das funcionalidades básicas descritas no documento, para que seja possível a execução do projeto no tempo determinado.

A.5. Viabilidade do Projeto

Durante o estudo das quatro viabilidades essenciais do projeto proposto não foram identificadas características que inviabilizam o progresso do trabalho ou que geram obstáculos que prejudique a entrega do produto e resultados. Portanto, infere-se que o projeto e seu planejamento de execução são completamente viáveis.

Referências

- [Cardoso et al. 2016] Cardoso, B. M., Quintiliano, G. A., Passos, K. M. N. d., Gomes, M. C. S., Passos, S. D. N. d., and Luz, L. P. d. (2016). Sistrans (sistema de controle de passageiros para transporte escolar). *FATEC Garça*, 6(1).
- [Cardoso et al. 2011] Cardoso, E. J., Santos, M. J., and Carniello, M. F. (2011). O processo de urbanização brasileiro. Disponível em: www.inicepg.univap.br/cd/INIC_2011/anais/arquivos/0088_0295_01.pdf. XV Encontro Latino Americano de Iniciação Científica e XI Encontro Latino Americano de Pós-Graduação – Universidade do Vale do Paraíba.
- [Chaves et al. 2013] Chaves, I., Bittencourt, J. P., and Taralli, C. (2013). O design centrado no humano na atual pesquisa brasileira - uma análise através das perspectivas de klaus krippendorff e da ideo. *HOLOS*, 6(0):213–225.
- [ComTech 2018] ComTech, K. W. (2018). Android vs ios. Disponível em: <https://www.kantarworldpanel.com/global/smartphone-os-market-share>, Acesso em Maio 2018.
- [Design 2018] Design, G. (2018). Material design by google. <https://material.io/design/>, Acesso em Novembro de 2018.
- [Foundation 2014] Foundation, I. D. (2014). Interaction design foundation. <https://www.interaction-design.org/literature/article/google-s-material-design-android-design-language>, Acesso em Novembro de 2018.
- [Gandhewar and Sheikh 2010] Gandhewar, N. and Sheikh, R. (2010). Google android: An emerging software platform for mobile devices. *International Journal on Computer Science and Engineering (IJCSSE)*, (NCICT 2010 Special Issue):12–17.
- [Google] Google. Firebase helps mobile app teams succeed. <https://firebase.google.com/>.
- [Lima et al. 2014] Lima, A., Alves, A., Costa, A., and Sales, E. (2014). Metodologia design thinking no projeto de software para mobilidade urbana: relato de aplicação. *AtoZ: novas práticas em informação e conhecimento*, 3(2):128–138.
- [Magagnin and Silva 2008] Magagnin, R. C. and Silva, A. N. R. (2008). A percepção do especialista sobre o tema mobilidade urbana. *TRANSPORTES*, XVI(1):25–35.

- [MEC and INEP 2017] MEC and INEP (2017). Censo da educação superior. http://portal.inep.gov.br/artigo/-/asset_publisher/B4AQV9zFY7Bv/content/mec-e-inep-divulgam-dados-do-censo-da-educacao-superior-2016/21206, Acesso em Fevereiro de 2018.
- [Nguyen 2016] Nguyen, P. (2016). Mobile Backend as a Service: The Pros and Cons of Parse Case company: SuperApp Oy. *Theseus - ammattikorkeakoulujen opinnäytetyöt ja julkaisut verkossa*, pages 8 – 11.
- [Zheng et al. 2014] Zheng, Y., Capra, L., Wolfson, O., and Yang, H. (2014). Urban computing: Concepts, methodologies, and applications. *ACM Trans. Intell. Syst. Technol.*, 5(3):38:1–38:55.