

**CENTRO UNIVERSITÁRIO DE ANÁPOLIS – UniEVANGÉLICA
BACHARELADO EM ENGENHARIA DE COMPUTAÇÃO**

**SISTEMA GERENCIADOR DE LINHAS DO TRANSPORTE
ESCOLAR RURAL DO MUNICÍPIO DE SILVÂNIA**

RAPHAEL AUGUSTO NASCIMENTO RODRIGUES

**ANÁPOLIS - GO
2019**

RAPHAEL AUGUSTO NASCIMENTO RODRIGUES

**SISTEMA GERENCIADOR DE LINHAS DO TRANSPORTE
ESCOLAR RURAL DO MUNICÍPIO DE SILVÂNIA**

Projeto de Pesquisa apresentado como requisito parcial para a conclusão da disciplina de Trabalho de Conclusão de Curso I do curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis – UniEVANGÉLICA.

Orientador(a): Prof. Millys Fabrielle de Araújo Carvalhaes


ANÁPOLIS - GO
2019

RAPHAEL AUGUSTO NASCIMENTO RODRIGUES

**SISTEMA GERENCIADOR DE LINHAS DO TRANSPORTE ESCOLAR RURAL DO
MUNICÍPIO DE SILVÂNIA**

Trabalho de Conclusão de Curso II apresentado como requisito parcial para a obtenção de grau do curso de Bacharelado em Engenharia de Computação do Centro Universitário de Anápolis – UniEVANGÉLICA.

Aprovado(a) pela banca examinadora em 28 de novembro de 2019, composta por:



Millys Fabrielle Araujo Carvalhaes
Presidente da Banca



Kleber Silvestre Diogo
Prof(a). Convidado(a)



Walquiria Fernandes Marins
Prof(a). Convidado(a)

SISTEMA GERENCIADOR DE LINHAS DO TRANSPORTE ESCOLAR RURAL DO MUNICÍPIO DE SILVÂNIA

RESUMO

Tem-se como objetivo o desenvolvimento de um sistema para ser utilizado pela secretaria de Educação do município de Silvânia para gerenciar as rotas do transporte escolar rural, controlar dados de alunos que utilizam o serviço de transporte e gerar relatórios sobre as viagens feitas diariamente. O sistema a ser apresentado neste projeto deve funcionar em locais sem conexão com a *internet*, e a metodologia de desenvolvimento com *Progressive Web Apps* (PWA) será o modelo adotado para a resolução, pois há a necessidade de utilizar o sistema em momentos que possa não haver conexão, sendo possível, com essa tecnologia, registrar dados em modo *offline* e obter pontos da viagem sempre que a conexão seja reestabelecida. A obtenção de dados sobre as viagens feitas é importante para o levantamento de verbas e recursos para o transporte, assim como o sistema poderá atuar como ferramenta mais precisa do controle dos dados.

Palavras-chave: PWA. Transporte Escolar Rural. Multiplataformas. Software.

Abstract

The objective is to develop a system to be used by the Silvânia Department of Education to manage rural school transport routes, control student data using the transport service and generate reports on daily trips. The system to be presented in this project needs to be functional during times when there is no internet connection, and the development methodology with Progressive Web Apps (PWA) will be the model adopted for the resolution, because there is a need to use the system in moments that may not be connected, with this technology it is possible to record data offline and to obtain travel points whenever the connection is reestablished. Obtaining data on travel is important for the collection of funds and resources for transportation, as well as the system can act as a more accurate tool for data control.

Keywords: PWA. Rural School Transport. Multiplatforms. Software

LISTA DE ILUSTRAÇÕES

Figura 1 – Quadro Kanban.....	15
Figura 2 – Tecnologias relacionadas ao Ionic.....	20
Figura 3 – Arquitetura do Cordova.....	21
Figura 4 – Demonstração da estrutura do banco de dados JSON	23
Figura 5 – Estrutura do arquivo <i>manifest</i>	25
Figura 6 – Ciclo de funcionamento de um PWA.....	25
Figura 7 – Representação do armazenamento em <i>cache</i>	26
Figura 8 – Ciclo de vida de um <i>service worker</i>	27
Figura 9 – Visão geral das <i>sprints</i>	28
Figura 10 – Diagrama de representação da arquitetura do sistema.....	33
Figura 11 – Diagrama de Implantação.....	34
Figura 12 – Classe AuthProvider.....	36
Figura 13 – Classe AlunosProvider.....	37
Figura 14 – Classe AlunosProvider, funções <i>save</i> e <i>remove</i>	38
Figura 15 – Protótipo de tela “Novo Aluno”	39
Figura 16 – Protótipo de tela “Rotas”	40
Figura 17 – Protótipo de tela “Rota selecionada”	40
Figura 18 – Auditoria do sistema sendo executado no navegador Google Chrome.....	41
Figura 19 – Resultado da auditoria do sistema	42
Figura 20 – Resultado final da auditoria	44
Figura 21 – EAP.....	45

LISTA DE TABELAS

Tabela 1 - Formato de <i>User History</i>	18
Tabela 2 – Permissões de usuário.....	35

LISTA DE SIGLAS

GESTA	Galeria de Estudos e Avaliação de Iniciativas Públicas
SIGE	Sistema Integrado de Gestão Educacional
URL	<i>Uniform Resource Locator</i>
API	<i>Application Programming Interface</i>
SDK	<i>Software Development Kits</i>
PWA	<i>Progressive Web Apps</i>
HTML	<i>HyperText Markup Language</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
HTTPS	<i>Hyper Text Transfer Protocol Secure</i>
CSS	<i>Cascading Style Sheets</i>
CLI	<i>Command-line Interface</i>
DOM	<i>Document Object Model</i>
TLS	<i>Transport Layer Security</i>
SSL	<i>Secure Sockets Layer</i>
JSON	<i>JavaScript Object Notation</i>
SQL	<i>Structured Query Language</i>
NPM	<i>Node Package Manager</i>
UML	<i>Unified Modeling Language</i>
CRUD	<i>Create, Read, Update e Delete</i>

SUMÁRIO

1. Introdução	11
1.2. Objetivo Geral.....	12
1.2. Objetivos Específicos.....	12
1.2. Justificativa.....	12
2. Fundamentação Teórica	13
2.1. Engenharia de Software.....	13
2.2. Metodologias Ágeis.....	14
2.2.1. Kanban.....	15
2.2.2. Scrum.....	16
2.2.2.1. Sprint.....	17
2.2.3. User History.....	17
2.2.4. Casos de Uso.....	18
2.3. Framework.....	19
2.3.1. Ionic.....	20
2.3.2. Apache Cordova.....	21
2.3.3. Banco de dados.....	22
2.4. PWA.....	23
2.4.1. Service Worker.....	26
3. Desenvolvimento	28
3.1. Metodologia de pesquisa.....	28
3.2. Visão Geral.....	30
3.3. Sprint 1 – Modelagem do sistema	30
3.4. Sprint 2 – Configuração de ambiente, arquitetura, diagrama de implantação e UH001.....	31
3.4.1. Arquitetura.....	32
3.4.2. Diagrama de implantação.....	34
3.4.3. Desenvolvimento UH001 – Login de Usuário.....	35
3.5. Sprint 3 – Desenvolvimento UH002, UH003, UH004 e UH005.....	36
3.6. Sprint 4 – Desenvolvimento UH006 e UH007.....	39
3.7. Sprint 5 – Aplicando os recursos de PWA.....	41
3.7.1. Criando o Service Worker.....	43
3.7.2. Manifesto.....	43
3.7.3. Auditoria final.....	44
3.8. Resultados.....	45

4. Trabalhos Futuros.....	46
5. Considerações finais.....	47
Referências Bibliográficas.....	48
APÊNDICES.....	51
APÊNDICE A – Documento de Visão.....	51
APÊNDICE B – Diagrama de Caso de Uso.....	54
APÊNDICE C – UH001 LOGIN DE USUÁRIO.....	55
APÊNDICE D – UH002 CADASTRAR ALUNO.....	57
APÊNDICE E – UH003 LISTAR ALUNO.....	60
APÊNDICE F – UH004 ALTERAR ALUNO.....	62
APÊNDICE G – UH005 EXCLUIR ALUNO.....	64
APÊNDICE H – UH006 MATER ROTAS.....	66
APÊNDICE I – UH007 REGISTRAR VIAGEM.....	68
APÊNDICE J – TELAS DO SISTEMA.....	70

1. Introdução

No Brasil, 2.802.258 crianças e adolescentes de 4 a 17 anos estão fora da escola, segundo a Pesquisa Nacional por Amostra de Domicílios. Os maiores índices de exclusão se concentram na zona rural. Dispersos pelo Brasil, 661.110 meninos e meninas estão longe das salas de aula fora das cidades, vilas e áreas urbanas isoladas (UNICEF, 2017).

Um dos grandes agravantes que contribuem para a evasão escolar em todo o Brasil, segundo o GESTA – Engajamento Escolar (2018), é a distância da residência do aluno até a escola. Por falta de empenho dos municípios para oferecer um transporte de qualidade e seguro, muitas crianças deixam de frequentar as aulas, e este cenário não é diferente no município de Silvânia/GO. Segundo a Constituição Federal (1988), município e estado devem atuar de forma prioritária, fornecendo o transporte escolar gratuito ou meios para que os alunos cheguem até as escolas, determinado pela Lei nº 9.394/96. Mesmo se tratando de uma questão indispensável, a grande maioria dos municípios brasileiros passam por dificuldades com o transporte escolar, basta uma rápida pesquisa em noticiários sobre a situação do transporte nos municípios para notar que o problema acontece a muito tempo. Qual seria uma iniciativa que poderia ser feita pelos municípios para melhorar este cenário?

A princípio, uma gestão bem definida dos recursos, rotas, e distribuição de alunos é fundamental para que a qualidade do transporte seja inicialmente construída, pois periodicamente, a gestão do transporte precisa atualizar dados dos alunos, bem como situações dos ônibus e estradas em um sistema do governo estadual para obtenção de verbas e recursos para a manutenção do transporte. O cenário do transporte escolar rural do município de Silvânia dispõe de 35 ônibus, atendendo 1.204 alunos alocados em 36 rotas, percorrendo uma média diária de 5.348 km, de acordo com dados do Sistema Integrado de Gestão Escolar (SIGE, 2018).

Neste cenário, há um fator de empecilho, que é a falta de conexão com a internet por se tratar de zonas rurais. A cobertura de sinal de rede móvel nessas regiões é pequena, de acordo com os mapas de cobertura da Vivo e TIM (operadoras de telefonia móvel com maior cobertura e disponibilidade da região), o que dificulta a utilização de um sistema. Portanto, utilizando recursos da tecnologia, como será possível gerenciar rotas e armazenar dados sobre as viagens, em regiões onde há ausência de conexão?

1.2 Objetivo Geral

Desenvolver um *software* para realizar o controle de rotas das linhas do transporte rural do município de Silvânia.

1.3 Objetivos Específicos

- Levantar os requisitos para se desenvolver um software para controle de dados de alunos e viagens feitas pelo transporte escolar;
- Projetar o sistema utilizando a tecnologia de desenvolvimento capaz de executar a aplicação em multiplataformas e em ausências de conexão;
- Implementar os requisitos mínimos com o intuito de criar um protótipo do sistema;
- Validar o sistema para verificar o funcionamento conforme os objetivos definidos da utilização deste sistema.

1.4 Justificativa

As tecnologias de desenvolvimento de *softwares* estão em constante evolução, e essa evolução se passa por meio da grande ascensão de dispositivos móveis no mundo. Segundo dados da GSMA Intelligence (2018), já existem mais dispositivos *mobile* do que pessoas no planeta. O número de *gadgets* já supera os 8,5 bilhões de aparelhos.

Por meio da metodologia PWA, do inglês *Progressive Web App*, o aplicativo se distingue da aplicação nativa pelas suas capacidades de ser executada em várias plataformas, independentemente do sistema operacional, pois é executada na *web* através do *browser*. Isto dispensa a instalação de uma aplicação, estando acessível através do típico URL, o que poupará recursos e memória no dispositivo. Independentemente da situação de conexão com a rede, uma aplicação em PWA funcionará normalmente. Por meio de um *service worker*, escrito em *Javascript* que funciona como um *proxy* do lado do cliente, a aplicação pode armazenar dados e recursos em *cache*, sendo possível eliminar a dependência da rede para utilizar a aplicação. (Google Developers, 2019)

Sendo assim, a motivação para a construção deste projeto é apresentar uma solução tecnológica para a utilização de um *software* que funcione em cenários que a conexão de rede seja inexistente. Com base em tecnologias atuais e pertinentes ao problema, é possível a construção de um *software* para o uso em locais remotos como apresenta o cenário deste

projeto. A população do município que necessita do transporte escolar poderá se beneficiar com uma possível melhora na gestão do transporte, pois a administração terá em mãos os dados e relatórios de todas as viagens, bem como a possibilidade de encaminhar estes relatórios e a relação total de alunos, motoristas e frota para os órgãos superiores do estado, na busca de novos recursos e verbas para o transporte.

2. Fundamentação Teórica

O projeto de pesquisa deste tema é baseado nas diretrizes da engenharia de *software*, com foco na solução para *softwares* com suporte a multiplataformas e que seja funcional com a ausência de conexão de rede. É importante que se defina cada tecnologia a ser estudada para chegar ao objetivo deste projeto.

As definições presentes na engenharia de *software* são importantes para se projetar um sistema aceitável e um produto de qualidade, em conjunto com a metodologia de desenvolvimento em PWA e utilizando metodologias ágeis na produção da documentação do projeto.

2.1. Engenharia de Software

O desenvolvimento de um *software* é um processo que se tem evoluído constantemente. Sempre que se ouve falar em desenvolvimento de *software* não se imagina que nos primórdios da informática a programação era elaborada por técnicos que ligavam e desligavam os cabos de acordo com tomadas de decisões pré-estipuladas. Pode-se dizer que os primeiros computadores do mundo, caso do ENIAC americano e Z3 alemão, trouxeram os primeiros *softwares* do mundo. Representavam verdadeiras montanhas repletas de válvulas capazes de ocupar algum pequeno prédio de quatro andares (PLANTIER, 2013).

Na atualidade, os *softwares* são desenvolvidos com linguagem específica de programações armazenáveis até mesmo em pequenos chips menores do que o dedo mindinho. Se for levado em consideração este tipo de *software*, se pode dizer que a primeira definição aceita no mundo acadêmico surgiu na Inglaterra, no ano de 1948, três anos após a Segunda Grande Guerra (PLANTIER, 2013).

Segundo Pressman (2005), há 60 anos, ninguém poderia prever que o *software* se tornaria uma tecnologia indispensável para negócios, ciência e engenharia; que *software* viabilizaria a criação de novas tecnologias a extensão de tecnologias existentes e a mudança radical nas tecnologias mais antigas.

Nesse cenário surge a engenharia de software que foi definida por Friedrich em 1969 na Conferência NATO¹ *Science Committee*, como “o estabelecimento e uso de sólidos princípios de engenharia para obter software economicamente confiável e que trabalhe de forma eficiente em máquinas reais” (NATO, 1969).

Pádua (2009) trata a engenharia de *software* como uma necessidade humana. Nisso, ela tem escopo bem diverso da ciência. O conhecimento é certamente uma necessidade humana, mas uma entre várias outras de uma hierarquia. Todo produto de engenharia se justifica através da satisfação de uma necessidade; portanto, da geração de algo que tenha valor para alguém.

É neste ponto onde o *software* demonstra sua importância na sociedade. Como ressalta Pádua (2009) a Engenharia de Software se preocupa com o *software* como produto. Partindo deste ponto o *software* é um produto como qualquer outro produto palpável, havendo todo um processo de sua criação, uma equipe envolvida, um processo de desenvolvimento bem definido, controle de qualidade, testes e entrega.

2.2. Metodologias Ágeis

Os métodos ágeis, como o próprio nome diz, envolve um conjunto de metodologias que servem para acelerar o ritmo dos processos de desenvolvimento de *software*. Com sua origem datada em meados dos anos 1990, o conceito de *Agile* não demorou a ser difundido entre os especialistas, o que resultou na criação de diferentes modelos que dão suporte à gestão de projetos (JUNIOR, 2017).

Segundo Junior (2017), as metodologias ágeis contribuem no desenvolvimento do projeto trazendo benefícios tanto para a equipe de desenvolvimento, quanto para o cliente. Na visão da equipe pode-se destacar os benefícios:

- Entregas rápidas e frequentes
- Qualidade do produto
- Previsão de cronogramas e custos
- Mitigação de Riscos

Quando se trata do lado do cliente, os benefícios que as metodologias ágeis oferecem são:

- Agilidade

¹ *North Atlantic Treaty Organization*

- Múltiplas entregas
- Participação no projeto
- Customização do produto

Todo projeto precisa ter um início e um fim bem definidos, mas sua execução pode se estender por períodos de mais de anos – durante os quais muita coisa pode acontecer. Então, era preciso desenvolver métodos inteligentes e eficientes que conseguissem contornar esses problemas e que pudessem simplificar a forma como os projetos eram executados – gerando impactos positivos em sua finalização (SAMBATECH, 2019).

Neste projeto, a utilização de metodologia ágil acontece na fase de levantamento de requisitos, mais especificamente na adoção do uso de *User Story* (em português, Histórias de Usuário), que é uma metodologia de levantamento de requisitos originada a partir da criação da metodologia ágil *Extreme Programming* (XP) criada em 1998.

2.2.1. Kanban

Surgida no Japão na década de 1960 na fábrica da Toyota, o sistema Kanban era usado para definir cada uma das etapas de produção e levantar possíveis problemas que poderiam atrapalhar o processo (INVETTI, 2018).

No Kanban são utilizados cartões com cores e tamanhos diferentes para definir e descrever as tarefas que precisam ser feitas, sendo divididas em 3 etapas de processo, “A fazer”, “Em Desenvolvimento” e “Entregue”. A Figura 1 abaixo representa um quadro de Kanban simples.

Figura 1 – Quadro Kanban



Fonte: DevMedia

O Kanban é um dos métodos de desenvolvimento de software menos prescritivo, se tornando adaptável a quase qualquer tipo de cultura. Ao contrário de outros métodos que forçam uma mudança desde o início, o Kanban busca a evolução, não a revolução.

Possui quatro princípios fundamentais, são eles: Comece com o que você faz agora; concordar em buscar mudanças evolucionárias; inicialmente, respeite os papéis, responsabilidades e cargos atuais; incentivar atos de liderança em todos os níveis.

O Kanban lhe ajuda a assimilar e controlar o progresso de suas tarefas de forma visual. É, normalmente, utilizado um quadro branco com alguns pequenos papéis (Post-it) colados, esses papéis representam as suas tarefas, ao término de cada tarefa o papel é puxado para a etapa seguinte até que a mesma seja finalizada. Ao olhar para um quadro Kanban é fácil enxergar como o trabalho seu e de sua equipe fluem, permitindo não só comunicar o status, mas também dar e receber feedbacks (BERNARDO, 2014).

2.2.2. Scrum

Quando se trata de metodologias de desenvolvimento ágil, o Scrum pode ser uma metodologia a ser utilizada para o processo de desenvolvimento além de possibilitar o gerenciamento do projeto. Inicialmente o Scrum era um estilo de gerenciamento de produtos em empresas de automóveis, concebido por Takeuchi e Nonaka em 1986, e foi adaptado a uma metodologia ágil de desenvolvimento no ano de 1990.

Segundo Bernardo (2015), o Scrum é um *framework* para gerenciamento de projetos complexos, sendo um dos métodos ágeis mais populares do mundo, que busca entregar resultados de maneira mais rápida e com menor custo, focando em fornecer produtos e serviços que se alinhem as necessidades do cliente.

Ao implementar o Scrum, é importante que todas as partes envolvidas se concentrem com a ideia de o “cliente em primeiro lugar”. Estando com essa visão bem estabelecida, será possível entregar produtos que realmente impactem os negócios do cliente.

Implementar o Scrum não precisa ser uma coisa assustadora. Claro, o Scrum é sim diferente das abordagens tradicionais para gerenciamento de projetos, mas essa diferença é a principal razão pela qual ele funciona. Como em qualquer tipo de adoção você irá passar por pontos altos e desafiadores, mas os resultados poderão ser bem gratificantes. O Scrum fornece resultados para melhorar suas taxas de sucesso e, ao mesmo tempo, consegue construir equipes mais engajados e com níveis mais altos de satisfação. (BERNARDO, 2015)

2.2.2.1 Sprint

O Scrum tem como característica a produção de atividades por meio de pequenos ciclos de atividades que são chamadas de *Sprint*. “Uma sprint se integra ao método Scrum como uma forma de facilitar a divisão de um projeto em etapas ao longo do tempo com reuniões diárias, definição de metas e um bom fluxo de trabalho” (JUNIOR, 2017).

Pode ser considerado um dos pilares da metodologia Scrum pois nele são aplicados os eventos, produzidos os artefatos e desenvolvido de fato o produto. É durante uma *sprint* em que ocorre a produção do produto, ou parte dele.

A criação de um sprint envolve um trabalho constante de comunicação entre os times de desenvolvimento, o *Scrum Master* e o *Product Owner*. Eles devem compartilhar suas necessidades, sua capacidade de produção e sua evolução no alcance das metas, a fim de evitar a quebra de expectativas ao final de cada etapa. Antes de se criar os sprints de um projeto, é preciso definir quais são as funcionalidades do produto a ser desenvolvido e que são desejadas pelo *Product Owner*. (JUNIOR, 2017)

2.2.3. User History

Bernardo (2014) define as histórias de usuário como uma curta e simples descrição da necessidade do cliente, normalmente contada a partir da perspectiva de quem precisa da nova necessidade, sendo geralmente um usuário, cliente do sistema ou representante de negócios do cliente.

Segundo Meller (2016), as histórias de usuário são uma descrição de uma necessidade do usuário do produto sob o ponto de vista deste usuário, em palavras simples, é o requisito sendo contado pelo usuário. A *User History* busca descrever essa necessidade de uma forma simples e leve.

Uma história de usuário deve explicar bem para quem, o que e por que está sendo criada. Isso é importante, principalmente, para desenvolvedores que poderão construir softwares com menor dificuldade e maior qualidade, entendendo os objetivos e necessidades do cliente de forma mais rápida (BERNARDO, 2014).

Para se criar as Histórias de Usuário é definido quem receberá essa funcionalidade, a descrição da funcionalidade e o motivo para qual a funcionalidade será desenvolvida. São informações fundamentais para qualquer história e precisam constar na especificação dos requisitos.

O formato a ser seguido das histórias pode ser genérico, dependendo apenas que siga a estrutura base da metodologia. A Tabela 1 a seguir é o formato utilizado para o levantamento de requisitos deste projeto.

Tabela 1 – Formato de User Story

<p>Eu como: Usuário do tipo gestor</p>
<p>Quero: Realizar o cadastro de um aluno</p>
<p>Para: Manter no sistema a base de dados de todos os indivíduos que farão parte do processo de transporte escolar do município</p>

Fonte: do Autor

Com essas informações ficará mais fácil para o desenvolvedor elaborar uma funcionalidade que expresse realmente a necessidade do cliente.

2.2.4. Casos de Uso

Segundo Vieira (2015), o diagrama de Casos de Uso auxilia no levantamento dos requisitos funcionais do sistema, descrevendo um conjunto de funcionalidades do sistema e suas interações com elementos externos e entre si.

Esse diagrama documenta o que o sistema faz do ponto de vista do usuário. Em outras palavras, ele descreve as principais funcionalidades do sistema e a interação dessas funcionalidades com os usuários do mesmo sistema. Esse diagrama não se aprofunda em detalhes técnicos que dizem como o sistema faz.

Quando falamos de casos de uso, temos que ter em mente o conceito de cenários, que seriam instâncias de casos de uso. Um cenário pode ser compreendido como uma sequência de passos que descreve uma interação entre um usuário e o sistema (VIEIRA, 2015).

Em resumo, os diagramas de Casos de Uso:

- auxiliam na comunicação entre o cliente e os analistas.
- apresentam as principais funcionalidades do sistema com foco no cliente.
- descrevem cenários de interação entre as partes internas/externas de um sistema, com foco no usuário.
- é muito utilizado na fase de levantamento de requisitos.

Segundo Vieira (2015), os principais elementos de um diagrama de casos de uso são:

- **Atores:** boneco com rótulo que representa um humano ou um sistema computacional.
- **Caso de Uso:** elipse com rótulo que representa uma funcionalidade do sistema, sendo que esta pode estar estruturada em outra(s).
- **Relacionamentos:** auxiliam na descrição dos casos de uso, podendo ser: entre um ator e um caso de uso, entre atores e entre casos de uso.

2.3. Framework

Segundo Jaques (2016), um *framework* é um conjunto de códigos abstratos, geralmente classes, desenvolvidos em alguma linguagem de programação, que se relacionam entre si para disponibilizar funcionalidades específicas ao desenvolvedor de *software*. Basicamente, um *framework* trabalha com diversas funcionalidades já implementadas, testadas e prontas para a construção de um *software*, com o intuito de evitar tarefas repetitivas em funcionalidades que são comuns a vários *softwares*.

A utilização de *frameworks* é recomendada quando, ao desenvolver uma aplicação, você precisa realizar tarefas repetitivas ou que são comuns a vários sistemas, não há por que perder tempo montando e testando um sistema para validação de dados se existe uma ferramenta que já faz isso, sendo já utilizada e verificada por diversos desenvolvedores (JAQUES, 2016).

Jaques (2016) ainda reforça que embora a utilização de *frameworks* seja bastante interessante aos olhos dos desenvolvedores, é preciso cautela na hora de escolher o *framework* certo para se trabalhar, além disso, é importante ressaltar alguns pontos negativos dessa ferramenta:

- Iniciar um desenvolvimento utilizando um *framework* pode causar uma dependência sobre a tecnologia escolhida, sendo bastante difícil migrar todo um sistema já escrito utilizando determinado *framework*, podendo a única saída ser reescrever todo o sistema novamente.
- É necessário conhecer bem a linguagem em que está trabalhando com o *framework*, já que, um *framework* em si não é uma linguagem, e sim uma abstração da

linguagem, podendo forçar o desenvolvedor a aprender peculiaridades do *framework* e não da linguagem que está por baixo dele.

- Modificar um *framework* não é tarefa fácil. Se for necessário modificar o núcleo de uma funcionalidade do *framework*, essa tarefa pode se tornar um problema, a menos que o desenvolvedor conheça totalmente a sua estrutura.

2.3.1. Ionic

Seguindo a metodologia do projeto, onde será preciso utilizar um *framework* capaz de criar aplicações progressivas (PWA), pode-se apresentar o Ionic, um *framework open source* que possibilita o desenvolvimento de aplicações mobile multiplataformas. Sua implementação se baseia em tecnologias comumente empregadas na construção do *front-end* de soluções *web* como HTML, CSS e JavaScript, com um diferencial sobre esses recursos, ao adotar como base o Apache Cordova, que traz recursos que simplificam ainda mais o desenvolvimento.

Na Figura 2 a seguir, são apresentadas as ferramentas que compõe o framework. Utiliza o Angular, que é um *framework* JavaScript desenvolvido pela Google. Os aplicativos desenvolvidos com Ionic são como pequenos sites que funcionam em um uma camada embutida do navegador em um aplicativo, o qual possui acesso à camada da plataforma nativa através de bibliotecas fornecidas pelo próprio Ionic (DA SILVA; SOTTO, 2018).

Figura 2 – Tecnologias relacionadas ao Ionic



Fonte: DevMedia

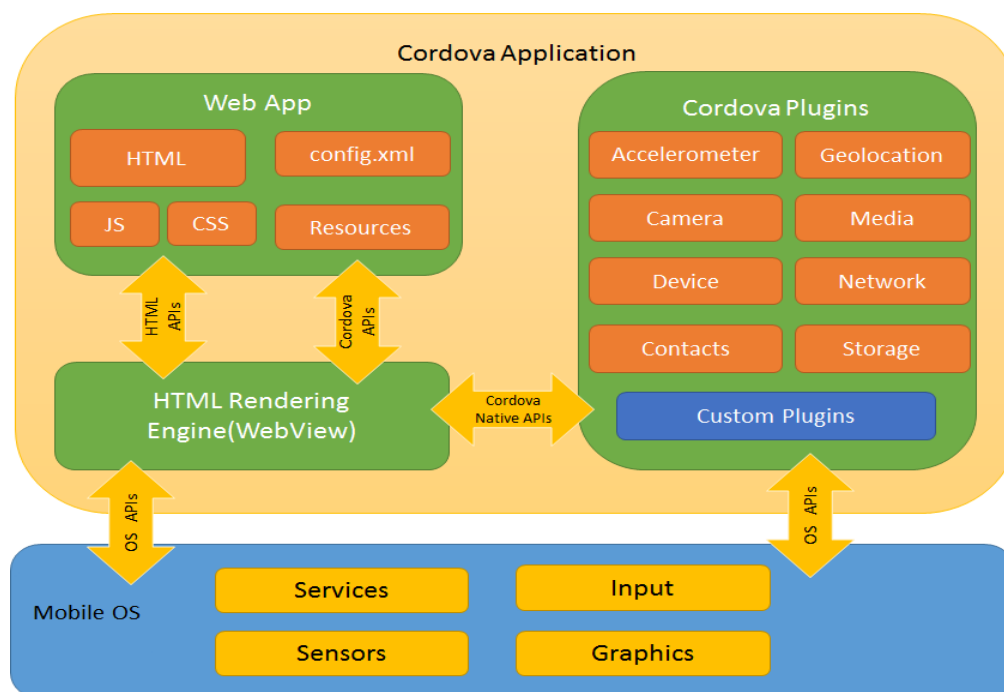
2.3.2. Apache Cordova

Conhecido basicamente como Cordova, este é um *framework* que segue a proposta do projeto de desenvolvimento multiplataformas, oferecendo suporte a iOS, Android e Windows Phone (descontinuado).

Assim como os outros frameworks abordados, o Cordova tem como objetivo a redução de trabalho, reaproveitamento de código para plataformas distintas e executar em qualquer plataforma uma aplicação escrita com o mesmo código. Se baseia nas linguagens HTML, CSS e Javascript, trazendo alguns diferenciais em relação a outros frameworks como uma extensa biblioteca de extensões, disponibilidade de ferramentas gratuitas de apoio ao desenvolvimento, como a IDE XDK, criada pela Intel, além de seu uso ser de licença de código aberto, sem custos e permitindo reprodução, exibição, distribuição, etc. (GASPAROTTO, 2017).

Na Figura 3 a seguir, pode-se observar a estrutura do *framework*.

Figura 3 – Arquitetura do Cordova



Fonte: Apache Cordova

Uma aplicação Cordova é construída basicamente como uma página web, utilizando CSS, HTML e Javascript para definir sua aparência, estrutura e comportamento. O programador pode utilizar, em meio ao seu código Javascript, extensões que dão acesso aos recursos de hardware do aparelho e disponibilizam funcionalidades básicas às aplicações, incluindo acesso a sensores diversos, câmera, memória secundária e algumas funcionalidades

de interface. Além das extensões centrais, mantidas pela Apache, o desenvolvedor tem a opção de incluir na aplicação diversas extensões gratuitas desenvolvidas por terceiros. E, caso não encontre uma extensão específica para atender determinada finalidade, é possível desenvolver sua própria extensão customizada. (CAMDEN, 2015, p.23)

A utilização do Cordova, apesar de agilizar bastante o processo de desenvolvimento de uma aplicação, deve-se entender que há aspectos que atuam negativamente na aplicação. Segundo Camden (2015), a sobreposição de camadas necessária para o funcionamento do framework pode impactar o desempenho da aplicação, além de requerer maior espaço em memória do dispositivo que uma aplicação nativa.

2.3.3.1 Banco de dados

Dada a natureza da aplicação desenvolvida, é necessário implementar um banco de dados para armazenamento de informações sobre os usuários do sistema, rotas e dados relacionados às viagens.

Para a implementação deste banco, foi constatado que a utilização do *Firebase*, que é uma API da Google que possui diversos serviços gratuitos com um limite de utilização, é a ferramenta mais adequada para o desenvolvimento de um protótipo do sistema, tendo em vista que o projeto não necessita de grandes volumes de armazenamento de dados e o limite de 100 acessos simultâneos na versão gratuita atende as necessidades do projeto.

Desenvolvido pela Google, o *Firebase* (FIREBASE,2019) é um conjunto de produtos e serviços para o desenvolvimento de aplicações distribuído gratuitamente, com um certo limite de utilização ou em larga escala de uso conforme planos que são pagos. Entre estes produtos, existem serviços de hospedagem, armazenamento em nuvem, e, o que foi utilizado neste projeto, um banco de dados em tempo real e o serviço de autenticação.

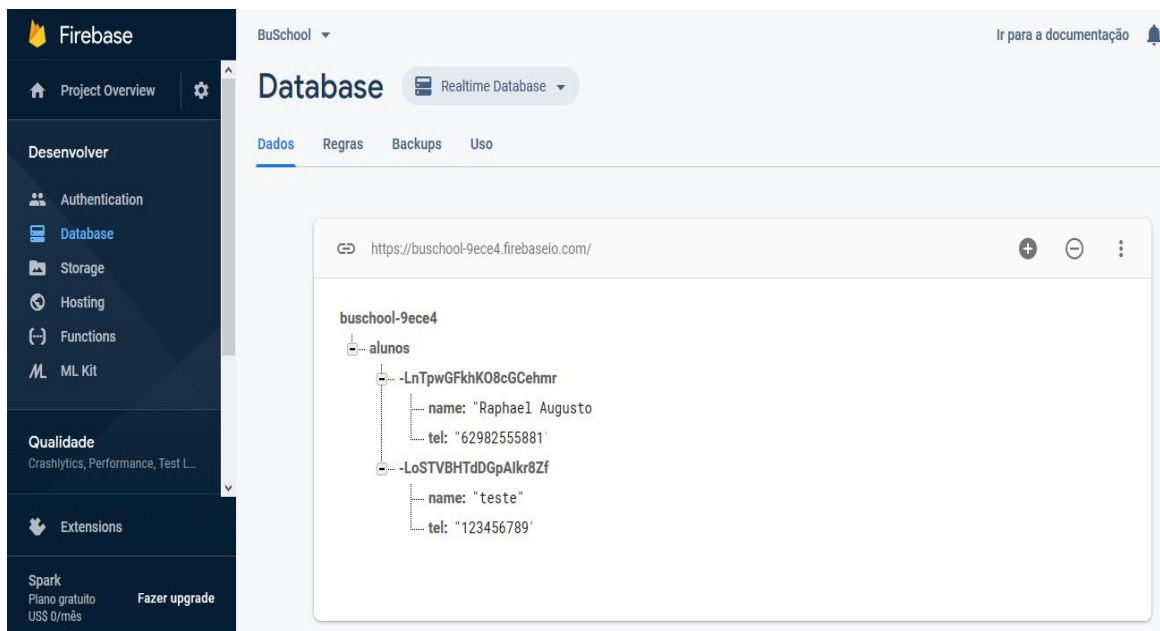
Essa ferramenta permite, com poucas linhas de código, adicionar o banco de dados em aplicações *web*, Android e iOS para que se conectem ao mesmo banco, sem que seja necessário conhecer sobre a infraestrutura do sistema.

Os dados do banco do *Firebase* são estruturados no modelo de árvore JSON, como é visto na Figura 4. Diferentemente de um banco de dados relacional (SQL), um banco de dados em JSON não possui tabelas ou registros.

Todos os dados do *Firebase Realtime Database* são armazenados como objetos JSON. Pense no banco de dados como uma árvore JSON hospedada na nuvem. Ao contrário de um banco de dados SQL, não há tabelas nem registros. Quando você adiciona dados à árvore JSON, eles se tornam um nó na estrutura JSON com uma chave associada. Forneça

chaves próprias, como códigos de usuário e nomes semânticos, ou gere uma usando *push()*. (FIREBASE, 2019)

Figura 4 – Demonstração da estrutura do banco de dados JSON



Fonte: Adaptado do console do Firebase

2.4. PWA

Progressive Web App é definido como uma metodologia de desenvolvimento de *software* capaz de construir uma aplicação híbrida entre páginas *web* regulares e um aplicativo móvel, isto é, um sistema que trabalha de forma dinâmica e responsiva para ser executado a partir de uma página *web* e ser utilizado em um *desktop* ou dispositivo móvel (DE SOUZA, 2017).

Segundo De Souza (2017), as aplicações em PWA são páginas *web* tecnicamente regulares, mas podem aparecer ao usuário como aplicativos tradicionais ou aplicativos móveis (nativos). Este novo tipo de aplicativo tenta combinar os recursos oferecidos pela maioria dos navegadores modernos com os benefícios da experiência móvel.

De Souza (2017) explica que PWA é uma aplicação *web* com tecnologias que permitem termos a experiência de uso muito próxima da oferecida pelos apps nativos de dispositivos móveis. As funcionalidades que estas tecnologias nos permitem são:

- *Push Notification* (Notificações);
- Ícone na tela *home* do *smartphone*;

- Processos executando em *background*;
- Suporte a funcionamento em modo *offline*;
- Acesso à câmera e galeria;
- Acesso à geolocalização;
- Acesso aos contatos.

Justen (2018) levantou um *checklist*, com base nas recomendações da Google, que diz o mínimo necessário para que uma aplicação seja considerada um PWA:

1 – Use HTTPS e redirecione sempre para HTTPS

O HTTPS é uma implementação do protocolo HTTP, sobre uma camada adicional de segurança que utiliza o protocolo SSL/TLS. Essa camada adicional permite que os dados sejam transmitidos por meio de uma conexão criptografada e também é feita uma verificação de autenticidade do servidor e do cliente por meio de certificados digitais (JUSTEN, 2018).

2 – Tenha um *site* responsivo

Se a ideia é dar uma interação como se fosse de um app, isso é uma regra mais que fundamental. E dentro dela, não se esqueça de definir a *meta tag* correta para a viewport:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

3 – Tenha uma cor tema no *site*

Para adicionar um tema no site, basicamente segue uma regra e para fazer isso é só adicionar essa meta tag:

```
<meta name="theme-color" content="cor em hexadecimal">
```

4 – Tenha um arquivo Manifest com informações do *site*

É um arquivo chamado *manifest.json* que vai conter informações relevantes do seu site, permitindo que o browser entenda que seu site é um PWA e vai inclusive habilitar uma mensagem para o usuário para que ele possa instalar o site na tela principal do celular, o que faz com o site fique com um visual igual de um aplicativo (JUSTEN, 2018).

E como esse *manifest.json* se parece? Segue abaixo, na Figura 5, o exemplo simples de algumas características presentes no arquivo.

Figura 5 – Estrutura do arquivo Manifest

```

{
  "name": "Willian Justen Blog",
  "short_name": "WJusten",
  "theme_color": "#005f97",
  "background_color": "#005f97",
  "display": "standalone",
  "scope": "/",
  "start_url": "?utm_source=homescreen",
  "lang": "pt-BR",
  "orientation": "any",
  "icons": [
    {
      "src": "/assets/img/icons/favicon-512x512.png",
      "sizes": "512x512",
      "type": "image/png"
    }
  ]
}

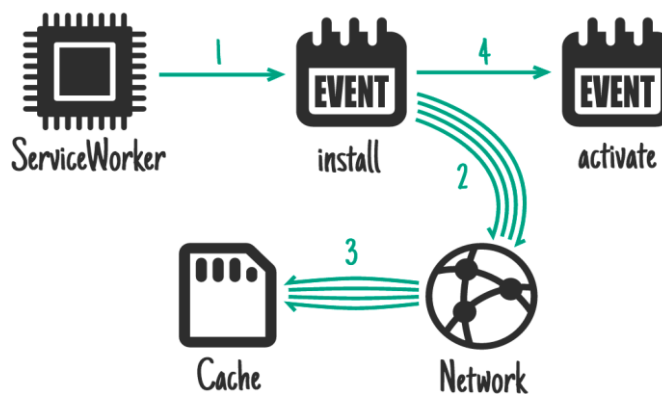
```

Fonte: Adaptado do Blog Willian Justen

5 – Registre um *Service Worker*

Justen (2018) explica que um *Service Worker* é um *script* que seu navegador executa em segundo plano, separado da página da *web*, possibilitando recursos que não precisam de uma página da *web* ou de interação do usuário. Atualmente, os *service workers* já incluem recursos como notificações *push* e sincronização em segundo plano. Possui a capacidade de interceptar e tratar solicitações de rede, respondendo com um *cache*, conforme indica a explicação resumida do funcionamento do *service worker* na Figura 6 a seguir.

Figura 6 – Ciclo de funcionamento de um PWA



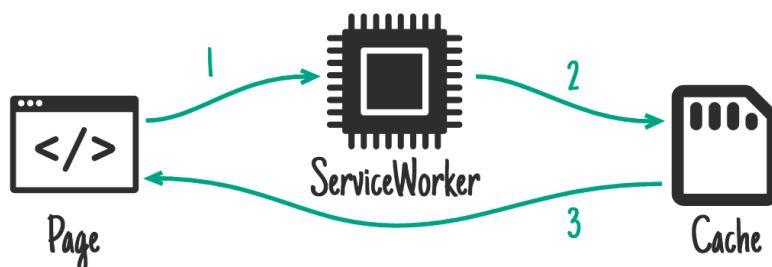
Fonte: CodeLabs – Google Developers

2.4.1. Service Worker

Para um PWA funcionar existe por trás dessa metodologia um *script* chamado *Service Worker*, e é importante definir e entender o que é um *service worker* para o desenvolvimento de aplicações em PWA.

Esse *script* é executado em segundo plano pelos navegadores, de um dispositivo *mobile* ou *desktop*, separado da página da *Web*, possibilitando que recursos sejam utilizados sem precisar da que a página *web* ou o usuário solicite seu uso, com isso um *service worker* se torna uma API que permite experiências *off-line*, armazenando dados em *cache*, como é representado na Figura 7 a seguir (GAUNT, 2019).

Figura 7 – Representação do armazenamento em *cache*



Fonte: Codelabs – Google Developers

Gaunt (2019) explica que mesmo que haja conexão, o PWA pode utilizar seus recursos do *service worker* para por exemplo carregar páginas *web* mais rapidamente. Ao obter o *cache* local armazenado, elimina qualquer variabilidade de rede. Não importa em que tipo de rede o usuário esteja (Wi-Fi, 4G, 3G ou até mesmo 2G), os principais recursos que serão executados, estarão disponíveis quase que imediatamente.

Justen (2018) destaca que um *service worker* tem características importantes, como:

- O SW² funciona num thread separado no *browser*, com isso não tem acesso ao DOM³;
- O arquivo do SW precisa sempre ter o mesmo nome e ficar sempre no mesmo lugar. Caso contrário, vai gerar uma duplicação de Service Worker;

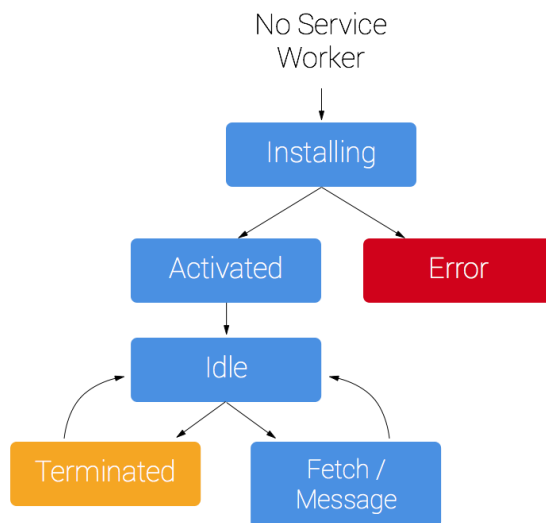
² Abreviação de Service Worker.

³ Document Object Model, em português, documento de modelo de objeto, é uma interface que representa como os documentos HTML e XML são lidos pelo navegador.

- O arquivo do SW não pode cachear de forma alguma, senão ele pode agir contra você e deixar um cache infinito na máquina do usuário. O certo é dentro do seu servidor você definir o *max-age* e colocar para sempre carregar de novo, sem cache;
- Fique atento ao ciclo de vida do SW, ao mesmo tempo que ele ajuda, também pode atrapalhar. Lembre-se sempre de deletar o cache antigo quando atualizar algo no site.

Um SW tem um ciclo de vida definido e totalmente separado de uma página *web*. Inicialmente instala-se no sistema, registra o SW, e assim o navegador inicia a etapa de instalação do SW em segundo plano. Durante a instalação, recursos estáticos são armazenados em *cache* e assim o SW estará instalado. Após a instalação o SW estará no controle em dois possíveis estados: encerrado, ou tratando eventos e mensagem gerados pela página ao enviar uma solicitação de rede, como mostra a figura 8 a seguir. (GAUNT, 2019).

Figura 8 – Ciclo de vida de um *service worker*

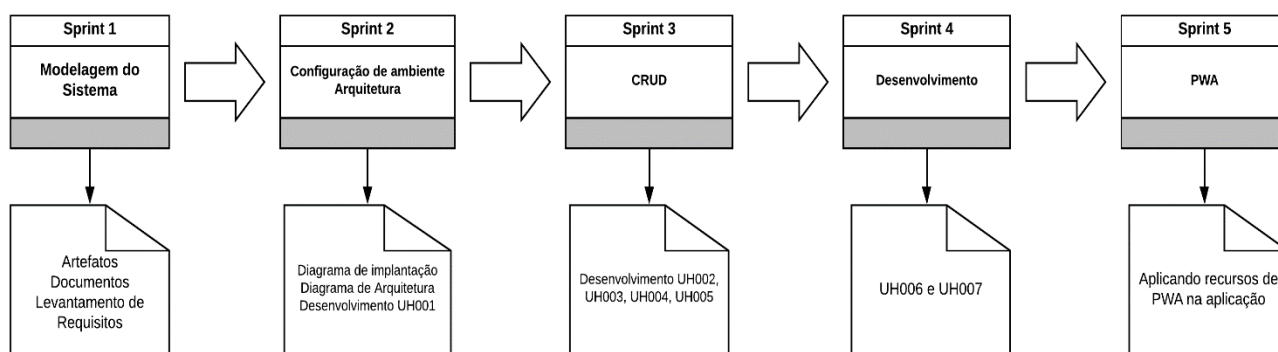


Fonte: Google Developers

3. Desenvolvimento

Neste capítulo, as etapas do desenvolvimento deste trabalho são detalhadas. Inicialmente, é mostrada uma visão geral sobre como o sistema foi desenvolvido, detalhando as técnicas e recursos utilizados. O processo de desenvolvimento foi dividido em *sprints*, com duração de 20 dias para cada *sprint*, sendo que no planejamento de sprints fora determinado a modelagem do sistema na primeira *sprint*, a arquitetura e configurações de ambiente na segunda, o desenvolvimento de requisitos para cada *sprint* seguinte, e no final a configuração da aplicação para um PWA, como indica a figura 9 a seguir.

Figura 9 – Visão geral das *sprints*



Fonte: do Autor

A partir da definição da arquitetura de desenvolvimento, aplicada ao *framework Ionic*, e o desenvolvimento dos requisitos em conjunto com os serviços da API do *Firebase*, foi possível obter um produto mínimo viável, ou seja, um protótipo para a demonstração do funcionamento do sistema.

3.1. Metodologia de pesquisa

A metodologia a ser empregada a fim de alcançar os objetivos do projeto define as tecnologias existentes que possam solucionar o problema apresentado, a forma como essa tecnologia deve ser trabalhada e os possíveis resultados que poderão ser obtidos para o software.

O método de pesquisa utilizado é o qualitativo, apoiando-se em técnicas de análise de frameworks, SDK's, ferramentas e outras tecnologias que possam suprir as necessidades encontradas no campo de atuação do problema.

O levantamento de requisitos do sistema será feito por meio da reunião com o responsável pela administração do transporte escolar do município, definindo necessidades e objetivos do software e discutindo os resultados que se espera assim que o projeto estiver concluído.

A aplicação *mobile* será utilizada pelos motoristas para registrarem suas viagens e também qualquer ocorrência ou divergência na rota. Por conta da mobilidade de um aparelho celular, a aplicação poderá acompanhar o motorista durante a viagem.

Na aplicação *web*, os gestores poderão administrar todo o fluxo de entradas, saídas e geração de dados, além de gerar relatórios que irão servir de objeto de estudo para elaborar melhorias no sistema de transporte escolar.

Por se tratar de um *software* que necessita da utilização de uma aplicação *web* e *mobile*, foi constatado que, o desenvolvimento utilizando a metodologia de desenvolvimento em PWA é o caminho certo para o desenvolvimento de um *software* que faça suprir as necessidades do problema, pois com essa tecnologia é possível criar uma aplicação híbrida que pode ser executada em um computador e em um *smartphone*.

É importante destacar que existem diversas tecnologias existentes no mercado que utilizam PWA, e é importante para a solução deste projeto levantar as características dessas tecnologias, esclarecer pontos positivos de cada uma e definir qual delas será pertinente para a construção de um *software* que atenda às necessidades do projeto.

Na fase final do projeto, será feita a validação do sistema, sendo necessário executar o *software* no cenário em que ele será aplicado, para garantir a conformidade do sistema com os requisitos levantados e funcionalidades desejadas.

A estrutura do projeto é formada por *front-end* e *back-end*. O *front-end* representa a “parte da frente” da aplicação, que segundo Viana (2000) é responsável por dar vida à interface, trabalhando com a parte da aplicação que interage diretamente com o usuário com a experiência do usuário. Já o *back-end* como a própria tradução (parte de trás) sugere trabalha o funcionamento da aplicação por trás das telas. Viana (2000) ressalta que este módulo de aplicação é responsável, em termos gerais, pela implementação da regra de negócio.

3.2. Visão Geral

Neste projeto é desenvolvido apenas o *front-end*, sendo que a parte de *back-end* da aplicação ficará por conta da API do Google Firebase, onde é possível desenvolver a aplicação com os serviços disponíveis pela plataforma, fazendo uso do banco de dados e do serviço de autenticação da API.

Para chegar às funcionalidades essenciais de funcionamento do sistema, foram utilizados o *framework* Ionic, que une HTML e JavaScript/TypeScript para desenvolvimento *front-end*, o gerenciador de pacotes NPM para configuração do ambiente de desenvolvimento e o editor de texto *Visual Studio Code*, que é a plataforma utilizada para codificação e execução do código.

3.3. Sprint 1 – Modelagem do sistema

O objetivo da primeira *Sprint* do projeto foi o desenvolvimento de artefatos que são importantes para a modelagem do sistema, que serão o caminho para o desenvolvimento do projeto. Baseando-se em reuniões com os *stakeholders* do projeto, foi possível levantar um documento de visão, as histórias de usuário que contemplam os requisitos do sistema e o diagrama de casos de uso.

Independente do porte de uma solução, todo projeto de *software* é caracterizado por um estágio inicial, sendo este momento uma fase de análise em que se procura estudar de que forma o mesmo será conduzido. A ênfase que é dada a este tipo de atividade depende, basicamente, não apenas do tamanho do sistema a ser construído, como também da forma como a equipe envolvida encontra-se estruturada e do conhecimento desta última acerca de padrões e metodologias próprios da área de *software*. (GROFFE, 2013)

O documento de visão (APÊNDICE A) do projeto contempla o escopo de alto nível e o propósito do projeto. Uma instrução clara do problema, solução proposta e os recursos de alto nível do produto ajudam a estabelecer expectativas e a reduzir riscos. Tem grande relevância durante as primeiras fases, permitindo a captura de todas as perspectivas que o sistema pode abranger.

Com base no documento de visão, foi possível construir o Diagrama de Caso de Uso (APÊNDICE B), que demonstra por meio de atores que interagem com o sistema, a forma como o sistema irá funcionar, representando as ações e responsabilidades do usuário dentro do sistema. O sistema terá dois atores essenciais para o funcionamento do sistema:

- Gestor: este ator possui todas as funções de cadastro do sistema, ele irá manter todos os outros atores, sendo responsável por gerenciar o sistema;
- Motorista: este ator terá função de acompanhar em quais ônibus o mesmo foi alocado, em quais rotas, quais alunos estão sob sua responsabilidade e registrar possíveis ocorrências sobre a viagem.

Com a finalização do diagrama de caso de uso foi possível levantar os requisitos do sistema de acordo com as metodologias ágeis de especificação de requisitos no formato User History (APÊNDICES C até I).

Segundo Silva (2017), a fase de levantamento de requisitos é definida como todas as atividades realizadas para identificar, analisar, especificar e definir as necessidades de negócio que um aplicativo deve prover para solução do problema levantado.

Para a construção deste projeto foram levantados os requisitos por meio de reuniões com funcionários da secretaria de Educação de Silvânia para chegar a um produto aceitável e que atenda a demanda do transporte escolar por um sistema de controle de rotas.

Sendo assim foram especificadas as seguintes User History:

- UH001. LOGIN DE USUÁRIO
- UH002. CADASTRAR ALUNO
- UH003. LISTAR ALUNO
- UH004. ALTERAR ALUNO
- UH005. EXCLUIR ALUNO
- UH006. MANTER ROTAS
- UH007. REGISTRAR VIAGEM

3.4. Sprint 2 – Configuração de ambiente, arquitetura, diagrama de implantação e UH001

Na segunda *Sprint* do projeto, teve como objetivo a configuração do ambiente de desenvolvimento, definição da arquitetura de desenvolvimento e também o início do desenvolvimento do primeiro requisito do sistema.

Para se desenvolver um *software* utilizando o *framework Ionic* e a API do *Firebase* fora necessário instalar as dependências existentes de cada ferramenta, e para essa finalidade foi utilizado o NPM que é um gerenciador de pacotes do Node.JS. “Por anos, o *Node* tem

sido amplamente usado por desenvolvedores *JavaScript* para compartilhar ferramentas, instalar vários módulos e gerenciar suas dependências” (LOPES, 2019).

Para a codificação do sistema foi utilizado o editor de textos Visual Studio, e as seguintes dependências instaladas:

- `npm install -g ionic@3.9.5`
- `npm install firebase@4.6.0`
- `npm install angularfire2@5.0.0-rc.3`

O processo de instalação de dependências adiciona e configura toda a conexão entre o código gerado pelo Ionic com o serviço de banco de dados do Firebase, sem que seja necessária a codificação manual da conexão com o banco de dados. Esta etapa adiciona linhas de código nos arquivos *manifest.json* e *package.json*, sendo estes, arquivos que possuem as dependências e configurações do ambiente.

3.4.1. Arquitetura

Uma arquitetura de desenvolvimento bem definida é importante na fase de início do desenvolvimento do projeto, pois contempla a estrutura de organização dos códigos, a forma de comunicação entre diversos componentes do sistema e facilita a manutenibilidade do sistema, já que oferece um padrão de desenvolvimento em que separa módulos do sistema, para que não seja apenas um bloco de código.

Garlan e Shaw (1993), sugerem que a arquitetura de *software* é um nível de design voltado para problemas.

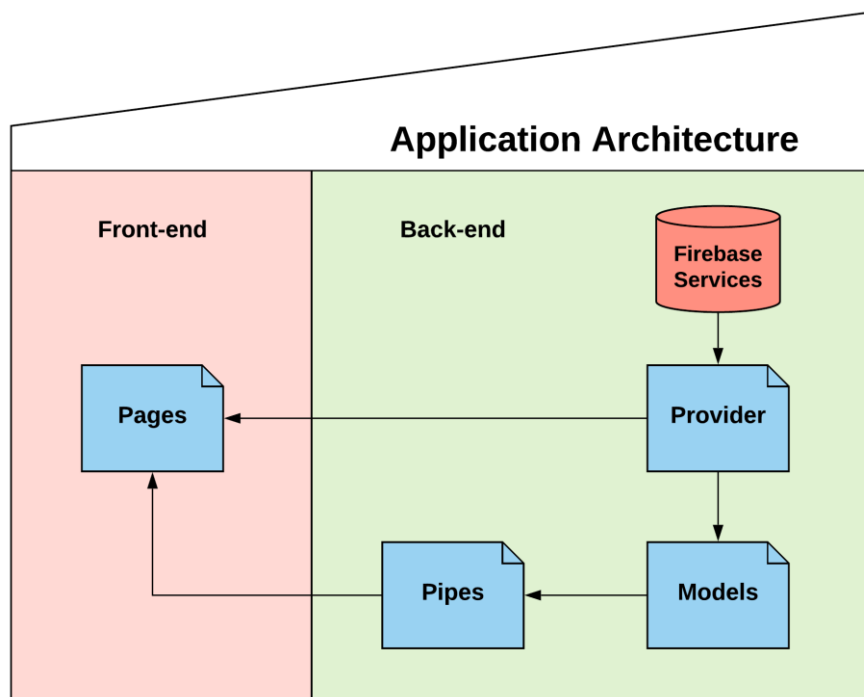
Além dos algoritmos e das estruturas de dados da computação; o *design* e a especificação da estrutura geral do sistema emergem como um novo tipo de problema. As questões estruturais incluem organização total e estrutura de controle global; protocolos de comunicação, sincronização e acesso a dados; designação de funcionalidade a elementos de design; distribuição física; composição de elementos de design; escalação e desempenho; e seleção entre as alternativas de design. (GARLAN, SHAW, 1993, p. 21)

A arquitetura, nessa etapa, é definida por meio de uma visão de implementação que contém uma visão geral do modelo de implementação e sua organização em termos de módulos em pacotes e camadas.

Neste projeto foi adotada a arquitetura baseada em rotas, sendo que a aplicação tem como base uma aplicação *web* e é possível o funcionamento do sistema por meio de mudanças de componentes do sistema através de rotas.

Como pode ser observado na figura 10, as camadas da aplicação serão divididas em *pipes*, *models*, *providers* e *pages*.

Figura 10 – Diagrama de representação da arquitetura



Fonte: do Autor (Lucidchart)

Responsável por manter os componentes de serviço do sistema que fazem comunicação com o Firebase, o *provider* (fornecedor/provedor) é a camada onde serão implementados os serviços de interação com o banco de dados, autenticação e funções que serão comuns a toda aplicação, para que não seja implementada cada regra de negócio em uma página diretamente, o *provider* é responsável por essa divisão de regras, deixando as páginas mais leves, com menos código e ainda sendo possível reaproveitar as funções em qualquer página da aplicação.

Na camada *models* são criados os atores que fazem parte da aplicação (aluno, motorista, rota) e suas respectivas características como nome, endereço, CPF, escola, etc. Esta camada tem o papel de servir como referência quando é necessário manipular os dados de um dos atores, por exemplo. A camada *pipes* pode instanciar estes objetos e armazenar cada informação em seus atributos para manipular estes dados dentro da aplicação, podendo

por exemplo criar uma lista de alunos, motoristas ou rotas e fazer uma busca, classificação ou ordenação dos dados.

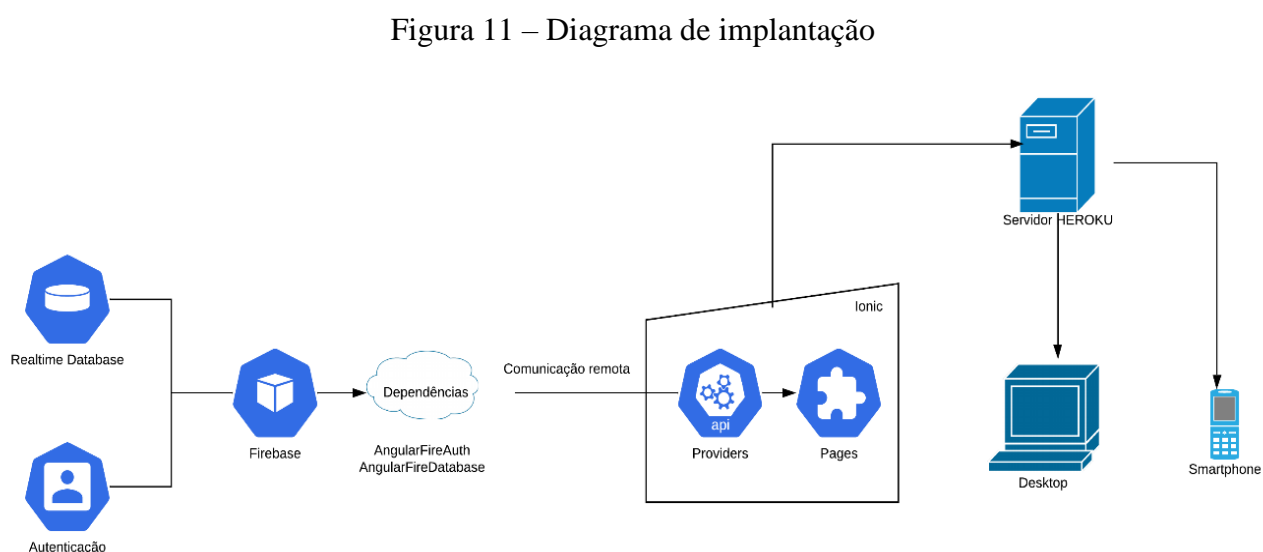
Segundo Adriano (2017), as *pipes* do Angular são uma maneira de realizar transformações no front-end. Com ela é possível criar funções ou filtros, que podem ser utilizadas em qualquer parte do *template* do projeto. Esta camada, presente neste projeto, serve para gerenciar funções de manipulação de dados para o front-end, como por exemplo, fazer a busca de alunos por nome, classificar os dados por ordem alfabética, por nome da escola, por endereço, etc.

A camada *pages* (páginas), fica encarregada de manter os componentes visuais do sistema, como as telas de usuário (*front-end*) que é escrito em HTML e também os métodos escritos em *TypeScript* que executam os procedimentos necessários para alimentar as páginas da aplicação.

3.4.2. Diagrama de implantação

Para demonstrar o processo de implantação da aplicação, utiliza-se um diagrama UML, que é um acrônimo para o termo em inglês *Unified Modeling Language*.

O diagrama que estabelece o fluxo de distribuição entre *back-end* e *front-end* da aplicação é o diagrama de implantação. Como pode ser observado na figura 11 a seguir, os serviços de banco de dados e autenticação do *Firebase* são chamados para a aplicação por meio de dependências instaladas e os dados podem ser manipulados dentro do bloco de *front-end* do sistema.



Fonte: do Autor (Criado no LucidChart)

Para a implantação deste projeto foi definido que o sistema será distribuído por meio do serviço Heroku, que segundo Adamatti (2017) “é um *PaaS*, Plataforma como serviço. Isso significa que você pode fazer *deploy* de seus serviços sem se preocupar com *configs* de *hardware* e sistema operacional.”

Com o Heroku é possível hospedar e implantar o sistema de forma gratuita (até certo limite de acessos simultâneos) em uma infraestrutura externa e utilizá-lo no *desktop* e nos *smartphones*.

3.4.3. Desenvolvimento UH001 – Login de Usuário

Após a arquitetura e o ambiente de desenvolvimento estarem configurados, se tornou possível a implementação do primeiro requisito do sistema que é a tela de *login* de usuário.

Conforme especificado no requisito (APÊNDICE C) e no diagrama de caso de uso (APÊNDICE B), o sistema possui dois tipos de usuário, gestor, que é o responsável pela coordenação do setor de transporte escolar do município e motorista, pessoa encarregada pela condução dos veículos do transporte escolar.

Na tabela 2, a seguir, é indicado as permissões e responsabilidades que cada um destes tem dentro do sistema.

Tabela 2 – Permissões de usuário

	Gestor	Motorista
Realizar login no sistema	SIM	SIM
Visualizar rotas	SIM	SIM
Visualizar lista de alunos	SIM	SIM
Cadastrar aluno	SIM	NÃO
Cadastrar rotas	SIM	NÃO
Gerar relatórios	SIM	NÃO
Iniciar viagem no sistema	NÃO	SIM
Registrar ocorrência de viagem	NÃO	SIM

Fonte: do Autor

Utilizando a API de autenticação do *Firebase*, foi possível criar a tela de *login* de usuário, sendo necessário a utilização de *e-mail* e senha para acesso. Com a classe *AngularFireAuth*, que faz parte do pacote de dependências do *AngularFire2*, foi necessário

apenas criar os métodos de *signIn()* e *signOut()* – que representam entrar e sair do sistema respectivamente – passando o objeto *User* como parâmetro, este que por sua vez, possui o *e-mail* e senha do usuário para autenticação.

O objeto *Observable* do tipo *firebase.user* tem como finalidade verificar e monitorar na API de autenticação se o usuário é válido e continua autenticado e autorizado conforme regras estabelecidas pela aplicação. No caso deste projeto, o usuário se mantém autenticado mesmo finalizando a aplicação, sendo necessário que o usuário se desconecte do sistema para voltar a tela de *login*. A figura 12 a seguir representa o código criado para implementação do *provider* de autenticação do sistema.

Figura 12 – Classe AuthProvider

```

6
7 @Injectable()
8 export class AuthProvider {
9
10   user: Observable<firebase.User>;
11
12   constructor(private angularFireAuth: AngularFireAuth) {
13     this.user = angularFireAuth.authState;
14   }
15
16   signOut(){
17     return this.angularFireAuth.auth.signOut();
18   }
19
20   signIn(user: User){
21     return this.angularFireAuth.auth.signInWithEmailAndPassword(user.email, user.password);
22   }
23
24 }
25

```

Fonte: do Autor (Adaptado do Visual Studio Code)

3.5. Sprint 3 – Desenvolvimento UH002, UH003, UH004 e UH005

A Sprint 3 foi planejada para o desenvolvimento do CRUD, que, segundo NASCIMENTO (2018), é o acrônimo de *Create*, *Read*, *Update* e *Delete* na língua Inglesa, para as quatro operações básicas utilizadas em bases de dados ou em interface para utilizadores para criação, consulta, atualização e destruição de dados.

O CRUD é a base principal do sistema, pois contempla os 4 requisitos que foram trabalhados nesta *Sprint*, sendo este um código em comum que poderá ser utilizado como base para a manipulação de dados de alunos e rotas dentro do sistema.

Como fora descrito na definição da arquitetura, os componentes, páginas, e serviços se comunicam e são chamados por meio de rotas, que são definidas por caminhos (PATH) e passados como parâmetros para que a API do *Firebase* identifique que tipo de dado será salvo, onde será salvo, onde será exibido e utilizado pela aplicação.

Uma das vantagens de se desenvolver um *software* utilizando *frameworks* pode ser observado na figura 13 a seguir, pois é necessário apenas um embasamento lógico sobre a forma como um CRUD funciona e chamar os métodos e procedimentos já previamente existentes no *framework* para a criação deste bloco de código que tem como função buscar todos os registros do banco de dados para aquele caminho especificado, e também buscar um registro específico passando como parâmetro o caminho e a chave identificadora do registro desejado.

Figura 13 – Classe AlunosProvider

```
4  @Injectable()
5  export class AlunosProvider {
6      private PATH = 'alunos/';
7
8      constructor(private db : AngularFireDatabase) {
9
10     }
11
12     getAll(){
13         return this.db.list(this.PATH)
14             .snapshotChanges()
15             .map(changes => {
16                 return changes.map(c => ({
17                     key: c.payload.key, ...c.payload.val()
18                 }));
19             });
20     }
21
22     get(key : string){
23         return this.db.object(this.PATH + key)
24             .snapshotChanges()
25             .map(c => {
26                 return { key: c.key, ...c.payload.val()}
27             });
28     }
29 }
```

Fonte: do Autor (Adaptado do Visual Studio Code)

A função *getAll()* retorna todos os registros e seus respectivos dados, dentro do caminho especificado (/alunos) e a função *get(key : string)* busca um registro específico dentro do banco de dados utilizando um identificador como parâmetro para busca.

Na continuação do código se encontra outras duas funções que compõem o CRUD deste projeto, são elas a função *save* e *remove* que representam os métodos de salvar e remover dados respectivamente. É importante ressaltar que a função *save* possui procedimentos que pode salvar dados no banco e também atualizar dados no banco, sendo assim, contempla duas operações do CRUD, *create* e *update*, como pode ser observado na figura 14 a seguir.

Por fim, foi realizado a comunicação do CRUD com o desenvolvimento do *front-end* desta funcionalidade, onde apresenta-se um formulário para inserção de dados sobre os alunos e o botão de salvar que vai disparar a função *save* da classe *AlunosProvider*, enviando os dados para o banco de dados da aplicação. O resultado da tela de cadastro de aluno pode ser observado na figura 15.

Figura 14 – Classe *AlunosProvider*, funções *save* e *remove*

```
30     save(aluno : any){
31         return new Promise((resolve, reject) => {
32             if(aluno.key){
33                 this.db.list(this.PATH)
34                     .update(aluno.key, {
35                     name: aluno.name, tel: aluno.tel})
36                     .then(() => resolve())
37                     .catch((e) => reject(e));
38             } else {
39                 this.db.list(this.PATH)
40                     .push({name: aluno.name, tel: aluno.tel})
41                     .then(() => resolve());
42             }
43         });
44     }
45
46     remove(key : string){
47         return this.db.list(this.PATH).remove(key);
48     }
49
50 }
51
```

Fonte: do Autor (Adaptado do Visual Studio Code)

Figura 15 – Protótipo de tela “Novo Aluno”

O protótipo da tela "Novo Aluno" apresenta o seguinte layout:

- Barra de status superior com ícones de Wi-Fi, sinal de rede e bateria, e o horário 15:10.
- Barra de título com uma seta para trás e o texto "Novo Aluno".
- Formulário com os seguintes campos:
 - Nome Completo *
 - Endereço/Região *
 - CPF ou RG do aluno*
 - Sexo (menu suspenso)
 - Data de Nascimento * (com ícone de calendário)
 - Nome do Responsável *
 - Telefone do Responsável * (campo de texto)
 - Escola (menu suspenso)
 - Observações (campo de texto)
- Dois botões de ação: "SALVAR" (em azul) e "LIMPAR" (em cinza).
- Barra de navegação inferior com ícones de voltar, home e recentes.

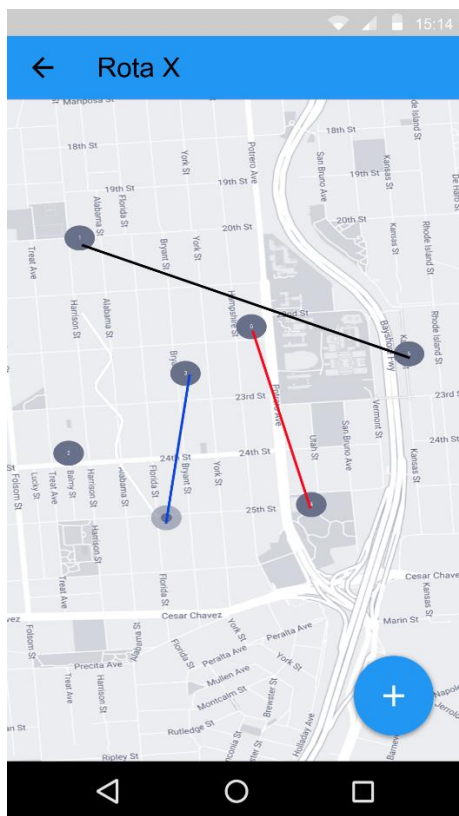
Fonte: do Autor

3.6. Sprint 4 – Desenvolvimento UH006 e UH007

Na Sprint 4, com base no requisito Manter Rotas (UH006) e Registrar Viagem (UH007), foi desenvolvido um protótipo da funcionalidade que necessita de uma conexão com a API do Google Maps, sendo que este requisito mostra o mapa da região rural e urbana do município e pode-se cadastrar uma rota, selecionando geograficamente os pontos de início e fim da rota no mapa, ou inserindo manualmente as coordenadas da rota, sendo mais prático a segunda opção, sendo que a secretaria de Educação possui os dados das coordenadas de cada rota existente no transporte escolar.

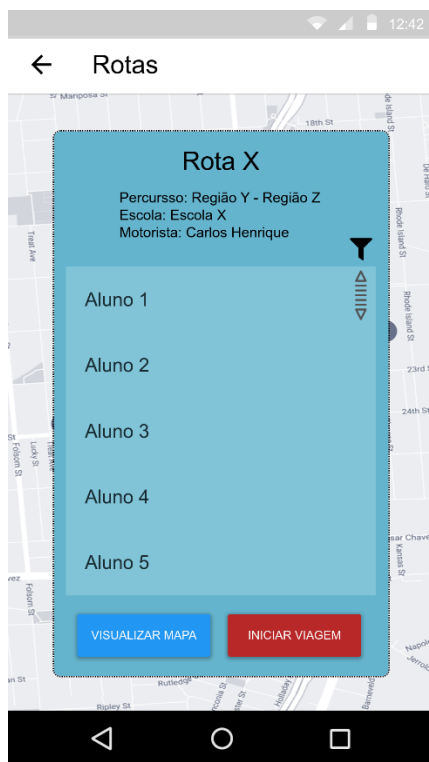
A figura 16 demonstra a tela de exibição das rotas (mapa meramente ilustrativo), indicando com linhas coloridas no mapa, cada uma das rotas e também com a opção de clicar sobre uma rota e expandir as informações sobre ela, além de exibir a funcionalidade de iniciar viagem, como mostra a figura 17.

Figura 16 – Protótipo de tela “Rotas”



Fonte: do Autor

Figura 17 – Protótipo de tela “Rota selecionada”



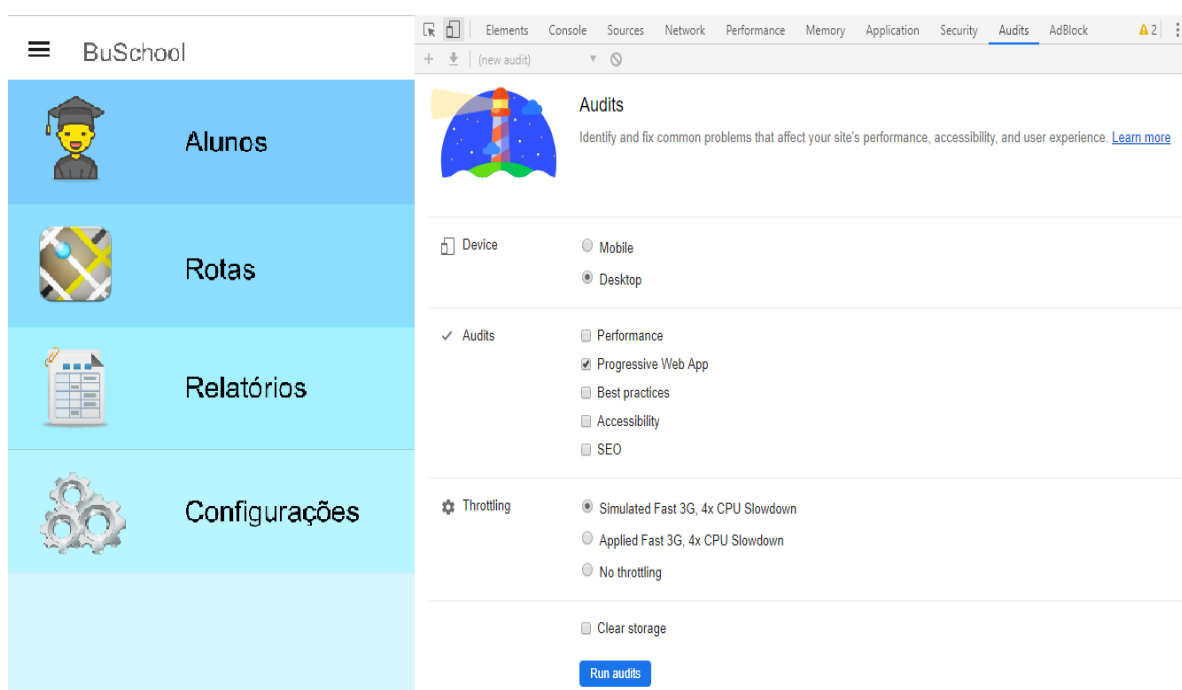
Fonte: do Autor

3.7. Sprint 5 - Aplicando os recursos de PWA

Para uma aplicação poder ser chamada de PWA é preciso adicionar alguns recursos ao código para resolver algumas dependências que os navegadores exigem para uma aplicação executar todas as particularidades de um sistema em PWA.

Estes recursos podem ser conferidos por meio de uma ferramenta de auditoria fornecida pela Google, no navegador Chrome. Para verificar estas auditorias basta, com o sistema sendo executado, abrir as ferramentas de desenvolvedor (atalho F12), e ir até a aba *Audits*, como exibido na figura abaixo.

Figura 18 – Auditoria do sistema sendo executado no navegador Google Chrome.



Fonte: Adaptado do navegador Google Chrome

Com a aplicação de exemplo executando ao lado, uma simples aplicação de lista de tarefas desenvolvida utilizando Ionic para utilizar como exemplo, basta escolher em qual dispositivo quer executar, *mobile* ou *desktop*, marcar a opção *Progressive Web App*, e executar a auditoria. Os resultados serão exibidos a seguir.

Figura 19 – Resultado da auditoria do sistema

Progressive Web App

These checks validate the aspects of a Progressive Web App. [Learn more.](#)

Fast and reliable

1	Page load is fast enough on mobile networks	✓
2	Current page does not respond with a 200 when offline	✗
3	start_url does not respond with a 200 when offline No usable web app manifest found on page.	✗

Installable

4	Uses HTTPS	✓
5	Does not register a service worker that controls page and start_url	✗
6	Web app manifest does not meet the installability requirements Failures: No manifest was fetched.	✗

PWA Optimized

7	Does not redirect HTTP traffic to HTTPS	✗
8	Is not configured for a custom splash screen Failures: No manifest was fetched.	✗
9	Does not set an address-bar theme color Failures: No manifest was fetched, No ` <meta >`="" found.<="" name="theme-color" tag="" td=""/> <td>✗</td>	✗
10	Content is not sized correctly for the viewport The viewport size is 460px, whereas the window size is 1366px.	✗
11	Has a <code><meta name="viewport"></code> tag with width or initial-scale	✓
12	Does not provide fallback content when JavaScript is not available The page body should render some content if its scripts are not available.	✗

Fonte: Adaptado do navegador Google Chrome

Nesta execução de auditoria, a aplicação ainda se encontra sem qualquer recurso de PWA, tendo como resultado diversos erros quanto a estrutura de uma aplicação em PWA, como pode ser observado na imagem os itens mais importantes:

- Item 2: A página não responde quando está *offline*.
- Item 3: A página não possui uma *web app manifest*, por isso não responde quando está offline
- Item 5: Não há um *service worker* registrado para controlar a página
- Item 6: O *web app manifest* não está configurado com os requisitos de estabilidade.

Os outros itens presentes na auditoria são recursos para tornar o PWA otimizado e, portanto, podem ser atendidos posteriormente. Para criar uma aplicação que atenda

minimamente os requisitos de um PWA não é necessário cumprir os requisitos de otimização, apesar de ser indicado seguir também estas recomendações.

4.6.1. Criando o *Service Worker*

Conforme exigido pela auditoria da aplicação, um PWA precisa de um *service worker*, que é um *script* executado em segundo plano que abre portas para recursos do dispositivo, melhorando a implementação tradicional da *web* e oferecendo uma experiência de confiabilidade e desempenho equivalentes a uma aplicação nativa.

Para adicionar o *service worker* ao projeto basta utilizar o CLI no terminal e executar o seguinte comando: ***ng add @angular/pwa***.

Basicamente este comando cria toda a estrutura do *service worker* dentro do projeto, evitando a necessidade de criar toda a estrutura unicamente exclusiva para o projeto, pois um *service worker* possui características comuns para ser aplicadas a qualquer tipo de projeto.

Ao executar este comando, são inferidos os seguintes recursos:

- Adiciona o pacote *@angular/service-worker* ao projeto;
- Permite o suporte à construção do *service worker*;
- Importa e registra o *service worker* no módulo da aplicação;
- Inclui o *link* no arquivo *index* e adiciona no arquivo *manifest*;
- Cria o arquivo de configuração do *service worker*;
- Especifica os comportamentos de armazenamento em *cache* e outras configurações.

4.6.2. Manifesto

Segundo De Abreu (2018), o manifesto é um arquivo JSON que disponibiliza a capacidade de controlar a aparência da aplicação para o usuário nas áreas onde ele pode ver aplicativos, direcionar o conteúdo que o usuário pode acessar e como pode acessar.

O manifesto criado para o PWA disponibiliza:

- Adicionar a aplicação a tela inicial do dispositivo;
- Ser iniciado na tela inteira sem a presença da barra de URL;

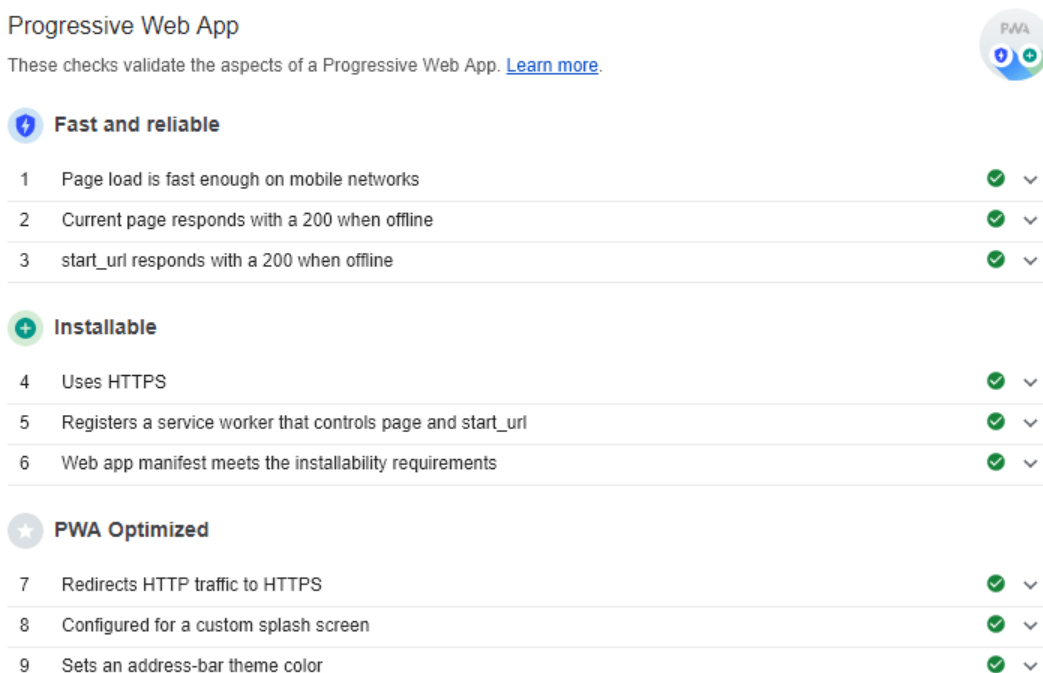
- Controlar a orientação da tela;
- Define uma experiência de inicialização com tela de apresentação e um tema para o site;
- Verificar se a aplicação foi iniciada da tela inicial ou de uma barra URL.

4.6.3. Auditoria final

Após concluir as etapas de configuração do projeto para as características do PWA, é necessário fazer a *build* do projeto e então executar novamente a auditoria da aplicação para verificar se as exigências foram atendidas.

Executando o comando `ng build --prod --base-href`, o projeto vai ser compilado e finalizado. Posteriormente, basta executar novamente a auditoria no navegador para obter os seguintes resultados, como mostra a figura abaixo:

Figura 20 – Resultado final da auditoria



Progressive Web App		PWA	
These checks validate the aspects of a Progressive Web App. Learn more.			
Fast and reliable			
1	Page load is fast enough on mobile networks	✓	▼
2	Current page responds with a 200 when offline	✓	▼
3	start_url responds with a 200 when offline	✓	▼
Installable			
4	Uses HTTPS	✓	▼
5	Registers a service worker that controls page and start_url	✓	▼
6	Web app manifest meets the installability requirements	✓	▼
PWA Optimized			
7	Redirects HTTP traffic to HTTPS	✓	▼
8	Configured for a custom splash screen	✓	▼
9	Sets an address-bar theme color	✓	▼

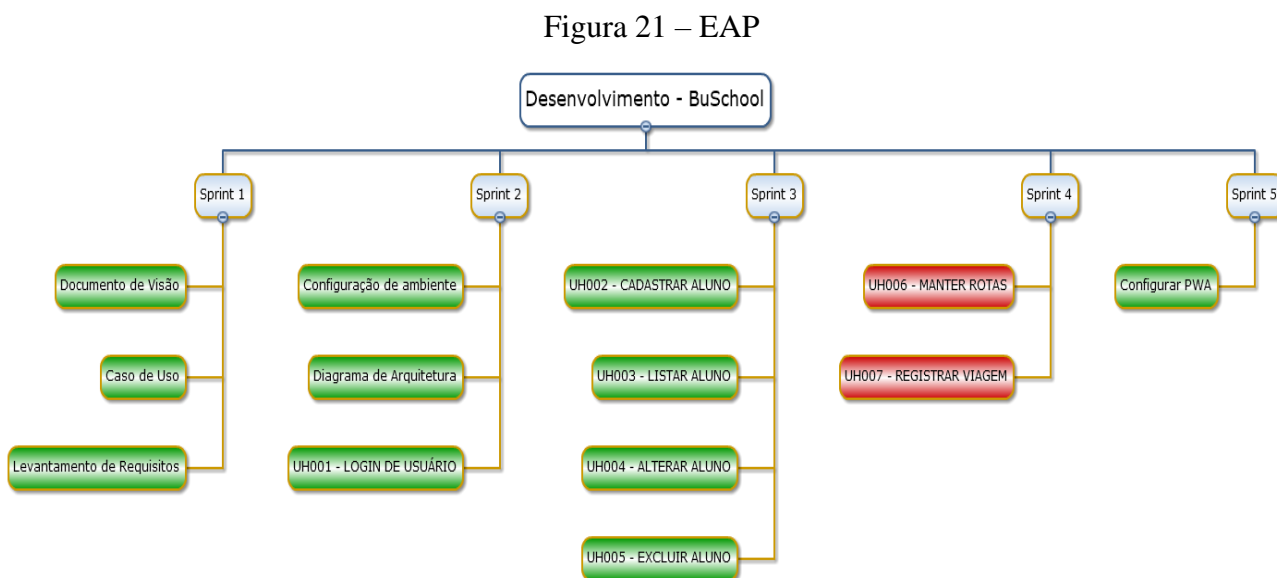
Fonte: Adaptado do navegador Google Chrome

Como resultado, o navegador agora considera que todas as exigências para se denominar uma aplicação como PWA foram atendidas e os recursos predominantes de uma aplicação em PWA que solucionam as exigências do projeto em questão podem agora ser

utilizadas, como a responsividade ao utilizar a aplicação em diferentes tipos de dispositivos (*web* e *mobile*) e a característica de ser utilizável na ausência de conexão de rede, armazenando dados em *cache*.

3.8. Resultados

Devido a impedimentos no desenvolvimento, como prazo restrito e conhecimento superficial sobre o desenvolvimento com Ionic, que gerou dificuldades na implementação, algumas funcionalidades do sistema foram planejadas e documentadas, porém, ainda não implementadas. A Figura 21 a seguir representa uma estrutura analítica de projeto (EAP) que demonstra o que foi planejado e executado (cor verde), e o que foi planejado, mas ainda não executado (cor vermelha).



www.wbstool.com

Fonte: Do autor (Wbstool)

A integração de uma API do Google Maps com o sistema é o ponto inicial para o desenvolvimento dos requisitos UH006 e UH007. Devido ao prazo restrito para entrega do projeto e por conta do pouco referencial teórico para auxiliar a implementação desta API no Ionic houve dificuldades para prosseguir com o desenvolvimento do sistema.

A configuração do PWA foi executada a partir do que fora desenvolvido até o momento da Sprint 5, mesmo que o sistema não se apresentasse totalmente concluído, pois é importante que se obtenha o funcionamento da aplicação em PWA para analisar e concluir um dos objetivos do projeto.

4. Trabalhos Futuros

O sistema sugerido neste projeto gerencia um fluxo de dados sobre as viagens dos veículos do transporte escolar rural do município, sendo importante que sejam gerados relatórios dentro do sistema utilizando estes dados e o desenvolvimento do requisito Gerar Relatórios é a funcionalidade para trabalhos futuros do projeto.

Neste requisito deverá ser trabalhado o levantamento de dados obtidos durante as viagens como distância percorrida diariamente, ocorrências sobre as viagens, relatórios sobre os alunos de cada escola e quantidade de combustível consumido diariamente, semanalmente ou mensalmente. Estes relatórios poderão ainda ser exportados para arquivos externos como *.doc* (Microsoft Word) ou *.xls* (Microsoft Excel), somente sendo possível gerar relatórios se o usuário autenticado no sistema for um gestor da Secretaria de Educação do município.

Outro diferencial a ser desenvolvido em trabalhos futuros é a geração de um arquivo externo das rotas para GPS, de forma que o sistema forneça o arquivo de GPS para o gestor, além de também aceitar que seja inserida uma nova rota a partir de um arquivo importado para o sistema a partir do GPS utilizado pelo gestor.

5. Considerações Finais

O projeto de TCC, desde a sua concepção na disciplina de Métodos e Técnicas de Pesquisa Científica até a sua conclusão neste momento, contempla um processo de muita importância e aprendizado para o acadêmico, que utiliza de todo o conhecimento até então adquirido para construir um processo de pesquisa que determina o desenvolvimento pessoal do acadêmico dentro do curso.

O desenvolvimento deste projeto foi de grande ganho pessoal, tendo em vista que reforça todo um processo de construção de um software, desde a concepção da ideia, a criação de documentação, casos de uso, levantamento de requisitos, desenvolvimento do código, e todo um processo que se espelha na prática como é o desenvolvimento de um *software*.

O uso de metodologias ágeis reforçou a importância de se conhecer estas ferramentas para se ter sucesso no mercado. A produtividade do processo de desenvolvimento deste projeto também é consequência do uso de metodologias ágeis como a divisão de tarefas com o Kanban, o trabalho sendo desenvolvido por meio de *sprints* que determinam um processo contínuo de desenvolvimento, contemplando e delegando atividades na equipe para se obter um processo com melhor desempenho e com tempo reduzido.

Utilizando o *framework* Ionic foi possível, em conjunto com o Firebase, obter maior produtividade na criação de um protótipo do sistema, devido ao fato de que os *frameworks* disponibilizam funcionalidades já implementadas sendo necessário apenas utilizá-las, sendo assim, o foco fica apenas nas regras de negócio do sistema.

É notável que o processo de desenvolvimento de um *software* engloba diversos métodos e procedimentos desde a concepção do projeto até a sua primeira entrega. Desenvolver um projeto é muito mais que apenas codificar, sendo que a parte do processo de desenvolvimento é muito importante para se obter um produto de qualidade. Foi perceptível, ao desenvolver este projeto, que construir um *software* de qualidade é consequência de um processo bem estruturado, definido e delegado.

Referências Bibliográficas

ADAMATTI, Marcelo. 2017. **O básico sobre heroku**. Disponível em: <<https://adamatti.github.io/blog/git/2017/06/04/heroku.html>>. Acesso em: 28 out. 2019.

ADRIANO, Thiago. 2017. **Angular: Criação de Pipes**. Disponível em: <<https://imasters.com.br/desenvolvimento/angular-criacao-de-pipes>>. Acesso em: 28 out. 2019.

BEGGIORA, Helito. **Como entrar no modo desenvolvedor do Google Chrome**. TechTudo, São Paulo, 2015. Disponível em: <https://www.techtudo.com.br/dicas-e-tutoriais/noticia/2015/01/como-entrar-no-modo-desenvolvedor-do-google-chrome.html>. Acesso em: 13 abr. 2019.

BERNARDO, Kleber. **Kanban: do início ao fim! Estória de usuário. Você saberia contar?**. [S. l.], 8 dez. 2014. Disponível em: <https://www.culturaagil.com.br/kanban-do-inicio-ao-fim/>. Acesso em: 28 out. 2019.

BERNARDO, Kleber. **Como funciona o Scrum?** 30 nov. 2015. Disponível em: <https://www.culturaagil.com.br/como-funciona-o-scrum/>. Acesso em: 14 out. 2019.

BRASIL. **Constituição** (1988). Constituição da República Federativa do Brasil. Brasília, DF: Senado Federal: Centro Gráfico, 1988.

CAMDEN, R.K., 2015. **Apache Cordova in Action**. Manning Publications Co.

DA SILVA, Ewerthon ; SOTTO, Eder. **A utilização do ionic framework no desenvolvimento de aplicações híbridas em arquitetura orientada a serviço**. Interface Tecnológica, São Paulo, p. 97-108, 7 dez. 2018. Disponível em: <https://revista.fatectq.edu.br/index.php/interfacetecnologica/article/download/333/216/>. Acesso em: 12 jun. 2019.

DE SOUZA, Antonio Anderson. **O que é PWA e porque isso pode aumentar seus resultados mobile**. [S. l.], 9 ago. 2017. Disponível em: <https://vizir.com.br/2017/08/o-que-e-pwa-progressive-web-app-porque-isso-pode-aumentar-seus-resultados-mobile/>. Acesso em: 12 jun. 2019

DE ABREU, Lucas Henrique. **Progressive Web App: Aplicando as técnicas de PWA em Angular 6**. [S. l.], 19 maio 2018. Disponível em: <https://medium.com/@lucashenriquedeabreu/progressive-web-app-aplicando-as-t%C3%A9cnicas-de-pwa-em-angular-6-87ee22444d19>. Acesso em: 11 maio 2019.

FIREBASE. Firebase. 2019. **Documentação do Firebase**. Disponível em:<<https://firebase.google.com/docs/>>. Acesso em: 14 out. 2019.

GASPAROTTO, Henrique. **Xamarin, Ionic e Cordova: Conheça o que são e as principais diferenças**. Disponível em: <https://www.devmedia.com.br/xamarin-ionic-e-cordova-conheca-o-que-sao-e-as-principais-diferencas/37690>. Acesso em: 12 jun. 2019.

GAUNT, Matt. **Introdução aos service workers**. Web Fundamentals, Google. Disponível em: <https://developers.google.com/web/fundamentals/primers/service-workers/?hl=pt-br>. Acesso em: 03 maio. 2019

GESTA: Galeria de Estudos e Avaliação de Iniciativas Públicas. Disponível em: <http://gesta.org.br/tema/engajamento-escolar/>. Acesso em: 07/11/2018.

GROFFE, Renato Jose. 2013. **Modelagem de sistemas: uma visão geral**. Disponível em: <https://www.devmedia.com.br/modelagem-de-sistemas-atraves-de-uml-uma-visao-geral/27913>. Acesso em: 14 out. 2019.

Google Developers: Progressive Web Apps. Disponível em: <https://developers.google.com/web/progressive-web-apps/> Acesso em: 06/04/2019.

GSMA Intelligence: Definitive data and analysis for the mobile industry. Disponível em: <https://www.gsmainelligence.com/>. Acesso em: 27/03/2018.

INVETTI: Scrum x Kanban. Disponível em: <https://www.inventti.com.br/scrum-x-kanban/>. Acesso em: 29/10/2019.

JAQUES, Rafael. **O que é um Framework? Para que serve?** PhPhit, Rio Grande do Sul, 2016. Disponível em: <http://www.phpit.com.br/artigos/o-que-e-um-framework.phpit>. Acesso em: 13 abr. 2019.

JUNIOR, Carlos. **Scrum: o que é sprint e como executá-lo?** 31 mai. 2017. Disponível em: <https://www.projectbuilder.com.br/blog/scrum-o-que-e-sprint-e-como-executa-lo/>. Acesso em: 14 out. 2019.

JUSTEN, Willian. **Como fazer seu site funcionar offline com PWA**. [S. l.], 26 mar. 2018. Disponível em: <https://willianjusten.com.br/como-fazer-seu-site-funcionar-offline-com-pwa/>. Acesso em: 12 jun. 2019.

LOPES, Andrei. 2019. **O que é NPM**. Disponível em: <https://www.hostinger.com.br/tutoriais/o-que-e-npm>. Acesso em: 16 out. 2019.

MELLER, William. Scrum: **O que são User Stories**. [S. l.], 16 nov. 2016. Disponível em: <https://sitecampus.com.br/scrum-o-que-sao-user-stories/>. Acesso em: 14 maio 2019.

NASCIMENTO, Thiago. 2018. **Usabilidade em interfaces CRUD: o guia completo**. Disponível em: <http://thiagonasc.com/usabilidade/usabilidade-interfaces-crud> >. Acesso em: 17 out. 2019.

NATO, Science. **NATO Science and Technology Organization**. 1969. Disponível em: https://www.nato.int/cps/en/natohq/topics_88745.htm?selectedLocale=en. Acesso em: 19 jun. 2018.

PÁDUA, Wilson de. **Engenharia de software: fundamentos, métodos e padrões**. 3.ed. - Rio de Janeiro : LTC, 2009.

PLANTIER, Renato Duarte. **O primeiro software criado**: História da Informática. 2013 Disponível em: <<http://tecnologia.culturamix.com/tecnologias/o-primeiro-software-criado-historia-da-informatica>>. Acesso em: 23/04/2018.

PRESSMAN, Roger S. **Engenharia de Software**: Uma Abordagem Profissional. 8ª Ed. AMGH Editora Ltda., 2016.

SAMBATECH, Blog. **Metodologias ágeis: o que são e quais os principais tipos**. [S. l.], 11 abr. 2019. Disponível em: <https://sambatech.com/blog/insights/metodos-ageis/>. Acesso em: 12 jun. 2019.

SIGE: Sistema de Gerenciamento Escolar. Disponível em: <<http://sige.seduc.go.gov.br/sige/default.asp>>. Acesso em: 21/05/2018

SILVA, Marcio Andrade. **A importância do levantamento de requisitos no sucesso dos projetos de software**. Linha de Código, Disponível em: <http://www.linhadecodigo.com.br/artigo/1685/a-importancia-do-levantamento-de-requisitos-no-sucesso-dos-projetos-de-software.aspx>. Acesso em: 14 maio 2019.

UNICEF. **Cenário da exclusão escolar no brasil**. Brasil: 2017. Disponível em: <https://www.unicef.org/brazil/pt/cenario_exclusao_escolar_brasil.pdf>. Acesso em: 10 nov. 2018.

VIEIRA, Rodrigo. **UML—Diagrama de Casos de Uso**. [S. l.], 12 dez. 2015. Disponível em: <https://medium.com/operacionalti/uml-diagrama-de-casos-de-uso-29f4358ce4d5>. Acesso em: 12 jun. 2019.

APÊNDICES

APÊNDICE A – DOCUMENTO DE VISÃO

BuSchool

Documento de Visão

Introdução

O documento de visão define o escopo de alto nível e o propósito de um programa, produto ou projeto. Uma instrução clara do problema, solução proposta e os recursos de alto nível do produto ajudam a estabelecer expectativas e a reduzir riscos. Tem grande relevância durante as primeiras fases, permitindo a captura de todas as perspectivas que o sistema pode abranger. Pretende-se que sirva como ferramenta de auxílio, a evitar alguns dos problemas mais custosos com que as pessoas envolvidas no projeto poderão ter que se confrontar. Esta ajuda é proporcionada através da divulgação do conteúdo deste a todos aqueles que estejam integrados no sistema.

Descrição do Problema

O problema do	<i>controle de transporte escolar rural do município de Silvânia é que se encontra em um processo totalmente manual e trabalhoso. Dispondo de 35 ônibus, o município atende 1.204 alunos alocados em 36 rotas, percorrendo uma média diária de 5.348 km. A gestão de cada ônibus para cada região do município e a alocação de alunos dentro de uma rota têm se tornando insustentável de se gerir manualmente</i>
afetando	<i>os responsáveis por controlar toda a gestão de rotas e ônibus do transporte escolar do município, além de todo estudante que necessita do transporte</i>
o impacto disso	<i>é uma gestão feita de forma manual e inviável de se manter caso aumente a demanda de estudantes para o uso do transporte</i>
uma solução bem-sucedida seria	<i>Desenvolver um software para gerir as rotas do transporte escolar, que atualmente é executado de forma manual</i>

Descrição do Produto

Para	<i>Secretaria de Educação do Município de Silvânia</i>
Quem	<i>necessita gerenciar rotas de transporte escolar rural do município.</i>
O BuSchool	<i>sistema de controle de rotas e gerenciamento de recursos do transporte escolar rural</i>
Que	<i>permite a gestão completa de toda a frota de ônibus escolares que transportam alunos moradores de zonas rurais, definindo a rota a ser seguida e também o motorista responsável.</i>
Diferentemente do	<i>processo atual que é burocrático e manual, o software permite um controle de horários e quilometragem percorrida em cada dia, podendo gerar relatórios para possíveis análises e melhorias no processo de transporte.</i>
Nosso produto	<i>se destaca por ser uma solução simples, prática e eficiente para controle do transporte escolar rural.</i>

Descrição dos Stakeholders

Resumo dos Stakeholders

Nome	Descrição	Responsabilidades
<i>Coordenador do transporte escolar</i>	<i>Pessoa responsável pelo controle das rotas do transporte</i>	<i>Garante que o projeto é uma necessidade pontual para que o gerenciamento das rotas escolares seja feito de forma eficiente, trazendo benefícios sociais para a população que utiliza o transporte.</i>
<i>Motorista</i>	<i>Pessoa responsável por conduzir o ônibus escolar</i>	<i>Descreve o projeto como uma ferramenta de grande contribuição para o controle das viagens, onde podem inserir a quilometragem de maneira simples e eficiente.</i>

Ambiente de Usuário

O Sistema será administrado por integrantes da secretaria de educação do município em questão, que por função tem de gerenciar toda a rota de veículos, cadastro de alunos e regiões. O sistema também será utilizado pelos motoristas dos veículos. O software deverá ser executado na web, para que seja utilizado especificamente dentro da secretaria pelos responsáveis, e também por dispositivos móveis, para utilização do sistema pelos motoristas e porventura, para atualização de dados cadastrais caso tenha que ser feito externamente à secretaria. O software necessita da utilização de outra aplicação, que é o Google Maps, utilizando a API fornecida pela Google, será possível adicionar um mapa completo do município para traçar as rotas.

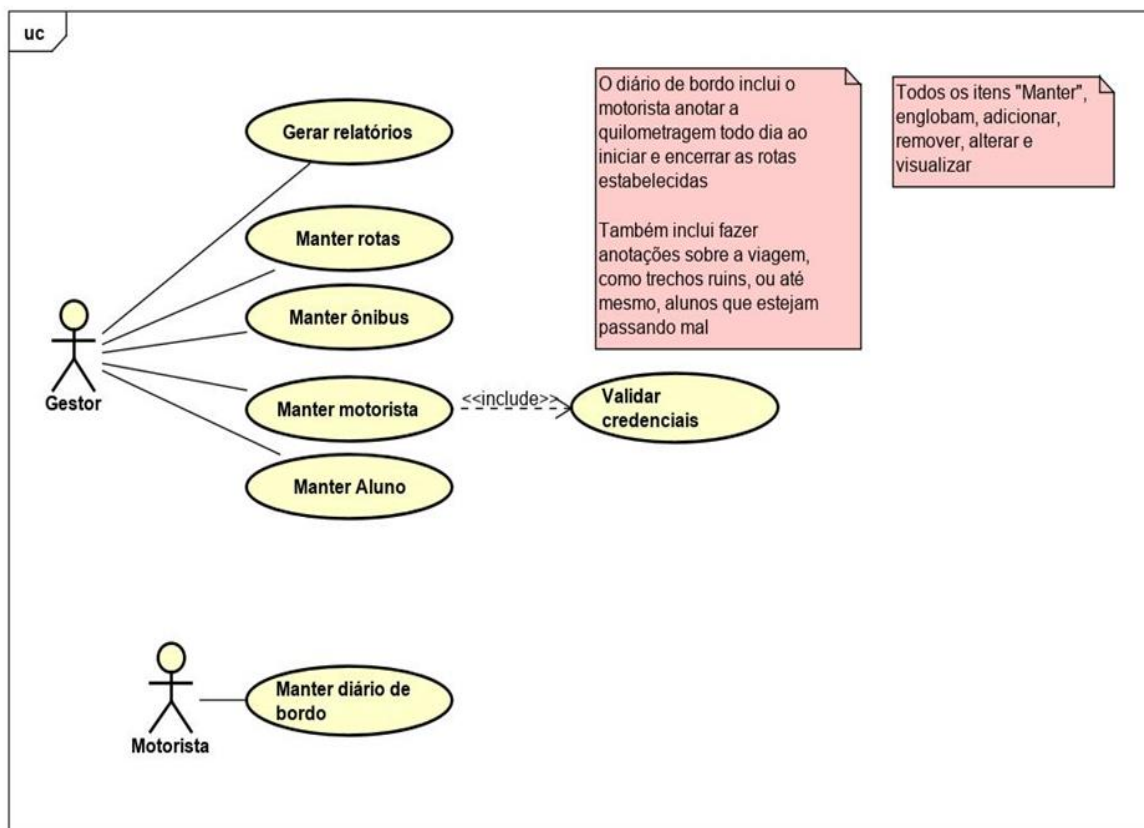
**Visão Geral do Produto
Necessidades e Recursos**

Necessidade	Prioridade	Recursos	Liberação Planejada
Realtime Database	Alta	Firebase	
Programação Web	Alta	Javascript	
Programação Multiplataformas	Alta	Ionic	
Mapas atualizados	Alta	Google Maps	

Outros requisitos do produto

Requisito	Prioridade	Liberação Planejada
Exportar dados para Excel	Media	
Exportar rotas para arquivo de GPS	Baixa	
Disponibilidade de serviço	Alta	
Desempenho	Baixa	
Manual do Usuário	Baixa	
Ajuda Online	Média	

APÊNDICE B – DIAGRAMA DE CASO DE USO



APÊNDICE C – UH001 LOGIN DE USUÁRIO

Visão Geral do Processo

Por meio deste requisito é possível que o gestor da secretaria ou o motorista entre no sistema utilizando suas credenciais cadastradas.

Atores/ Perfis do Processo

NOME DO PERFIL	TIPO DE PERFIL	DESCRIÇÃO DO PERFIL
Gestor	Interno	Usuário do sistema
Motorista	Interno	Usuário do sistema

Como demonstrar

Assim que abrir o sistema e não possuir nenhum usuário previamente já logado no sistema, esta funcionalidade irá se apresentar.

História de Usuário

HISTÓRIA DE USUÁRIO	DESCRIÇÃO	STATUS	PRIORIDADE								
UH. 001	Login de Usuário	A desenvolver	Essencial								
<p>Eu como: Usuário do tipo 'gestor' ou 'motorista' já cadastrado no sistema</p> <p>Quero: Realizar o login no sistema</p> <p>Para: Acessar todas as funcionalidades do sistema</p> <p style="text-align: center;">CRITÉRIOS DE ACEITAÇÃO</p> <table border="1"> <thead> <tr> <th>ITEM</th> <th>DETALHAMENTO</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>O sistema apresenta a tela "Realizar Login".</td> </tr> <tr> <td>2</td> <td>Usuário informa corretamente as credenciais do usuário já previamente cadastrado.</td> </tr> <tr> <td>3</td> <td>O sistema apresenta a tela inicial.</td> </tr> </tbody> </table>				ITEM	DETALHAMENTO	1	O sistema apresenta a tela " Realizar Login ".	2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.	3	O sistema apresenta a tela inicial.
ITEM	DETALHAMENTO										
1	O sistema apresenta a tela " Realizar Login ".										
2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.										
3	O sistema apresenta a tela inicial.										

Fluxo Alternativo

Se o usuário já estiver logado no sistema, de acordo com as regras estabelecidas na regra de negócio, o sistema apresentará diretamente a tela inicial da aplicação.

Fluxo de Exceção

FLUXO DE EXCEÇÃO	
ITEM	DETALHES
1	Caso as credenciais informadas sejam inválidas, informar que o email ou senha estão incorretos

Links e botões

NOME	TIPO	FUNÇÃO	REGRA
Entrar	Botão	Verificar as credenciais informadas no formulário e autenticar o usuário no sistema.	Habilitar o botão apenas quando todos os campos obrigatórios estiverem corretamente inseridos.
Esqueci minha senha	Botão	Acessa a página para recuperação de senha esquecida.	Sempre habilitado

Campos

NOME	TAMANHO	TIPO	DESCRIÇÃO
Email*	-	Texto	-
Senha*	-	Senha	-

*Campos obrigatórios

APÊNDICE D – UH002 CADASTRAR ALUNO

Visão Geral do Processo

Por meio deste requisito é possível que o gestor da secretaria cadastre os alunos no sistema.

Atores/ Perfis do Processo

NOME DO PERFIL	TIPO DE PERFIL	DESCRIÇÃO DO PERFIL
Gestor	Interno	O usuário deve estar autenticado como gestor para ter acesso à funcionalidade.

Como demonstrar

Acesso a tela de Login -> Fazer login no sistema como um usuário do tipo gestor -> Clicar em cadastro -> Clicar em alunos.

História de Usuário

HISTÓRIA DE USUÁRIO	DESCRIÇÃO	STATUS	PRIORIDADE														
UH. 002	Cadastrar Aluno	A desenvolver	Essencial														
<p>Eu como: Usuário do tipo 'gestor' já cadastrado no sistema</p> <p>Quero: Realizar o cadastro de um aluno</p> <p>Para: Manter no sistema a base de dados de todos os indivíduos que farão parte do processo de transporte escolar do município</p> <p style="text-align: center;">CRITÉRIOS DE ACEITAÇÃO</p> <table border="1"> <thead> <tr> <th>ITEM</th> <th>DETALHAMENTO</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>O sistema apresenta a tela "<u>Realizar Login</u>".</td> </tr> <tr> <td>2</td> <td>Usuário informa corretamente as credenciais do usuário já previamente cadastrado.</td> </tr> <tr> <td>3</td> <td>O sistema apresenta a tela inicial.</td> </tr> <tr> <td>4</td> <td>O usuário clica na opção "Cadastrar"</td> </tr> <tr> <td>4.1</td> <td>O sistema apresenta a tela de cadastro com as opções "Aluno", "Motorista" ou "Ônibus".</td> </tr> <tr> <td>5</td> <td>O usuário escolhe a opção "Aluno".</td> </tr> </tbody> </table>				ITEM	DETALHAMENTO	1	O sistema apresenta a tela " <u>Realizar Login</u> ".	2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.	3	O sistema apresenta a tela inicial.	4	O usuário clica na opção "Cadastrar"	4.1	O sistema apresenta a tela de cadastro com as opções "Aluno", "Motorista" ou "Ônibus".	5	O usuário escolhe a opção "Aluno".
ITEM	DETALHAMENTO																
1	O sistema apresenta a tela " <u>Realizar Login</u> ".																
2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.																
3	O sistema apresenta a tela inicial.																
4	O usuário clica na opção "Cadastrar"																
4.1	O sistema apresenta a tela de cadastro com as opções "Aluno", "Motorista" ou "Ônibus".																
5	O usuário escolhe a opção "Aluno".																

6	O usuário então preenche todo o formulário corretamente e clica em “Salvar”
----------	---

Fluxo Alternativo

Não se aplica

Fluxo de Exceção

FLUXO DE EXCEÇÃO	
ITEM	DETALHES
1	Caso os dados informados sejam inválidos, apresentar as mensagens abaixo do campo que esteja inválido.
1.1	Caso os campos obrigatórios estejam em branco, deve-se apresentar a mensagem “O <nome do campo> é obrigatório”.
1.2	Caso o CPF/RG seja inválido, apresentar a mensagem “Informe um <documento X> válido”.
1.3	Caso o ano da data de nascimento seja inferior à 1900 ou superior ao <u>ano atual</u> , apresentar a mensagem “Informe uma data válida”.
1.4	Caso o e-mail seja inválido, apresentar a mensagem " Informe um e-mail válido".
1.5	Caso o e-mail já exista na base de dados, apresentar a mensagem " E-mail já cadastrado no banco de dados".
1.6	Caso a “Senha” possua a quantidade inferior a 8 caracteres ou superior a 16 caracteres, deve-se apresentar a mensagem: “A senha deve conter de 8 a 16 caracteres”.
1.7	Caso as senhas não coincidam, apresentar a mensagem “As senhas não coincidem”.

Links e botões

NOME	TIPO	FUNÇÃO	REGRA
Salvar	Botão	Verificar a informação digitada no formulário e prosseguir para a próxima página.	Habilitar o botão apenas quando todos os campos obrigatórios estiverem corretamente inseridos.
Limpar	Botão	Limpa toda a informação digitada no formulário	Habilitar o botão apenas quando houver alguma informação inserida no formulário.

Campos

NOME	TAMANHO	TIPO	DESCRIÇÃO
Nome*	50	Texto	Não se aplica
CPF*	11	Numérico	Máscara: 999.999.999-99

			Colocar uma checkbox <input type="checkbox"/> Não possui CPF Caso marcado habilitar o campo RG
RG	10	Numérico	Não se aplica
E-mail	50	E-mail	Só deve ser aceito e-mails válidos.
Data de Nascimento*	-	Date	Máscara: 99/99/9999
Nome do Responsável*	50	Texto	Não se aplica
Telefone do Responsável*	11	Numérico	Máscara: (99) 99999-9999
Escola*	50	Dropdown - Texto	Lista com todas as escolas atendidas pelo transporte
Endereço/Região*	50	Texto	Não se aplica
Observações	100	Texto	Não se aplica

*Campos obrigatórios

APÊNDICE E – UH003 LISTAR ALUNO

Visão geral do Processo

Por meio deste requisito é possível listar os alunos cadastrados no sistema.

Atores/ Perfis do Processo

NOME DO PERFIL	TIPO DE PERFIL	DESCRIÇÃO DO PERFIL
Gestor	Interno	O usuário deve estar autenticado para ter acesso à funcionalidade.

Como demonstrar

Acesso a tela de Login -> Fazer login no sistema -> Clicar em aluno.

História de Usuário

HISTÓRIA DE USUÁRIO	DESCRIÇÃO	STATUS	PRIORIDADE
UH. 003	Listar Alunos	A Desenvolver	Essencial
Eu como: Usuário já cadastrado no sistema			
Quero: Realizar a listagem de alunos cadastrados			
Para: Visualizar a relação de alunos presentes na base de dados do sistema.			
CRITÉRIOS DE ACEITAÇÃO			
ITEM	DETALHAMENTO		
1	O sistema apresenta a tela “ Realizar Login ”.		
2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.		
3	O sistema apresenta a tela inicial.		
4	O usuário clica na opção “Aluno”		
5	O sistema apresenta a tela de listagem com os alunos cadastrados		

Fluxo Alternativo

FLUXO ALTERNATIVO	
ITEM	DETALHES

1	Ao finalizar o cadastro de um aluno, deverá abrir a tela para visualização da lista de alunos.
---	--

Fluxo de Exceção

FLUXO DE EXCEÇÃO	
ITEM	DETALHES
1	Caso não haja alunos cadastrados no sistema, deve-se exibir o alerta de “Não existem alunos cadastrados”
1.1	Logo após, destacar a opção de cadastrar um aluno, caso seja exibida a mensagem que não há alunos cadastrados
2	Se houver falha de comunicação com o banco de dados, exibe-se o alerta “Falha na listagem de alunos. Erro ao comunicar com o banco de dados. <Código do erro>”.

Links e botões

NOME	TIPO	FUNÇÃO	REGRA
Cadastrar	Botão	Cadastrar novo aluno	
Organizar	Botão Drop-down	Permite organizar a listagem por ordem alfabética, por escola, por região, por idade...	Deve haver no banco de dados ao menos 2 cadastros gravados.

APÊNDICE F – UH004 ALTERAR ALUNO

Visão Geral do Processo

Por meio deste requisito é possível listar os alunos cadastrados no sistema e alterar seu cadastro caso necessário.

Atores/ Perfis do Processo

NOME DO PERFIL	TIPO DE PERFIL	DESCRIÇÃO DO PERFIL
Gestor	Interno	O usuário deve estar autenticado para ter acesso à funcionalidade.

Como demonstrar

Acesso a tela de Login -> Fazer login no sistema -> Clicar em Aluno -> Procurar o aluno desejado -> Clicar no ícone de alteração de cadastro.

História de Usuário

HISTÓRIA DE USUÁRIO	DESCRIÇÃO	STATUS	PRIORIDADE										
UH. 004	Alterar Aluno	A Desenvolver	Essencial										
<p>Eu como: Usuário já cadastrado no sistema</p> <p>Quero: Realizar a alteração dos dados de um aluno já cadastrado no sistema</p> <p>Para: Manter atualizada a base de dados do sistema</p> <p style="text-align: center;">CRITÉRIOS DE ACEITAÇÃO</p> <table border="1"> <thead> <tr> <th>ITEM</th> <th>DETALHAMENTO</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>O sistema apresenta a tela “Realizar Login”.</td> </tr> <tr> <td>2</td> <td>Usuário informa corretamente as credenciais do usuário já previamente cadastrado.</td> </tr> <tr> <td>3</td> <td>O sistema apresenta a tela inicial.</td> </tr> <tr> <td>4</td> <td>O usuário clica na opção “Aluno”</td> </tr> </tbody> </table>				ITEM	DETALHAMENTO	1	O sistema apresenta a tela “ Realizar Login ”.	2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.	3	O sistema apresenta a tela inicial.	4	O usuário clica na opção “Aluno”
ITEM	DETALHAMENTO												
1	O sistema apresenta a tela “ Realizar Login ”.												
2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.												
3	O sistema apresenta a tela inicial.												
4	O usuário clica na opção “Aluno”												

5	O sistema então busca no banco de dados todos os alunos cadastrados e os apresenta na tela.
6	O usuário procura o aluno que deseja alterar
7	O usuário então clica no ícone que indica a funcionalidade de alterar os dados.
8	O sistema carrega a tela de cadastro com os dados atuais do aluno, para que possa alterar o que for necessário e salvar novamente.

Fluxo Alternativo

Não se aplica

Fluxo de Exceção

Não se aplica

Links e botões

NOME	TIPO	FUNÇÃO	REGRA
Alterar	Botão	Possibilita abrir a tela de cadastro para alterar os dados do aluno selecionado	Selecionar um aluno na listagem

APÊNDICE G – UH005 EXCLUIR ALUNO

Visão Geral do Processo

Por meio deste requisito é possível listar os alunos cadastrados no sistema e excluir seu cadastro caso necessário.

Atores/ Perfis do Processo

NOME DO PERFIL	TIPO DE PERFIL	DESCRIÇÃO DO PERFIL
Gestor	Interno	O usuário deve estar autenticado para ter acesso à funcionalidade.

Como demonstrar

Acesso a tela de Login -> Fazer login no sistema -> Clicar em Aluno -> Procurar o aluno desejado -> Clicar no ícone de exclusão de cadastro.

História de Usuário

HISTÓRIA DE USUÁRIO	DESCRIÇÃO	STATUS	PRIORIDADE
UH. 005	Excluir Aluno	A Desenvolver	Essencial
<p>Eu como: Usuário já cadastrado no sistema</p> <p>Quero: Realizar a exclusão de um aluno no sistema</p> <p>Para: Manter atualizada a base de dados do sistema</p>			
CRITÉRIOS DE ACEITAÇÃO			
ITEM	DETALHAMENTO		
1	O sistema apresenta a tela “ Realizar Login ”.		
2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.		
3	O sistema apresenta a tela inicial.		
4	O usuário clica na opção “Aluno”		
5	O sistema então busca no banco de dados todos os alunos cadastrados e os apresenta na tela.		
6	O usuário procura o aluno que deseja alterar		
7	O usuário então clica no ícone que indica a funcionalidade de excluir.		

Fluxo Alternativo

Não se aplica

Fluxo de Exceção

Não se aplica

Links e botões

NOME	TIPO	FUNÇÃO	REGRA
Excluir	Botão	Apagar os dados do aluno selecionado	Exibir uma mensagem de confirmação antes de executar a função.

APÊNDICE H – UH006 MATER ROTAS

Visão Geral do Processo

Por meio deste requisito é possível adicionar, editar ou remover uma rota do sistema.

Atores/ Perfis do Processo

NOME DO PERFIL	TIPO DE PERFIL	DESCRIÇÃO DO PERFIL
Gestor	Interno	O usuário deve estar autenticado para ter acesso à funcionalidade.

Como demonstrar

Acesso a tela de Login -> Fazer login no sistema -> Clicar em Rotas -> Clicar no ícone de adicionar nova rota.

História de Usuário

HISTÓRIA DE USUÁRIO	DESCRIÇÃO	STATUS	PRIORIDADE																
UH. 006	Manter Rotas	A Desenvolver	Essencial																
<p>Eu como: Usuário já cadastrado no sistema</p> <p>Quero: Realizar a inclusão de uma nova rota no sistema</p> <p>Para: Manter atualizada a base de dados do sistema</p> <p style="text-align: center;">CRITÉRIOS DE ACEITAÇÃO</p> <table border="1"> <thead> <tr> <th>ITEM</th> <th>DETALHAMENTO</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>O sistema apresenta a tela “Realizar Login”.</td> </tr> <tr> <td>2</td> <td>Usuário informa corretamente as credenciais do usuário já previamente cadastrado.</td> </tr> <tr> <td>3</td> <td>O sistema apresenta a tela inicial.</td> </tr> <tr> <td>4</td> <td>O usuário clica na opção “Rotas”</td> </tr> <tr> <td>5</td> <td>O sistema então busca no banco de dados todas as rotas cadastradas e os apresenta na tela.</td> </tr> <tr> <td>6</td> <td>O usuário deve clicar no ícone de inserir nova rota ou editar/excluir uma existente</td> </tr> <tr> <td>7</td> <td>O sistema então abre a API de geolocalização para definir o ponto de</td> </tr> </tbody> </table>				ITEM	DETALHAMENTO	1	O sistema apresenta a tela “ Realizar Login ”.	2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.	3	O sistema apresenta a tela inicial.	4	O usuário clica na opção “Rotas”	5	O sistema então busca no banco de dados todas as rotas cadastradas e os apresenta na tela.	6	O usuário deve clicar no ícone de inserir nova rota ou editar/excluir uma existente	7	O sistema então abre a API de geolocalização para definir o ponto de
ITEM	DETALHAMENTO																		
1	O sistema apresenta a tela “ Realizar Login ”.																		
2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.																		
3	O sistema apresenta a tela inicial.																		
4	O usuário clica na opção “Rotas”																		
5	O sistema então busca no banco de dados todas as rotas cadastradas e os apresenta na tela.																		
6	O usuário deve clicar no ícone de inserir nova rota ou editar/excluir uma existente																		
7	O sistema então abre a API de geolocalização para definir o ponto de																		

partida e chegada da rota.

Fluxo Alternativo

Não se aplica

Fluxo de Exceção

Não se aplica

Links e botões

NOME	TIPO	FUNÇÃO	REGRA
Adicionar	Botão	Criar uma nova rota	-
Editar	Botão	Editar uma rota existente	Só deve estar disponível quando houver rotas já cadastradas
Excluir	Botão	Excluir uma rota existente.	Só deve estar disponível quando houver rotas já cadastradas

APÊNDICE I – UH007 REGISTRAR VIAGEM

Visão Geral do Processo

Por meio deste requisito é possível registrar dados sobre a viagem de determinada rota

Atores/ Perfis do Processo

NOME DO PERFIL	TIPO DE PERFIL	DESCRIÇÃO DO PERFIL
Motorista	Interno	O usuário deve estar autenticado para ter acesso à funcionalidade.

Como demonstrar

Acesso a tela de Login -> Fazer login no sistema -> Clicar em Rotas> O sistema exibe as rotas vinculadas àquele motorista

História de Usuário

HISTÓRIA DE USUÁRIO	DESCRIÇÃO	STATUS	PRIORIDADE																				
UH. 007	Registrar viagem	A Desenvolver	Essencial																				
<p>Eu como: Usuário já cadastrado no sistema</p> <p>Quero: Realizar o registro dos dados sobre a viagem</p> <p>Para: Manter atualizada a base de dados do sistema</p> <p style="text-align: center;">CRITÉRIOS DE ACEITAÇÃO</p> <table border="1"> <thead> <tr> <th>ITEM</th> <th>DETALHAMENTO</th> </tr> </thead> <tbody> <tr> <td>1</td> <td>O sistema apresenta a tela “Realizar Login”.</td> </tr> <tr> <td>2</td> <td>Usuário informa corretamente as credenciais do usuário já previamente cadastrado.</td> </tr> <tr> <td>3</td> <td>O sistema apresenta a tela inicial.</td> </tr> <tr> <td>4</td> <td>O usuário clica na opção “Rotas”</td> </tr> <tr> <td>5</td> <td>O sistema então exibe as rotas vinculadas para aquele motorista</td> </tr> <tr> <td>6</td> <td>O usuário deve clicar na rota que irá trabalhar</td> </tr> <tr> <td>7</td> <td>O usuário deve clicar em ‘iniciar viagem’ assim que for iniciar a rota</td> </tr> <tr> <td>8</td> <td>Deve informar em seguida a quilometragem inicial do veículo</td> </tr> <tr> <td>9</td> <td>O sistema registra a data e hora de saída, além da quilometragem inicial</td> </tr> </tbody> </table>				ITEM	DETALHAMENTO	1	O sistema apresenta a tela “ Realizar Login ”.	2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.	3	O sistema apresenta a tela inicial.	4	O usuário clica na opção “Rotas”	5	O sistema então exibe as rotas vinculadas para aquele motorista	6	O usuário deve clicar na rota que irá trabalhar	7	O usuário deve clicar em ‘iniciar viagem’ assim que for iniciar a rota	8	Deve informar em seguida a quilometragem inicial do veículo	9	O sistema registra a data e hora de saída, além da quilometragem inicial
ITEM	DETALHAMENTO																						
1	O sistema apresenta a tela “ Realizar Login ”.																						
2	Usuário informa corretamente as credenciais do usuário já previamente cadastrado.																						
3	O sistema apresenta a tela inicial.																						
4	O usuário clica na opção “Rotas”																						
5	O sistema então exibe as rotas vinculadas para aquele motorista																						
6	O usuário deve clicar na rota que irá trabalhar																						
7	O usuário deve clicar em ‘iniciar viagem’ assim que for iniciar a rota																						
8	Deve informar em seguida a quilometragem inicial do veículo																						
9	O sistema registra a data e hora de saída, além da quilometragem inicial																						

10	Assim que terminar a viagem, o sistema apresenta a opção encerrar rota
11	Deve informar em seguida a quilometragem no momento
12	O sistema registra a data, hora de chegada e quilometragem final.

Fluxo Alternativo

Caso haja qualquer tipo de ocorrência durante a viagem, como por exemplo; Estrada interditada, impossibilidade de transitar, problemas no veículo, algum problema com algum aluno; o sistema exibe a opção ‘registrar ocorrência’, podendo detalhar o empecilho encontrado para o sistema registrar.

Fluxo de Exceção

Não se aplica

Links e botões

NOME	TIPO	FUNÇÃO	REGRA
Iniciar viagem	Botão	Inicia a rota desejada	-
Encerrar viagem	Botão	Encerra a rota que estava em curso	Só deve estar disponível quando uma rota estiver em andamento
Registrar Ocorrencia	Botão	Registra ocorrências sobre a viagem	Só deve estar disponível quando uma rota estiver em andamento

APÊNDICE J – TELAS DO SISTEMA

