UNIVERSIDADE REGIONAL DE BLUMENAU

CENTRO DE CIÊNCIAS EXATAS E NATURAIS

CURSO DE SISTEMAS DE INFORMAÇÃO – BACHARELADO

GESTÃO DE PRODUTIVIDADE EM EMPRESAS DE TRANSPORTE DE PASSAGEIROS UTILIZANDO SISTEMA WEB

STEFANIE VOSS

BLUMENAU
2016

STEFANIE VOSS

GESTÃO DE PRODUTIVIDADE EM EMPRESAS DE TRANSPORTE DE PASSAGEIROS UTILIZANDO SISTEMA WEB

Trabalho de Conclusão de Curso apresentado ao curso de graduação em Sistemas de Informação do Centro de Ciências Exatas e Naturais da Universidade Regional de Blumenau como requisito parcial para a obtenção do grau de Bacharel em Sistemas de Informação.

Prof. Adriano Gonçalves Polidoro, MBA - Orientador

BLUMENAU
2016

GESTÃO DE PRODUTIVIDADE EM EMPRESAS DE TRANSPORTE DE PASSAGEIROS UTILIZANDO SISTEMA WEB

Por

STEFANIE VOSS

Trabalho de Conclusão de Curso aprovado para obtenção dos créditos na disciplina de Trabalho de Conclusão de Curso II pela banca examinadora formada por:

Presidente:	Prof. Adriano Gonçalves Polidoro, MBA – Orientador, FURB
Membro:	Prof. Francisco Adell Péricas, Titulação – FURB
Membro:	Prof. Simone Erbs da Costa, MBA – FURB

Blumenau, 06 de dezembro de 2016

Dedico este trabalho a todos os empresários que merecem mais agilidade e produtividade na gestão e administração dos processos de suas empresas de transporte e turismo de passageiros. Dedico em especial ao meu pai, Alidor Voss, que foi e será o meu principal ator, inspirador e divulgador da ideia deste trabalho.

AGRADECIMENTOS

Gostaria de agradecer primeiramente a Deus, no qual me deu forças, fé e me dispôs de saúde para conseguir persistir e acreditar neste trabalho até o final, mesmo com tantos empecilhos e dificuldades.

À minha família, que tanto me incentivou e me fez manter a calma do início até o final do trabalho. Gostaria também de lembrar aqui o quanto sou grata por todo o estudo e a base familiar que meus pais me dispuseram para conseguir chegar aonde estou hoje. Cada momento e ensinamento que meus familiares repassaram, me fez ser quem sou hoje. Obrigada.

Aos meus amigos, em especial ao Lucas Amaral e ao Guilherme Dalmarco, que sempre se dispuseram e me auxiliaram em cada dúvida ou dificuldade técnica no decorrer do desenvolvimento do trabalho em questão.

Ao meu orientador, Adriano Gonçalves Polidoro, que se dispôs no suporte e auxílio em todas as etapas, desde a análise inicial até os pontos mais cruciais e finais deste trabalho. O mesmo sempre se esforçou ao máximo para me manter focada, organizada e planejada em cada etapa e versão a ser entregue.

Ao Deter-SC, em especial a Sra. Silmara D. Ribeiro e o Sr. Gustavo F. Santos, no qual concordaram em abrir um espaço para a minha ideia e com isso se dispuseram e auxiliaram no suporte e na criação da futura API de integração no decorrer do trabalho.

Muito obrigada a todos os envolvidos, vocês foram de suma importância em mais essa etapa da minha vida.

“A tarefa não é tanto ver aquilo que ninguém viu, mas pensar o que ninguém ainda pensou sobre aquilo que todo mundo vê”

Arthur Schopenhauer

RESUMO

Este trabalho apresenta a análise e a descrição do desenvolvimento de um sistema web para empresas do ramo do transporte turístico de passageiros, o qual controla e administra viagens, veículos, motoristas, contratantes, passageiros e dados das notas fiscais. Para empresas do estado de Santa Catarina, o sistema possuirá futuramente a opção de gerar a licença de viagem especial por meio da integração com o sistema do Deter-SC. O mesmo dispõe de uma interface moderna e de fácil usabilidade. Estas características foram desenvolvidas através dos *frameworks* Play e Spring integrados à linguagem de programação Java. Para o desenvolvimento do *front-end*, foram utilizados os *frameworks* Bootstrap e AngularJS integrados às linguagens de programação HTML5, CSS3 e Java-Script. O banco de dados PostgreSQL foi utilizado para a persistência dos dados.

Palavras-chave: Transporte de passageiros. Turismo. Licença de viagem especial. Deter-SC. Produtividade. Gestão. Integração de sistemas. AngularJS. Play framework. Spring. PostgreSQL. Sistema web.

ABSTRACT

This paper presents the analysis and description of the development of a web system which controls and manages trips, cars, drivers, contractors, passengers and data from invoices. For companies in the state of Santa Catarina - Brazil, the system also has the future option to generate a special trip through integration with Deter-SC's system. The latter has a modern interface and easy communication with the user. These characteristics were available with frameworks Play and Spring integrated with the Java programming language. Furthermore, for the development of the front-end it was used the frameworks Bootstrap and AngularJS integrated with HTML5, CSS3 and Java-Script programming languages. The PostgreSQL database was used for the storage of data.

Key-words: Passenger transport. Tourism. Special travel license. Deter-SC. Productivity. Management. Systems integration. AngularJS. Play framework. Spring. PostgreSQL. Web system.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLAS

API – Application Programming Interface

CSS - Cascade Style Sheets

DER - Diagrama Entidade Relacionamento

DETER – Departamento de Transportes e Terminais

HTML - HyperText Markup Language

HTTP - Hypertext Transfer Protocol

JavaEE - Java Enterprise Edition

MER - Modelo Entidade Relacionamento

MV - Model View

MVC – Model View Controller

REST - Representational State Transfer

RF – Requisito Funcional

RN – Regras de Negócio

RNF – Requisito Não Funcional

SC – Santa Catarina

TI – Tecnologia da Informação

TIC – Tecnologia de Informação e Comunicação

UC – Caso de Uso

UML - Unified Modeling Language

URI - Uniform Resource Identifier

URL - Uniform Resource Locator

SUMÁRIO

1 INTRODUÇÃO

Atualmente o transporte e o turismo caminham em conjunto, designando novos fatores e abordagens para a sociedade e o mundo em que se vive. Desta forma, temas como crescimento de companhias de transporte e agências de pequeno porte, impactos ambientais, turismo privado versus público, segurança e principalmente a Tecnologia da Informação (TI) e a oferta no transporte turístico, tendem a ser cada dia mais abordados e analisados.

Segundo Page (2008, p. 29), "o transporte é considerado um dos fatores que mais contribuíram com o desenvolvimento internacional do turismo". A importante relação que existe entre o transporte e o turismo é demonstrada por Lamb e Davidson (1996, p. 264), desta forma:

> O transporte é um dos três componentes fundamentais do turismo. Os outros dois são o produto turístico (ou a oferta) e o mercado turístico (ou a demanda). Sem transporte, a maioria das formas de turismo não existiria. Em alguns casos a experiência com o transporte é a experiência turística (p.ex., cruzeiros, viagens de trem em cenários pitorescos e históricos e roteiros de motorhome, automóvel e bicicleta). (LAMB; DAVIDSON, 1996, p. 264).

Como é observado, o deslocamento e o transporte são citados como facilitadores da expansão do turismo, uma vez que o desenvolvimento de novas tecnologias tem contribuído para o desenvolvimento do turismo como um produto de consumo em massa. Estas novas tecnologias vão desde novos tipos de máquinas, como motores a jato, tipos de produtos, marketing, e as principais tecnologias da informação e logísticas como softwares ágeis.

A "era da informação" está presente na sociedade atual, isto trouxe vários fatores que implicam na prestação dos serviços do transporte, entre eles o transporte turístico. Um dos impactos imediatos para os fornecedores de transporte turístico, segundo Page (2008, p. 251), "é que os fluxos de informações atualizadas são vitais quando existe a cadeia da oferta e que o fornecedor do transporte é apenas um dos componentes do produto turístico como um todo". Desta forma, várias empresas do ramo têm buscado se adequar às novas soluções logísticas por meio da TI, que auxiliam na praticidade e agilidade dos serviços, como também trazem mais rentabilidade.

Diante deste cenário, para acompanhar o mercado de logística e de TI, o presente trabalho propõe a ideia de auxiliar, automatizar, integrar e agilizar a produtividade e o funcionamento interno de viagens das empresas e agências privadas de transporte e turismo rodoviário de passageiros.

1.1 PROBLEMA

O cenário atual dos transportes e turismo necessita cada vez mais de Tecnologia da Informação e Comunicação (TIC) presentes no seu dia a dia. Esta tecnologia da informação, juntamente com a logística, pode ser um dos principais fatores de problemas e implicações dentro das empresas de transporte turístico.

Como Christopher (1994, p. 12) observa, a explosão do serviço ao consumidor mostra que existe a necessidade de provisão consistente do tempo e do espaço, em outras palavras, os produtos não têm valor até que estejam nas mãos dos consumidores no momento e no local correto. Complementando esta ideia, Page (2008, p. 251) ressalta que "a logística da prestação de serviços é um conceito vital a ser reconhecido, particularmente quando a TI é também introduzida, uma vez que a TI e a logística permitem que os fornecedores de transporte alcancem seus objetivos em um ambiente competitivo."

Neste contexto, pode-se lembrar a importância da TI no âmbito do turismo. Normalmente empresas nesta área possuem certa dificuldade para tornar as tarefas rotineiras mais automatizadas, ágeis e produtivas. Uma empresa de transporte de passageiros leva um longo tempo para obter as devidas informações de viagens, veículos, motoristas, contratantes e listas de passageiros para a realização de uma viagem. Além disso, no caso de empresas do estado de Santa Catarina (SC), pode-se citar a dificuldade para emitir uma autorização de viagem especial por meio do sistema do Departamento de Transportes e Terminais do Estado de Santa Catarina (DETER-SC). O retrabalho e o esquecimento de certas informações poderiam ser evitados caso houvesse um sistema de automatização das tarefas das empresas e organização dos históricos de dados.

1.2 JUSTIFICATIVA

No âmbito do transporte rodoviário turístico de passageiros há dificuldades em encontrar recursos disponíveis para auxiliar na organização, agilidade e facilidade dos serviços realizados por empresas do ramo. Embora exista no mercado algumas opções de sistemas que facilitem cadastros e históricos de transportadoras, nas pesquisas realizadas nos trabalhos correlatos não foram encontrados recursos que sejam focados em transportadoras do ramo turístico ou em transporte de passageiros, a maioria destes são focados em transportes rodoviários públicos ou em transportes ferroviários de cargas. Além disso, várias destas empresas de turismo possuem no seu dia a dia retrabalhos, perdas de informações, falta de

agilidade e desorganização, pontos negativos que poderiam ser evitados com um sistema que os abrangessem.

Desta forma, pode-se citar também o retrabalho e a demora em que uma empresa de transporte turístico de passageiros de Santa Catarina leva para consolidar novamente todos os dados de uma viagem e emitir a autorização de viagem especial por meio do site do Deter-SC. Segundo Voss (2016), dono e motorista da empresa Continental Turismo de Blumenau-SC, o seu trabalho facilitaria se estas informações já estivessem cadastradas previamente de forma útil e assim integradas diretamente com o sistema do Deter-SC, sem a necessidade de redigitar todos os dados das viagens, como: motoristas, veículos, contratantes, notas fiscais e listas de passageiros.

Baseado nessas informações verificou-se dificuldades e falta de recursos tecnológicos para essa área. Sendo assim, a proposta é organizar e resolver estes problemas de forma ágil, aumentando consequentemente a produtividade das empresas envolvidas como é clarificado abaixo nos objetivos gerais e específicos do trabalho.

1.3 OBJETIVOS

O objetivo geral do trabalho proposto é disponibilizar um sistema para donos de empresas de transporte de passageiros que mantenha, organize, auxilie e agilize as tarefas rotineiras da organização.

Os objetivos específicos do trabalho proposto são:

a) permitir que donos e/ou funcionários de empresas de transporte de passageiros, mantenham e organizem viagens, listas de passageiros, motoristas, veículos, contratantes e dados das notas fiscais;
b) disponibilizar uma base completa para realizar a integração com a futura API do sistema Deter-SC, com o intuito de gerar a autorização de viagem especial para cada viagem confirmada;
c) gerar lembretes e notificações via e-mail de tarefas pendentes, como viagens, confirmações, faturamento, entre outros;
d) disponibilizar gráficos com informações de viagens e faturamentos úteis para os donos e/ou funcionários das empresas.

2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo aborda assuntos a serem apresentados nas seções a seguir, tais como os transportes, planejamento dos transportes, transporte e turismo, logística e administração do turismo na TI, além dos trabalhos correlatos.

2.1 TRANSPORTES

Segundo Pena (2010, p. 1), “os transportes correspondem ao conjunto de materiais e instrumentos técnicos utilizados no deslocamento de pessoas e cargas de um lugar para o outro”. Sendo assim, pode-se compreender que os transportes movem o mundo e quanto maior for a gama por transportes de qualidade, maior será o desenvolvimento dos países e sociedades neles inseridos. Os meios de transportes são um dos principais elementos para garantir a infraestrutura e o crescimento das cidades, estados e países. Andrade (1994, p. I-2) explica que:

> A infraestrutura de transportes é um pré-requisito básico do desenvolvimento econômico, embora não se constitua em sua garantia. (...) O setor de transportes desempenha papel fundamental prestando serviços absorvidos, praticamente, por todas as unidades produtivas discriminadas no espaço econômico. (ANDRADE, 1994, p. I-2).

Percebe-se que as atividades de transporte exercem um papel muito importante no funcionamento de um sistema econômico. Pode-se ressaltar que os transportes de cargas, sejam eles nas áreas agrícolas, tecnológicas ou industriais, como também os transportes de pessoas para fins comerciais, trabalhistas ou turísticos, são de suma importância para qualquer cidade, estado e país que queira crescer financeiramente e se desenvolver de forma ágil e organizada. É possível entender que, quanto maior o crescimento econômico de uma região, maior será a demanda e a pressão sobre os meios de transportes. Andrade (1994, p. I-3) exemplifica ainda que:

> Os intensos esforços despendidos pelos países em desenvolvimento para aumentar sua produção agrícola requerem a disponibilidade oportuna de sementes, fertilizantes e outros insumos. Os produtores precisam de acessos razoáveis aos seus mercados. A expansão da indústria exige o transporte eficiente das matérias primas, bem como para a distribuição dos produtos acabados. A mobilização das populações nos grandes centros urbanos alcança volumes cada vez mais significativos e dela depende muito a performance das cidades. As exportações necessitam de instalações portuárias adequadas e de transporte terrestre para acesso aos portos. Sobem os custos dos produtos importados quando a demora dos navios nos portos se torna excessiva. (ANDRADE, 1994, p. I-3).

Estes exemplos mostram o quanto os transportes e as suas atividades movem o funcionamento de um sistema econômico dentro do desempenho da sociedade humana. O

produto do transporte depende também do desenvolvimento das demais áreas e setores que, por sua vez, se fortalecem e se alimentam da função transporte. Por isso, é possível dizer que os transportes, ao lado da comunicação e do planejamento são um elemento estratégico para qualquer país ou governo.

Ao longo da história, o próprio sistema capitalista necessitou da evolução dos meios de transporte para se desenvolver. Nos séculos XV e XVI, foram o desenvolvimento das técnicas de navegação e a produção das grandes caravelas que permitiram a expansão colonial europeia e a reprodução de seu sistema econômico para outras sociedades, dando início ao processo de mundialização do capitalismo, atualmente é chamado de globalização (PENA, 2010).

A globalização necessitou de toda a evolução do transporte para crescer e se difundir pelo mundo. Eventos históricos como a Revolução Industrial, a Revolução Técnico-Científica, a implementação do trem a vapor e dos grandes navios, a invenção dos automóveis, e mais tarde, o uso contínuo do transporte aéreo, trouxe um mundo em que pessoas e cargas em geral deslocam-se de uma maneira ágil e perspicaz. Desta forma, pode-se entender o quanto o transporte está inserido em nossas vidas e que todas essas mudanças repentinas fazem o mundo entrar em um contexto em que o planejamento, o desenvolvimento, a evolução e o estudo dos transportes devem estar em primeiro lugar.

2.2 PLANEJAMENTO DOS TRANSPORTES

Segundo Acerenza (2003, p. 23), planejamento “é a seleção consciente de determinada linha de ação, que se diferencia das ações adotadas por costume, impulsos irracionais e inclusive por ignorância”. Deste modo, pode-se observar que o planejamento pode ser entendido sobre como o ser humano costuma pensar logicamente e organizar-se antes de agir ou tomar qualquer ação seguinte. O planejamento consiste precisamente em determinar os objetivos do trabalho, ordenar os recursos materiais e humanos disponíveis, estabelecer os métodos e técnicas que serão usados e precisar a forma de organização exigida, bem como todas as especificações necessárias para que a conduta da pessoa ou grupo de pessoas se oriente de maneira racional para os resultados que devem ser alcançados (ACERENZA, 2003).

Com isso, unir os conceitos acima descritos de planejamento e de transportes se constitui, na realidade, em fundir as características pensantes do planejamento às funções que os transportes exercem no desenvolvimento das atividades de um sistema econômico

(ANDRADE, 1994). Por conta deste conceito deve-se atentar que os meios de transportes podem ser classificados em modos que costumam ser conhecidos como terrestre, aéreo e hidroviário juntamente com as suas respectivas ramificações, como mostra a Figura 1.

Figura 1 - Exemplificação dos modos de transportes

Modos de Transportes		
Terrestre	Rodoviário	Ônibus
		Caminhão
		Automóvel
		Motocicleta
		Bicicleta
		Carroça
	Ferroviário	Trem
		Metrô
		Bonde
Aéreo	Avião	
	Helicóptero	
Hidroviário	Marítimo	Navio
		Barco
	Fluvial	Navio
		Barco
Dutoriário	Líquido	
	Gás	
	Sólido	

Fonte: Adaptado de Transporte e Turismo (2008).

O avanço tecnológico e as demandas específicas têm conduzido a descoberta e o estudo de novos modos de transporte, desta forma é recomendável atentar-se antes de iniciar qualquer tipo de planejamento na área. Hutchinson (1979) adverte que o planejamento de um sistema de transportes consiste em um conjunto de definições operacionais que identificam as necessidades econômicas e sociais de uma coletividade que o sistema procura satisfazer. Contudo pode-se salientar que o sistema de transportes deverá desempenhar sua função de produtor de serviços intermediários de forma mais adequada à sua operação econômica. Andrade (1994, p. I-8), afirma que os objetivos do planejamento em transportes são:

a) minimizar os custos operacionais de prestação de serviço;
b) minimizar os custos de capital;
c) ampliar a segurança do transporte.

Andrade (1994, p. I-9) ainda complementa que:

> Fica claro, por conseguinte, que os objetivos gerais da economia estabelecem, muitas vezes, limitações às soluções ótimas de operação e investimento do sistema de transportes. Isso significa que o sistema de transportes estará sujeito a uma série de limitações de ordem física, social, técnica, legal, etc. Um plano de transportes

> deverá ressaltar, todavia, seus objetivos internos de tal forma que a função desempenhada pelo sistema se sobreponha a decisões de caráter arbitrário, que levam em pequena conta os aspectos econômicos da operação e dos investimentos. (ANDRADE, 1994, p. I-9).

Sintetizando o que já foi exposto, para se obter uma certa harmonia entre o plano de transportes e a estratégia de desenvolvimento, é preciso compreender que o desenvolvimento econômico exige, além do desempenho das funções, uma expansão acelerada e condicionada pelas próprias características de desenvolvimento. Desta forma, os critérios de decisão da política de transportes podem apresentar ênfase variável nos objetivos internos do seu planejamento, de acordo com os estágios de desenvolvimento do país ou região (ANDRADE, 1994). É necessário ter em mente que, dentro da área do transporte, para colocar qualquer ideia ou projeto de planejamento em ação, seja este para auxiliar, melhorar ou simplesmente para inserir algo novo, deve-se preocupar-se com vários fatores já citados anteriormente, e entender que o transporte em geral traz uma variedade de informações e burocracias envolvidas e que não podem ser ignoradas ou esquecidas em momento algum do planejamento.

Abrangendo uma das áreas mais atuais e diversificadas do planejamento do transporte, pode-se lembrar também do planejamento do turismo. Este é de competência do organismo nacional de turismo, o qual, por meio da elaboração e execução de planos nacionais ou regionais de desenvolvimento do turismo, promove e orienta o incremento da atividade em prol dos objetivos nacionais e internacionais (ACERENZA, 2003). Desta forma, o próximo capítulo abordará questões específicas do transporte e turismo mundial, especificando determinadas atividades e questões já descritas.

2.3 TRANSPORTE E TURISMO

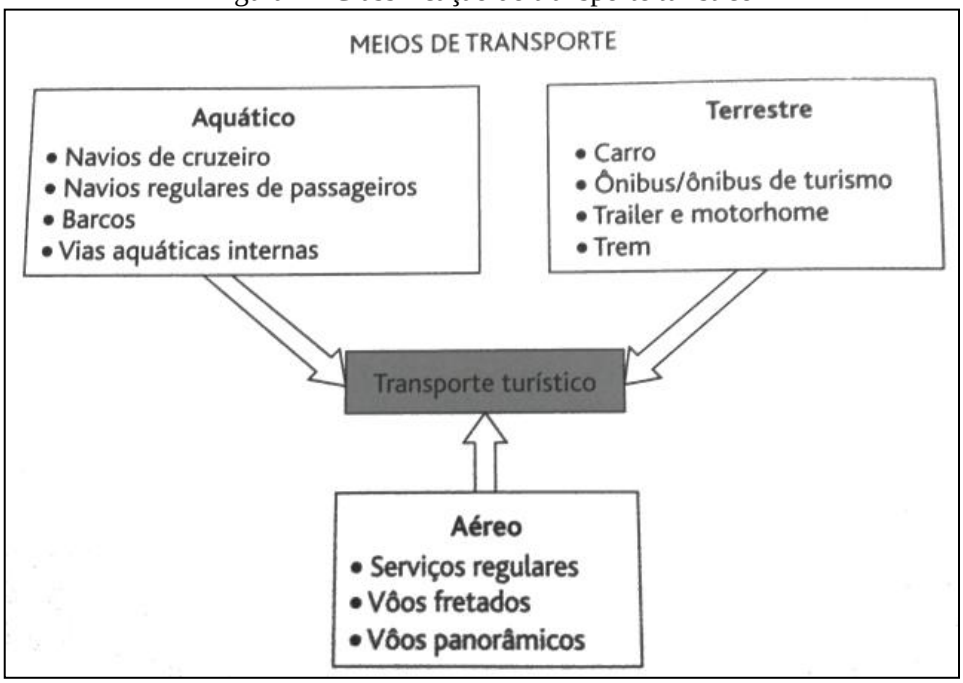
O transporte é considerado um dos fatores que mais contribuiu com o desenvolvimento internacional do turismo. Em termos globais, a expansão do turismo internacional continua a gerar uma demanda insaciável por viagens ao exterior. A Europa se mantém como a região mais visitada do mundo, com metade de todas as chegadas de turistas e quase dois terços das chegadas internacionais (PAGE, 2008). Mesmo havendo controvérsias sobre a real definição do turismo, ficando entre os conceitos de uma atividade industrial ou de serviços, de acordo com Page (2008, p. 30), está amplamente reconhecido que o turismo combina uma amplitude de atividades econômicas e serviços designados para atender as necessidades dos turistas. É evidente que o turismo é um termo amplo, que engloba muitas questões, tais como

acomodação, alimentação, serviços de apoio e por fim, mas não menos importante, o transporte. Page (2008, p. 34) reforça essa ideia ao afirmar que:

> O transporte fornece a ligação essencial entre os locais de origem e destino do turismo, facilitando o deslocamento de pessoas em férias, viajantes a negócios, pessoas visitando amigos e parentes e daqueles envolvidos em turismo por motivos de saúde ou educacionais. (...) Embora seja amplamente reconhecida a existência de meios de transporte próprios e especializados para o turismo (i. e. ônibus de turismo, voos fretados, e navios de cruzeiro), também existem outras formas de transporte que são utilizadas tanto por anfitriões, quanto por turistas em diversas dimensões. Por exemplo, os ônibus urbanos, os sistemas de metrô e os voos regulares existentes para regiões turísticas são usados por turistas e residentes locais e, em alguns casos, isto pode causar alguma concorrência. (PAGE, 2008, p. 34).

A análise de livros e textos de turismo indica que o deslocamento e o transporte são citados em relação ao seu novo papel como facilitadores da expansão do turismo, uma vez que o desenvolvimento de novas tecnologias e as novas formas de desenvolvimento de produtos e marketing têm contribuído para o desenvolvimento do turismo como um produto de consumo em massa. Collier (2004, p. 48) possui três visões sobre as necessidades do transporte turístico, sendo elas: "transportar os turistas da área de origem até o destino; transporte entre os destinos e transporte nos limites internos dos destinos". Collier (2004) também classifica o transporte turístico sob diversas bases, que pode ser verificada na Figura 2.

Figura 2 - Classificação do transporte turístico



Fonte: Collier (1994).

Para entender a complexidade e as relações que coexistem entre o turismo e o transporte, faz-se necessária a construção de uma estrutura que possa sintetizar os diferentes

fatores e processos que impactam na organização, operação e gestão de atividades associadas à viagem turística. De acordo com Lumsdon e Page (2004, p.1):

> O desenvolvimento de uma estrutura teórica de análise foca em uma abordagem sistêmica. O sistema do turismo é definido por McIntosh et al (1995, p.21) como o "conjunto de grupos inter-relacionados e coordenados para formar um todo unificado e organizado, para cumprir com uma série de metas". Mill e Morrison (1992), ao explicar o sistema turístico, destacam o processo de troca existente entre consumidores e fornecedores, através de quatro componentes integrados: o mercado, o elemento deslocamento, o destino e o mecanismo de marketing. Estes elementos estão conectados, em primeira instância, pelo fluxo de informações e, seguidos pelas viagens entre os locais de origem e de destino. Os processos, os quais permitem que isto aconteça, por exemplo, intermediários e provedores de transporte, facilitam a experiência turística. (LUMSDON; PAGE, 2004, p. 1).

Identifica-se ainda os seguintes elementos de um sistema turístico: o turista; a região geradora de turistas; regiões de destinos turísticos; rotas de trânsito para os turistas em deslocamento entre os locais de origem e o destino; e o setor de viagens e turismo, incluindo acomodações, transporte, empresas e organizações fornecedoras de serviços e produtos para turistas (LEIPER, 1990). Neste contexto, o transporte consolida-se como uma parte essencial do sistema turístico, conectando os locais de origem e destino, o que é representado em termos de volume de viagens.

A importância do transporte no sistema turístico é também aparente no modelo desenvolvido por Laws (1991), no qual uma série de subsistemas menores, também foram identificados (p.exe. o sistema de transportes), os quais podem ser analisados como uma atividade discreta na sua forma e, ao mesmo tempo, parte integral do sistema turístico mais amplo. Portanto, um sistema de transporte turístico é uma estrutura que contempla toda a experiência turística de viajar em um meio de transporte em particular (PAGE, 2008). Todos estes conceitos e análises instruiram Lumsdon e Page (2004, p. 4) a fazer a distinção entre:

> Transporte para o turismo, onde o transporte é um meio para um fim, sendo muito útil, e o nível de satisfação está relacionado ao custo e à velocidade da viagem, de forma que o meio de transporte não tenha valos direto intrínseco por si só. Portanto, como Prideaux (2000) percebeu, o deslocamento tem sido visto tradicionalmente como um custo ao invés de um benefício, quase como um custo de oportunidade. E o transporte como turismo, onde o meio de transporte é o contexto para o deslocamento, tal como um cruzeiro, e a base para a experiência turística. Aqui o princípio do custo do deslocamento, no caso do transporte para o turismo, não se aplica, pois o transporte é o maior benefício, ou pelo menos muitos dos atributos associados ao meio de transporte são benéficos. (LUMSDON; PAGE, 2004, p. 4).

Com isto, deve-se atentar na forma em que o turismo é visto no dia a dia e em como todo este processo está evoluindo rapidamente. Outro assunto muito citado dentro da área de transporte e turismo é a globalização, este é envolvido em todos os quesitos e evoluções do transporte e principalmente nas evoluções do turismo. De acordo com Lumsdon e Page (2004, p. 12), "na sua formação mais simples a globalização refere-se ao aumento da escala

geográfica das interações econômicas, sociais e políticas. Estas incluem a expansão da mobilidade de transações de capital e investimentos, o crescimento do turismo, conferencias globais e eventos esportivos". Deste modo, o transporte e o turismo conseguem auxiliar a globalização e consequentemente aumenta-se o índice de crescimento na economia e nas atividades turísticas de regiões.

Dentre todos os fatores e conceitos vistos e analisados, pode-se perceber o quanto o transporte e o turismo são abrangentes em áreas completamente diversificadas. Com isto, não se pode esquecer o quanto o marketing é importante e vem sendo destacado quando as pessoas sabem utilizá-lo dentro das diversas atividades turísticas. O marketing em geral aborda assuntos que são muito utilizados para qualquer planejamento de projeto e também para a realização do presente trabalho. O planejamento estratégico, por exemplo, é um dos assuntos abordados e discutidos em marketing e ele é muito utilizado no trabalho em questão. Em qualquer negócio ou empresa, existe a necessidade de manter um nível de organização ou estrutura para as suas atividades, para poder pensar no futuro. Isto se torna essencial quando as empresas desejam responder ao ambiente competitivo no qual elas operam. Por esta razão, um procedimento formal para planejar, conhecido como planejamento estratégico se faz necessário. Um dos pontos também abordado no turismo é a logística focada na TIC, na qual se encaixam todos os planejamentos, análises, conceitos e inclusive o marketing. Este assunto e as suas especificações será abordado na próxima sessão.

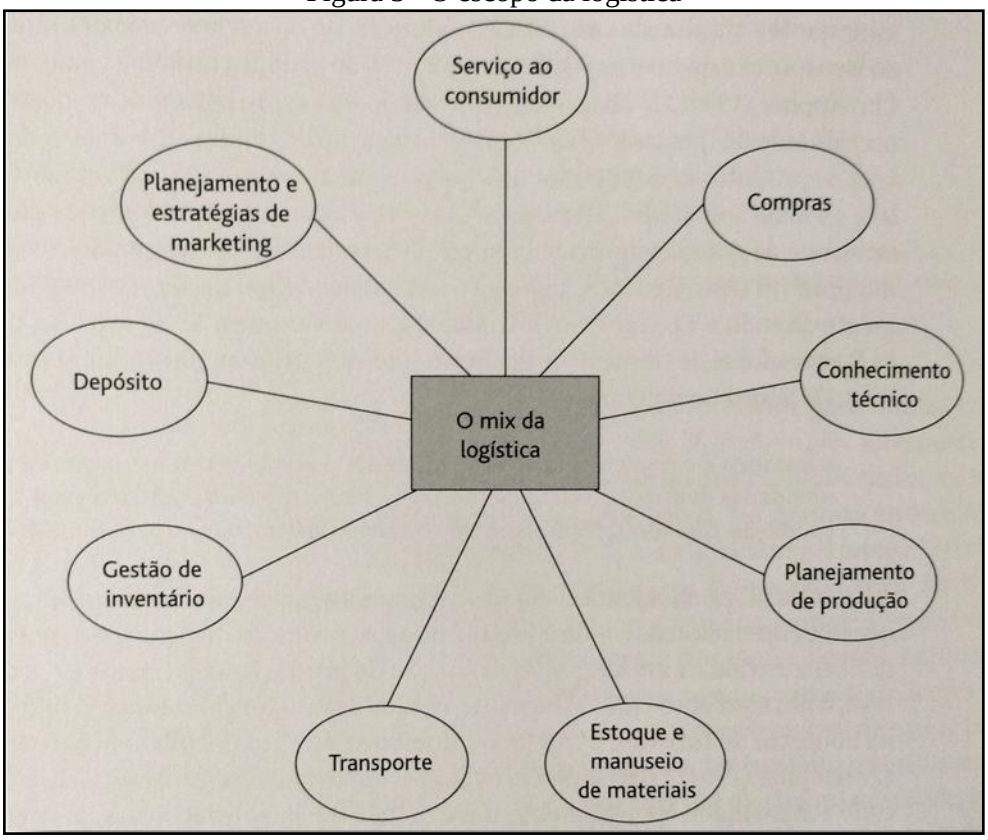
2.4 LOGISTICA E ADMINISTRAÇÃO DO TURISMO NA TIC

Na década de 90 a sociedade entrou na "era da informação" e isso causou implicações para a prestação de serviços de transporte (HEPWORTH E DUCATEL, 1992). Um dos impactos imediatos para os fornecedores de transporte turístico é que os fluxos de informações atualizadas são vitais quando existe a cadeia da oferta e que o fornecedor do transporte é apenas um dos componentes do produto turístico como um todo. A logística é um conceito vital de ser reconhecido, particularmente quando a TIC é também introduzida, uma vez que a TIC e a logística permitem que os fornecedores de transporte alcancem seus objetivos em um ambiente competitivo (PAGE, 2008). De acordo com Quayle (1993, p. 9):

> A logística é o processo que procura atender a gestão e a coordenação de todas as atividades dentro da cadeia da oferta, de terceirização à aquisição, passando pela produção, quando apropriada, e até os canais de distribuição para o consumidor. (QUAYLE, 1993. p. 9).

A logística proporciona uma vantagem competitiva, oferecendo uma visão estratégica das questões operacionais e uma compreensão das conexões dentro do sistema da oferta. Ela também auxilia na coordenação da função de prestação de serviços e o transporte, por si só é um elemento vital da logística, pois desloca o consumidor para perto do produto, no contexto do turismo. A figura 3 define as funções de negócio que estão sob o domínio da logística. O que emerge da Figura 3 é que os fluxos de informações são componentes críticos na logística e na gestão da oferta do transporte.

Figura 3 - O escopo da logística



Fonte: PAGE, Stephen J. Transportes e Turismo: Perspectivas Globais (2008).

Na TIC, o setor turístico gera grandes volumes de informação que precisam ser processadas e utilizadas na logística. Nesse sentido, Sheldon (1997, p. 123) observa que cada reserva aérea gera 25 transações que necessitam processamento. No modelo de Sheldon (1997) de fluxo de informações em turismo, há três agentes principais envolvidos: os viajantes, os fornecedores e os intermediários. Para o propósito dessa discussão, é o fornecedor que utiliza a TIC para manusear, utilizar e gerenciar esses fluxos de informações que são de seu interesse. Sob a perspectiva do fornecedor de transporte, a informação é essencial para permitir que as organizações funcionem propriamente e para os diferentes departamentos tomarem decisões sobre os objetivos corporativos, os seus consumidores e concorrentes.

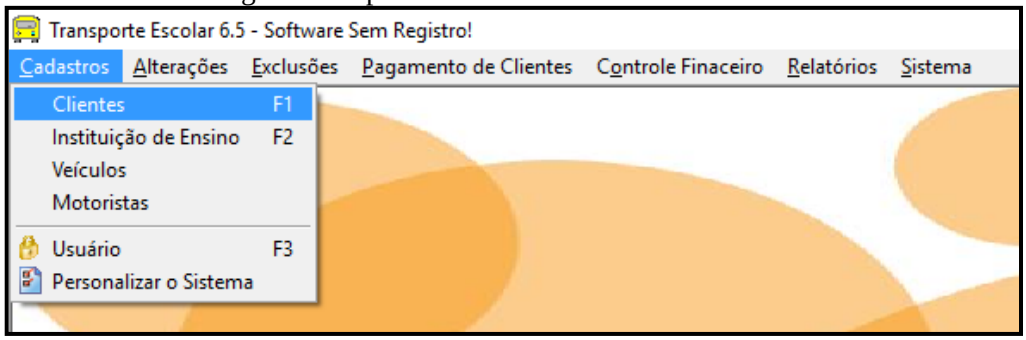
Sendo assim, a logística principalmente voltada para a TIC representa uma ótima visão de negócio em cima do turismo. Com isto, entra-se no aspecto do trabalho proposto, onde a junção de todos os conceitos já vistos incluindo a logística e a TIC são os principais requisitos presentes no trabalho em questão. Várias empresas do ramo de transporte e turismo têm buscado se adequarem às novas soluções logísticas por meio da TIC, que auxiliam na praticidade e agilidade dos serviços, como também lhes proporcionem maior rentabilidade. Diante deste cenário, para acompanhar o mercado de logística e de TIC, o presente trabalho adquire a ideia de auxiliar, automatizar, integrar e agilizar a produtividade e o funcionamento interno de viagens, das empresas e agências privadas de transporte e turismo rodoviário de passageiros.

2.5 TRABALHOS CORRELATOS

No âmbito do transporte rodoviário turístico de passageiros, não foram encontrados muitos recursos focados na área e disponíveis para auxiliar na organização, agilidade e facilidade dos serviços realizados por empresas do ramo. Embora exista no mercado algumas opções de sistemas que facilitam o cadastro e históricos de transportadoras, nas pesquisas realizadas para este trabalho não foram encontrados recursos focados em transportadoras do ramo turístico ou em transporte de passageiros, apenas em transportes rodoviários de cargas, ferroviários ou urbanos.

A maioria dos softwares encontrados no ramo turístico são focados no próprio passageiro, ou fornecem uma funcionalidade completamente distinta do âmbito do trabalho em questão. Um dos softwares que mais obteve relação com o assunto do trabalho é o Transpescolar - Software para Controle e Administração de Transporte Escolar. O mesmo foi desenvolvido para controlar as atividades cadastrais e financeiras de empresas de transporte escolar, e funciona conforme as necessidades do controle total para este tipo de transporte (PANIZIO, 2006). Na Figura 4, pode-se verificar a interface inicial do sistema Transpescolar.

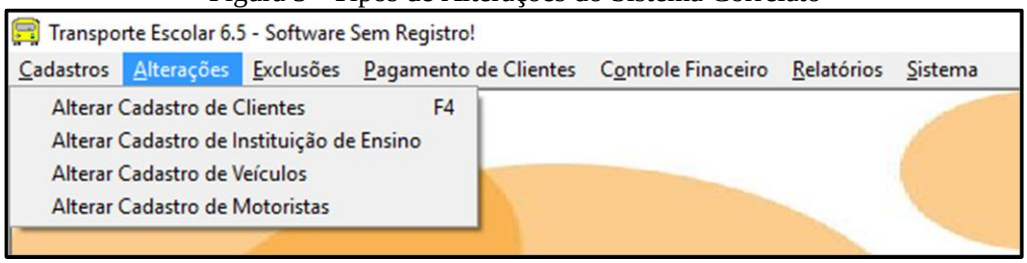
Figura 4 - Tipos de cadastros do Sistema Correlato



Fonte: PANIZIO, Jean C. Transpescolar (2006).

Apesar de o sistema correlato possuir uma interface desktop, a ideia é semelhante ao do trabalho em questão, pois fornece funcionalidades para facilitar e agilizar as tarefas e procedimentos de empresas de transporte escolar. Na Figura 5 pode-se visualizar a área de alteração dos dados já cadastrados no sistema.

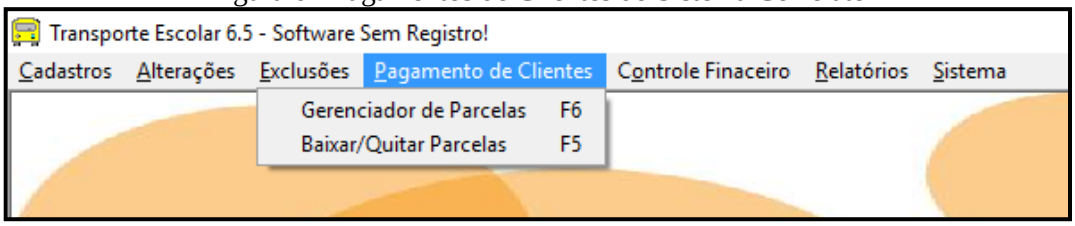
Figura 5 - Tipos de Alterações do Sistema Correlato



Fonte: PANIZIO, Jean C. Transpescolar (2006).

Em conjunto com administradores de pequenas, médias e grandes empresas, o software foi ajustado conforme as dicas de muitos voluntários e hoje a versão 6.5 parece bastante robusta, abrigando o que uma empresa necessita para controlar seus clientes. Também é possível controlar cheques, emitir lista de passageiros por veículo e por período, emitir contratos para prestação de serviços, e muitos tipos de relatórios. Nas Figuras 6 e 7 podem ser observadas as funcionalidades financeiras que o sistema disponibiliza.

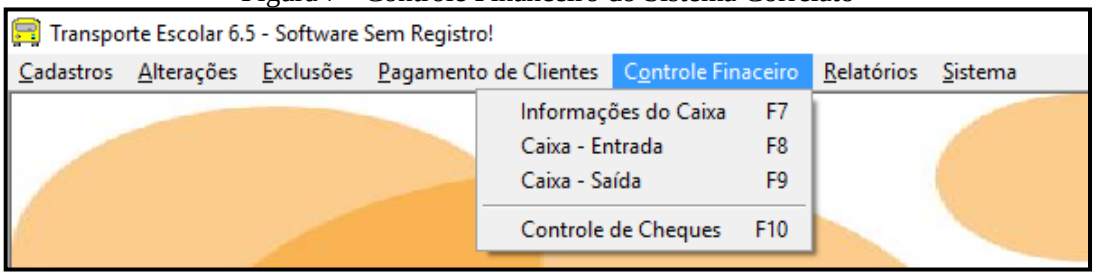
Figura 6 - Pagamentos de Clientes do Sistema Correlato



Fonte: PANIZIO, Jean C. Transpescolar (2006).

Na Figura 7 pode-se verificar um exemplo do controle financeiro do sistema correlato proposto. Neste verifica-se as `Informações do Caixa` financeiro, como também da `Entrada` e da `Saída` dos caixas. Existe também o `Controle de Cheques`.

Figura 7 - Controle Financeiro do Sistema Correlato



Fonte: PANIZIO, Jean C. Transpescolar (2006).

Diferentemente do sistema aqui proposto, onde o foco principal está em controlar e organizar dados de viagens e integração com o sistema do Deter-SC, o sistema Transporte Escolar 6.5 visa contribuir com as funcionalidades gerais de administração de uma empresa

de transporte escolar. As funcionalidades de controle financeiro são incluídas, como também diferentes tipos de relatórios e contratos para a prestação de serviço. Na Figura 8 verificam-se todas as funcionalidades do software Transporte Escolar 6.5. Deste modo, conforme descrito anteriormente é possível distinguir as funcionalidades do software proposto nesse trabalho com o sistema Transporte Escolar 6.5.

Figura 8 - Relatórios Disponíveis do Sistema Correlato

Fonte: PANIZIO, Jean C. Transpescolar (2006).

Outros softwares encontrados possuem maior nível de distinção do presente trabalho, alguns deles, por exemplo os sistemas Rodosoft (2007) e Bematech (2006) são focados apenas em vendas de viagens para os passageiros. Outros, como a TOTVS (2005) e a GSBUS (2011), possuem sistemas voltados apenas para a gestão de empresas de transportes de passageiros, em outros casos distintos, os softwares eram focados apenas em transportes de cargas ou em transporte urbano de passageiros.

Desta forma, pode-se concluir que o trabalho em questão possui certas particularidades na área do turismo onde não são encontrados tantos recursos da tecnologia da informação disponíveis e acessíveis. Na próxima seção serão abordados mais detalhes do desenvolvimento do sistema proposto.

3 DESENVOLVIMENTO

Neste capítulo estão descritos o detalhamento do sistema proposto, apresentando as suas características e particularidades técnicas, como também, será abordado o levantamento das informações incluindo a descrição dos requisitos funcionais, requisitos não funcionais e as regras de negócios do sistema. Após estes, será apresentado a especificação do sistema, incluindo o diagrama de casos de uso, os diagramas de classes, o diagrama de atividade e o modelo entidade relacionamento, como também suas breves descrições. Além disso, será descrito e demonstrado a parte prática do sistema, com a apresentação das tecnologias e ferramentas utilizadas, assim como a exemplificação por meio de códigos ou imagens do funcionamento do sistema.

3.1 SISTEMA PROPOSTO

Tendo em vista a escassez de aplicações que auxiliem na organização, produtividade e agilidade das tarefas e funcionamento interno das empresas de transporte e turismo, encontrou-se a oportunidade de criar um sistema voltado para o cadastro de viagens, motoristas, veículos, contratantes e lista de passageiros. Nesse contexto, além de auxiliar no dia a dia e nas tarefas das empresas, auxiliará também as empresas do estado de Santa Catarina na futura emissão das autorizações de viagens especiais pela integração com a Application Programming Interface (API) do sistema Deter-SC.

O sistema proposto está divido em etapas, as quais servirão para auxiliar no desenvolvimento em geral. Primeiramente foi realizado o levantamento dos requisitos, os quais se encontram logo abaixo. Após a estruturação de diagramas, foi necessário envolver o Deter-SC no ambiente do sistema e dos conceitos de análise de negócio. Para essa etapa fez-se necessário realizar visitas presenciais na empresa Deter-SC para contato e entendimento do problema e da integração do sistema em geral. Desta forma, o sistema oferecerá as seguintes funcionalidades:

a) cadastro de empresas, viagens, motoristas, veículos, contratantes, lista de passageiros e dados sobre a nota fiscal, possibilitando o registro das atividades para organizar históricos e utilizar dados úteis para futuras ações;
b) manter uma base para a futura emissão da autorização de viagem especial por meio da integração com o sistema do Deter-SC. Este tem o intuito de reduzir o tempo e o retrabalho dos donos ou funcionários das empresas de transporte e turismo;
c) visualização e busca de histórico de todas as informações cadastradas;

d) buscas e gráficos de viagens “confirmadas” ou “em espera”, as quais serão úteis para visualizar de forma ágil as próximas viagens marcadas, sejam elas confirmadas ou não;
e) notificações por e-mail referentes a tarefas em andamento ou em atraso.

3.2 LEVANTAMENTO DE INFORMAÇÕES

Nesta seção está contida a documentação que possibilitou implantar o sistema alcançando os objetivos propostos. Estão descritos a seguir os requisitos funcionais, não funcionais, regras de negócio e diagrama de atividades do sistema.

3.2.1 Especificação dos Requisitos

O Quadro 1 apresenta os requisitos funcionais previstos para o sistema e sua rastreabilidade, ou seja, vinculação com o(s) caso(s) de uso associado(s).

Quadro 1 - Requisitos Funcionais

Requisitos Funcionais	Caso de Uso
RF01: O sistema deverá permitir que o usuário realize a manutenção dos dados de sua empresa.	UC01
RF02: O sistema deverá permitir que o usuário efetue o *login* (autenticação).	UC02
RF03: O sistema deverá permitir que o usuário altere a sua senha.	UC03
RF04: O sistema deverá permitir ao usuário a manutenção de viagens.	UC04
RF05: O sistema deverá permitir ao usuário a manutenção de veículos.	UC05
RF06: O sistema deverá permitir ao usuário a manutenção de motoristas.	UC06
RF07: O sistema deverá permitir ao usuário a manutenção de contratantes.	UC07
RF08: O sistema deverá possibilitar ao usuário a manutenção da lista de passageiros.	UC08
RF09: O sistema deverá possibilitar ao usuário uma prévia dos dados necessários para a emissão de autorização de viagem especial.	UC09
RF10: O sistema deverá permitir ao usuário registrar os dados da nota fiscal.	UC10
RF11: O sistema deverá possibilitar ao usuário visualizar informações das viagens em aberto.	UC11
RF12: O sistema deverá possibilitar ao usuário visualizar informações das viagens confirmadas.	UC12
RF13: O sistema deverá possibilitar ao usuário um ambiente de consulta, por meio de gráficos ou tabelas com o histórico de viagens, veículos e motoristas.	UC13
RF14: O sistema deverá possibilitar ao usuário enviar notificações via e-mail de tarefas pendentes.	UC14

Fonte: elaborado pela autora (2016).

O Quadro 2 lista os requisitos não funcionais previstos para o sistema.

Quadro 2 - Requisitos Não Funcionais

Requisitos Não Funcionais
RNF01: O sistema deverá ser desenvolvido em ambiente Java integrado ao Play Framework e ao Spring, com o auxílio das linguagens de programação HTML, CSS e JavaScript unidos ao framework AngularJS.
RNF02: O sistema deverá utilizar o banco de dados PostgreSQL.
RFN03: O sistema deverá ser acessível via Safari (versão 7 ou superior), Mozilla Firefox (versão 28 ou superior), Google Chrome (versão 33 ou superior), Microsoft Edge.
RNF04: O sistema deve possuir um design responsivo a todos os tipos de layouts existentes no mercado.

Fonte: elaborado pela autora (2016).

3.2.2 Regras de Negócio

O Quadro 3 possui as descrições de doze regras de negócio previstas para o sistema.

Quadro 3 - Regras de Negócio

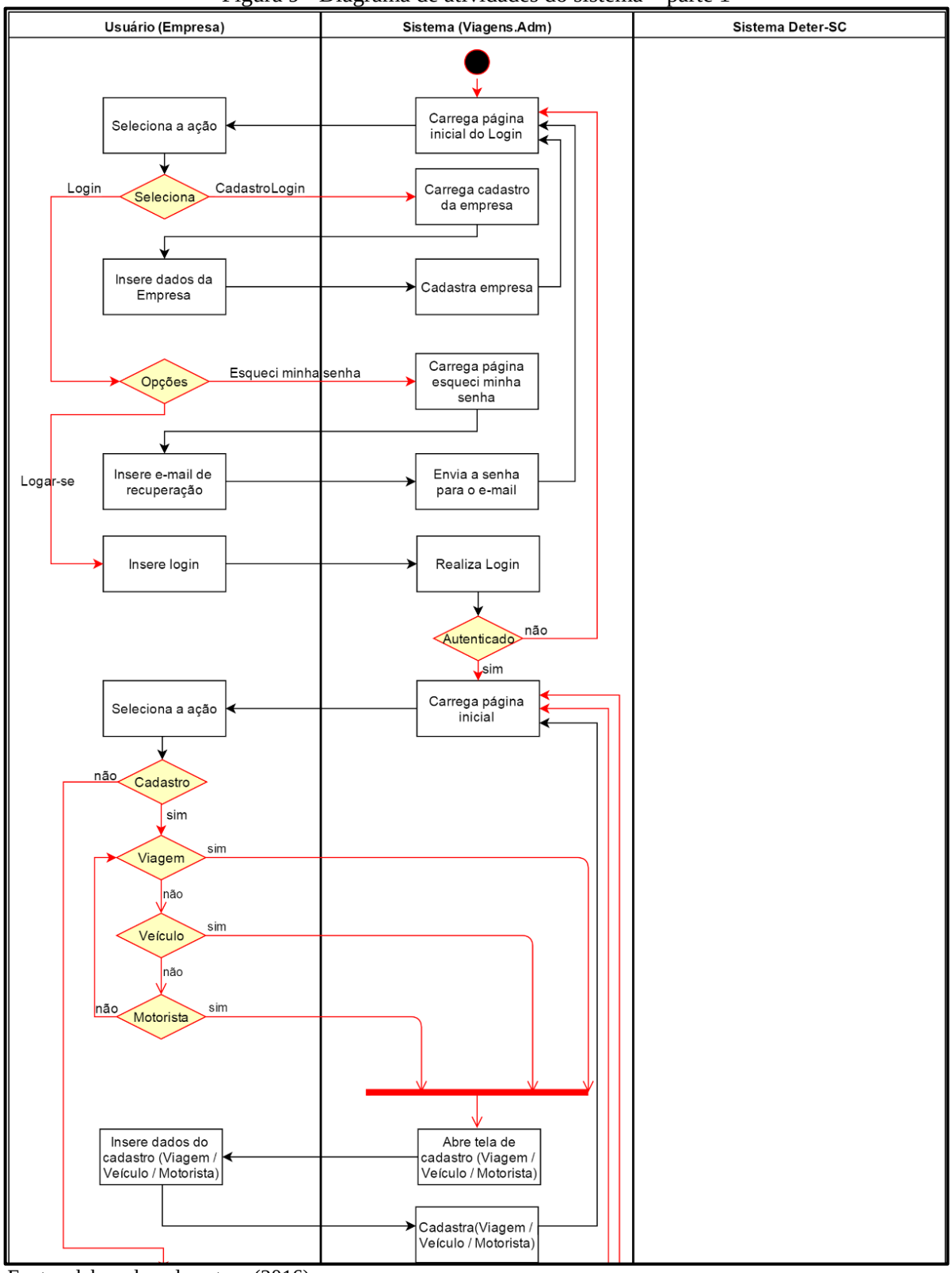
Regras de Negócio
RN01: A senha da empresa deverá possuir mais de 6 (seis) dígitos.
RN02: Uma empresa poderá manter um ou mais veículos, motoristas ou viagens relacionados.
RN03: A viagem poderá ser cadastrada apenas no instante em que um veículo, motorista ou contratante forem cadastrados antecipadamente.
RN04: Uma viagem possuirá apenas um veículo, um motorista, um contratante, uma nota fiscal, uma autorização de viagem especial ou uma empresa relacionado a mesma.
RN05: As rotas/trechos, os dados da nota fiscal e a lista de passageiros poderão ser cadastrados apenas após a viagem ser salva.
RN06: Uma rota/trecho estará relacionada apenas a uma viagem.
RN07: Uma viagem possuirá uma ou mais rotas/trechos relacionados ao mesmo.
RN08: Um passageiro poderá participar de uma ou mais viagens.
RN09: Uma viagem possuirá apenas uma lista de passageiros.
RN10: A visualização de todos os dados necessários para gerar a autorização de viagem especial poderá ser gerada apenas no momento em que o status da viagem estiver como “confirmado” e todos os cadastros subsequentes estiverem salvos no banco de dados. (Cadastros subsequentes: Veículo, Motorista, Viagem, Rota, Lista de Passageiros, Dados da Nota Fiscal).
RN11: A cidade da rota/trecho deverá ser selecionada apenas após a escolha do determinado estado.
RN12: A alteração de senha da empresa será realizada apenas no instante em que a senha atual e os dois campos das futuras senhas forem comparados e validados.

Fonte: elaborado pela autora (2016).

3.2.3 Diagrama de Atividades

O diagrama de atividades descreve determinadas etapas principais do processo de um sistema. Este normalmente é mais utilizado na facilitação da comunicação com o cliente dos projetos. Na Figura 9 pode-se analisar a primeira parte do diagrama de atividades do funcionamento do sistema.

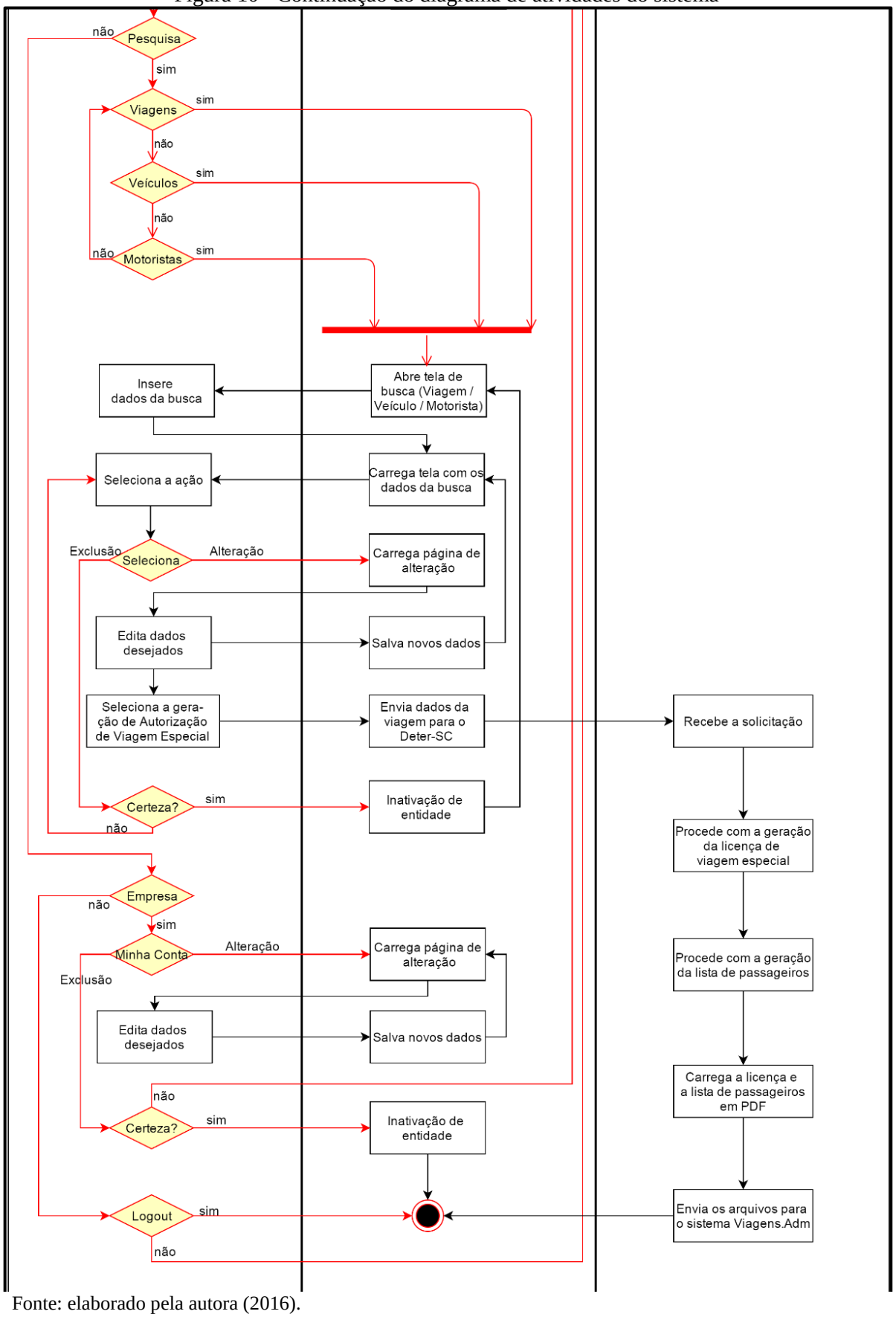
Figura 9 - Diagrama de atividades do sistema – parte 1



Fonte: elaborado pela autora (2016).

Na Figura 10 pode-se visualizar a continuação do diagrama de atividades do sistema proposto.

Figura 10 - Continuação do diagrama de atividades do sistema



Fonte: elaborado pela autora (2016).

O completo diagrama de atividade demonstrado anteriormente é de suma importância para o entendimento geral do processo, principalmente na visão do usuário. Neste pode-se visualizar os três atores principais que participam de todo o processo, sendo eles o `Usuário (Empresa)`, `Sistema (Viagens.Adm)` e `Sistema Deter-SC`. Este último participa temporariamente de todo o processo e é ativado pelo usuário apenas no instante em que o mesmo deseja emitir a licença de viagem especial, ou seja, quando a extensão do trabalho que prevê a integração com o sistema do DETER-SC estiver cumprida.

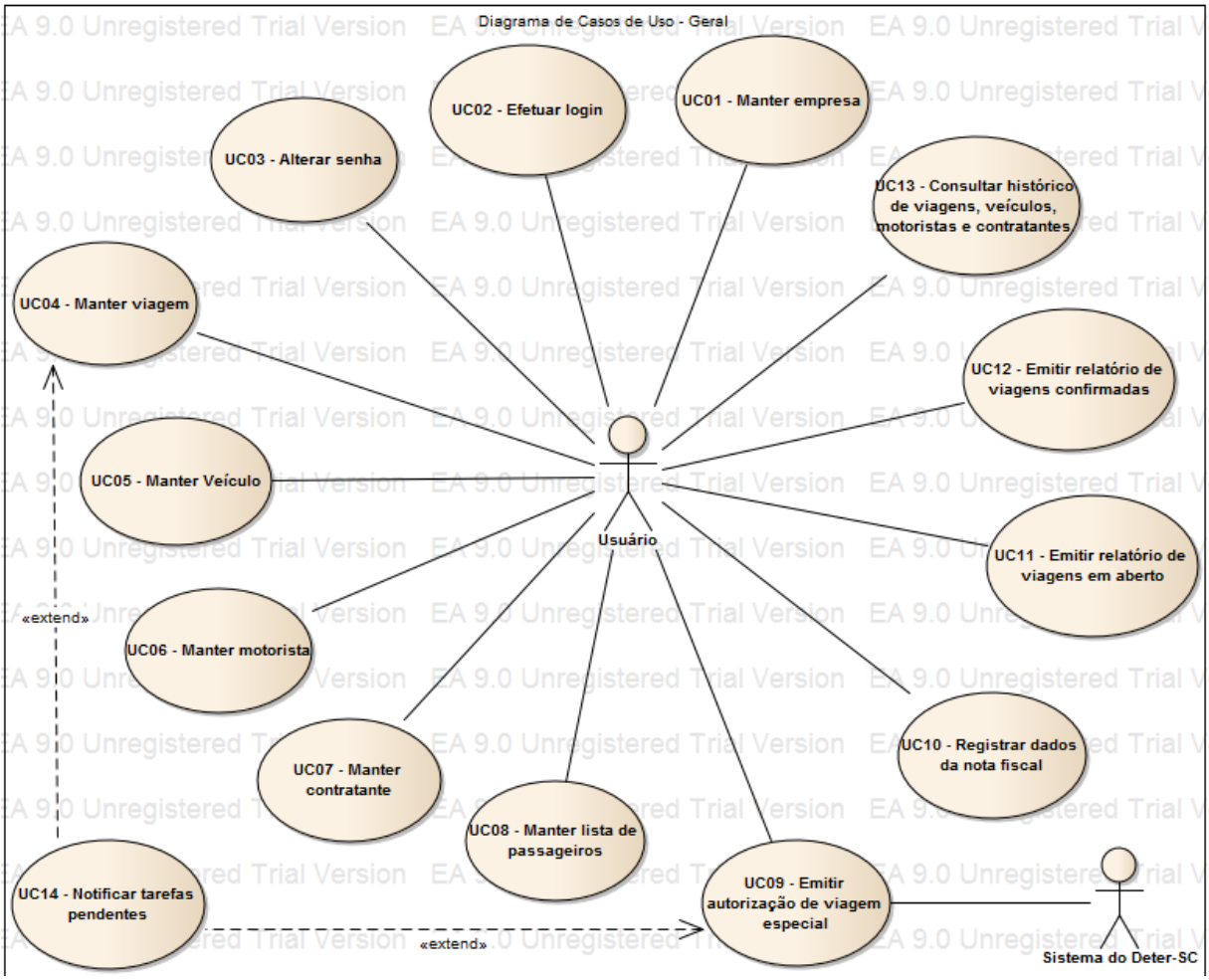
3.3 ESPECIFICAÇÃO

Nesta seção são apresentados os diagramas necessários para o entendimento do sistema proposto. A especificação deste trabalho foi desenvolvida através do diagrama de casos de uso, dois diagramas de classes e o modelo de entidade relacionamento, todos baseados nos padrões UML. Os mesmos foram criados nas ferramentas Draw.io e MySQL Workbench.

3.3.1 Diagrama de Casos de Uso

Esta seção apresenta o diagrama de casos de uso do sistema proposto, composto por quatorze casos de uso e dois atores distintos. O `Usuário` poderá ser representado pelo dono ou funcionário da empresa e será responsável pela a administração geral do sistema, e o Sistema do DETER-SC que será responsável pela futura integração da emissão da autorização de viagem especial. No apêndice A encontram-se descritos o detalhamento de todos os casos de uso que podem ser visualizados na Figura 11.

Figura 11 - Diagrama de casos de uso



Fonte: elaborado pela autora (2016).

O caso de uso `manter viagem` é citado como um dos principais deste sistema pelo fato de que para realizar o seu cadastro completo é necessário a criação antecipada de todos os outros cadastros subsequentes. Nesse conceito, para que uma viagem seja cadastrada por completo é preciso possuir o cadastro da empresa e, após isso, deve-se cadastrar um veículo e um motorista que realizará esta viagem. Além disso, o contratante da viagem deve ser cadastrado anteriormente e os dados da viagem serão cadastrados com sucesso. Após estes, é possível cadastrar a rota, os dados da nota fiscal e a lista de passageiros desta viagem.

O caso de uso, `manter viagem`, é detalhado no Quadro 4.

Quadro 4 - Detalhamento do UC04 - Manter Viagem

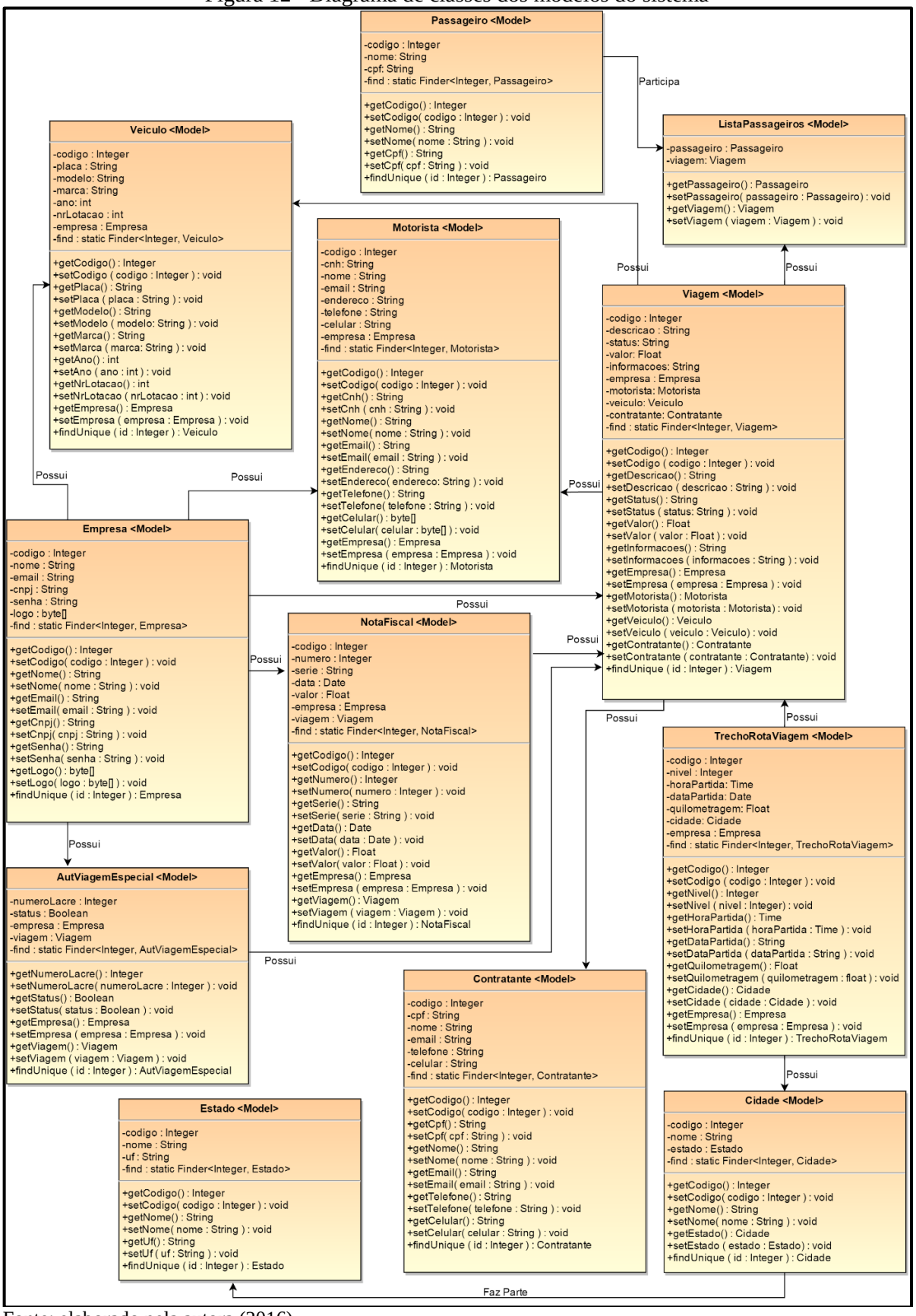
Caso de Uso	**UC04 - Manter Viagem**
Descrição	Permite ao usuário cadastrar os dados da viagem no sistema, bem como editar ou excluir informações da mesma. Além disso, o usuário pode visualizar suas informações já cadastradas.
Pré-condição	Usuário deve realizar *login* no sistema.
Pós-condição	Usuário incluiu, alterou ou excluiu dados da viagem.
Cenário - Cadastro	1) Usuário acessa a página de cadastro de novas viagens; 2) Usuário preenche todos os campos obrigatórios referentes a viagem; 3) Usuário seleciona a opção “Salvar”; 4) Sistema inclui a viagem e apresenta a mensagem “Cadastro efetuado com sucesso.”;
Cenário - Edição	1) Usuário acessa a página de edição de determinada viagem; 2) Usuário preenche novamente todos os campos que ele deseja alterar; 3) Usuário seleciona a opção “Salvar”; 4) Sistema inclui as alterações e apresenta a mensagem “Edição efetuada com sucesso.”;
Cenário - Exclusão	1) Usuário acessa a página de exclusão de determinada viagem; 2) Usuário seleciona a opção “Excluir”; 3) Sistema apresenta uma mensagem de confirmação da ação selecionada. 4) Caso a resposta seja positiva, o sistema exclui a viagem e apresenta a mensagem “A viagem foi excluída com sucesso.”;

Fonte: elaborado pela autora (2016).

3.3.2 Diagrama de Classes

A seguir é possível verificar os diagramas de classes do sistema. Foram criados dois diagramas como forma de aprimoramento e facilitação no entendimento do sistema proposto. Na Figura 12 é possível avaliar o diagrama de classes dos modelos bases do sistema.

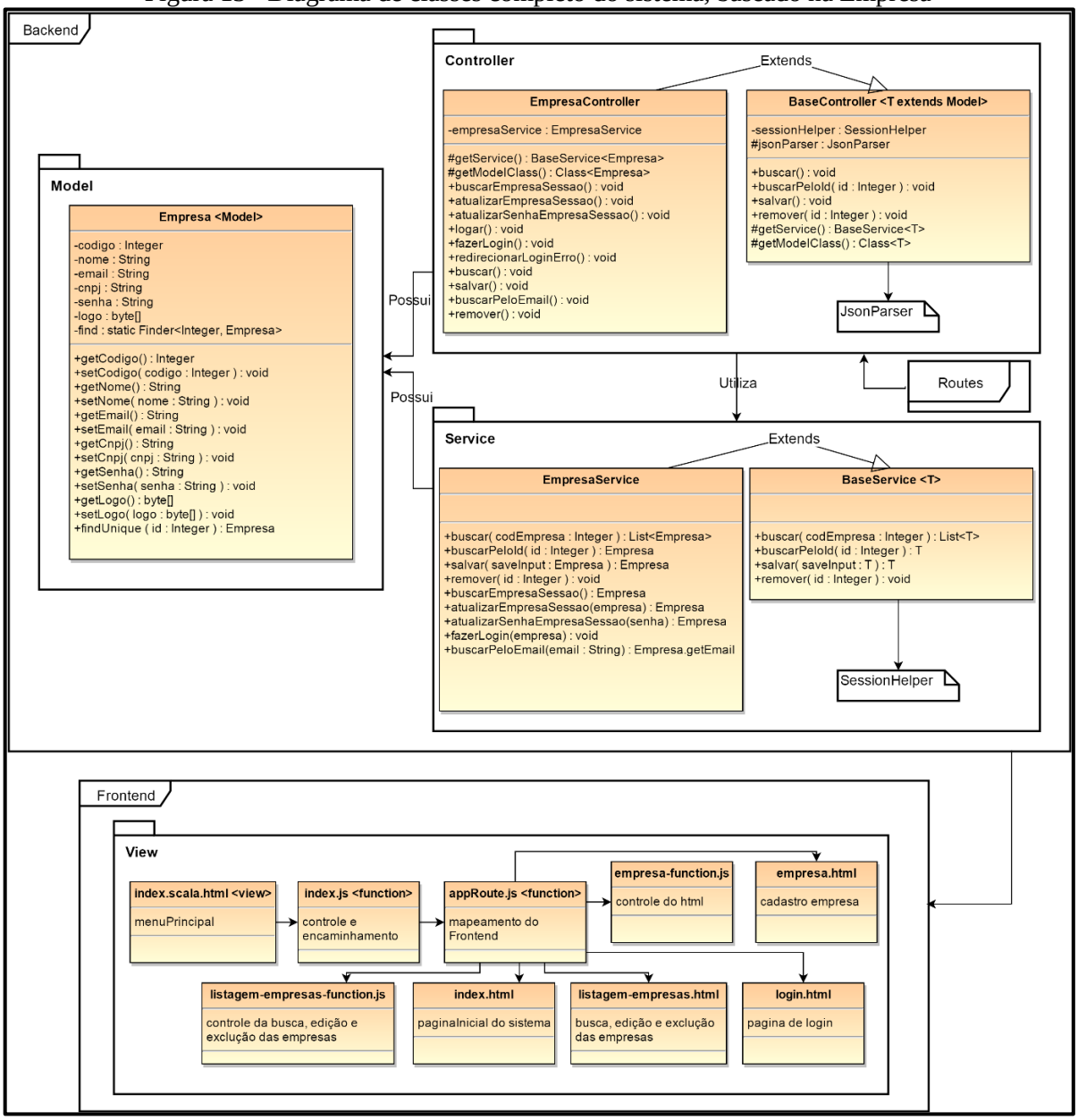
Figura 12 - Diagrama de classes dos modelos do sistema



Fonte: elaborado pela autora (2016).

Na Figura 12 é possível visualizar as classes de uma maneira superficial, apenas com os modelos. Desta forma, foi criado um diagrama de classes adicional exemplificando o funcionamento completo do sistema, baseando-se na empresa. Na Figura 13 o mesmo pode ser verificado.

Figura 13 - Diagrama de classes completo do sistema, baseado na Empresa



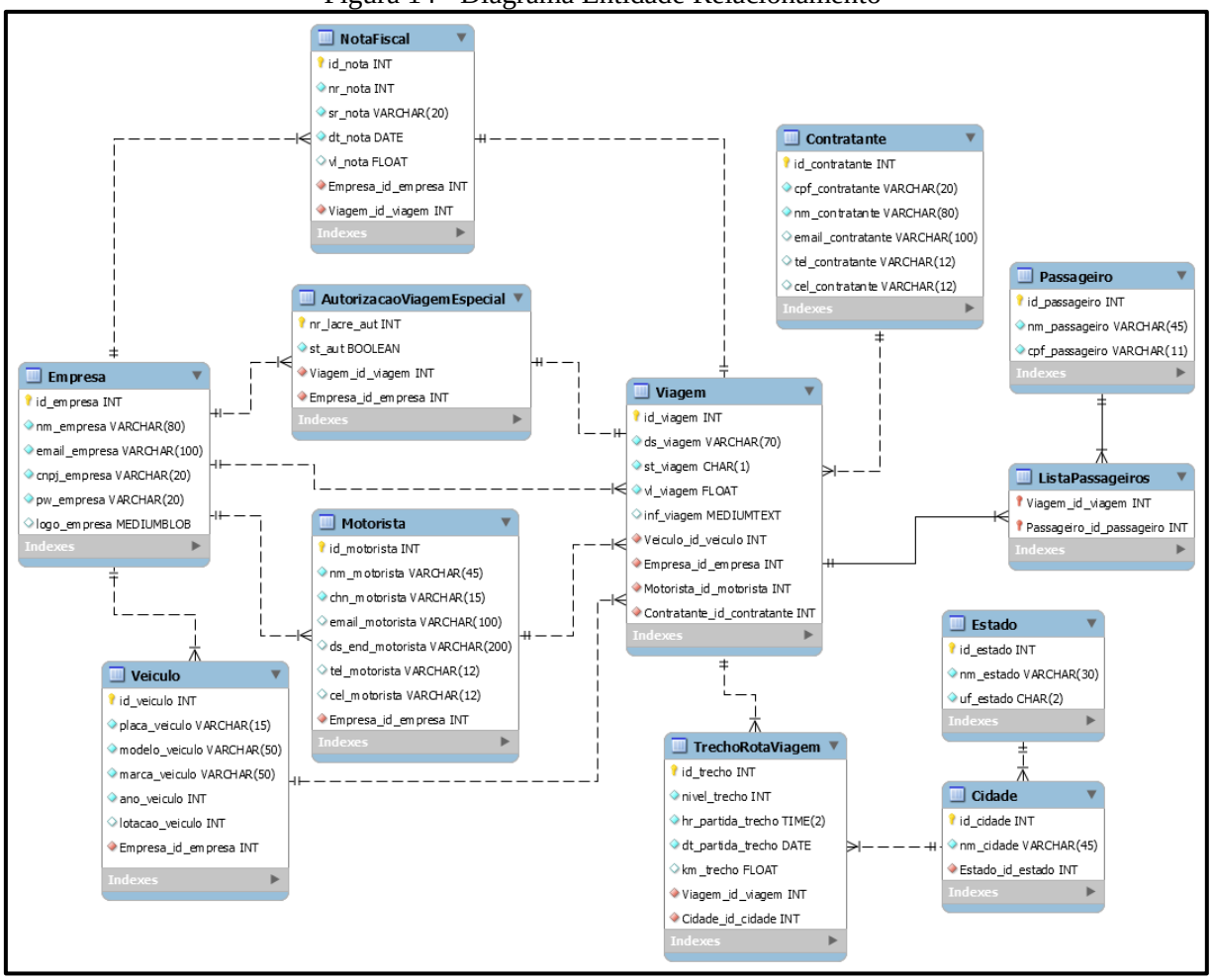
Fonte: elaborado pela autora (2016).

No diagrama de classes baseado na empresa, pode-se compreender o funcionamento e a comunicação geral das classes do sistema. Neste possui-se a divisão do *front-end* que administra e compila todas as interfaces (*views*) do sistema e do *back-end* que controla, modela e monitora os serviços de todos os dados ou comandos enviados pelo *front-end*. Mais informações sobre a comunicação detalhada das classes e das dependências serão descritas no capítulo da implantação.

3.3.3 Diagrama Entidade Relacionamento

O diagrama de entidade relacionamento (DER) é um modelo diagramático que descreve o modelo de dados do sistema, também conhecido como Modelo Entidade Relacionamento (MER). Na Figura 14 pode-se verificar o diagrama de entidade relacionamento completo do sistema proposto. No Apêndice B encontram-se descritos os quadros com o detalhamento de todas as entidades visualizadas na Figura 14.

Figura 14 - Diagrama Entidade Relacionamento



Fonte: elaborado pela autora (2016).

No diagrama acima representado pode-se analisar as entidades do sistema, suas dependências e seus atributos. Também pode-se perceber que a partir do sistema proposto a entidade `Empresa` dispõe da comunicação para abordar as demais entidades. Em segundo plano, mas não menos importante, pode-se verificar a `Viagem`. Esta se destaca como a principal de todo o processo de comunicação com as demais entidades, pois, a partir dela é possível cadastrar todas as outras dependências, restringindo-se apenas a própria `Empresa`. As entidades `Veículo`, `Motorista` e `Contratante` devem ser cadastradas antecipadamente a uma `Viagem`. A partir disto o `TrechoRotaViagem` poderá ser cadastrado no sistema. Este mantém

todas as rotas que uma `Viagem` possuirá, mantendo também um relacionamento direto com as entidades `Cidade` e `Estado`. Após o cadastro de todas as entidades anteriormente citadas, a `NotaFiscal`, os `Passageiros` e a `ListaPassageiros` poderão ser cadastrados no sistema. A possível integração com o sistema do DETER-SC e a relação com a entidade `AutorizaçãoViagemEspecial` seriam realizadas a partir deste ponto.

A seguir verifica-se com mais detalhes o modo em que essas entidades de fato funcionam na prática do sistema implementado.

3.4 IMPLEMENTAÇÃO

Nesta seção são descritas as técnicas e ferramentas utilizadas para o desenvolvimento do sistema, como também a operacionalidade da implementação.

3.4.1 Técnicas e ferramentas utilizadas

As técnicas e ferramentas utilizadas para a implementação do trabalho são descritas a seguir, juntamente com a explicação de algumas tecnologias utilizadas e a exemplificação dos principais códigos necessários para usufruir de forma correta destas tecnologias. Antes de qualquer demonstração ou explicação do funcionamento do sistema, será descrito de forma breve cada tecnologia utilizada no desenvolvimento do trabalho.

Para a construção e desenvolvimento desde trabalho foram utilizadas as seguintes ferramentas:

a) Eclipse Luna como plataforma de desenvolvimento;
b) PostgreSQL como banco de dados para armazenamento das informações;
c) MySQL Workbench para a criação do diagrama entidade-relacionamento;
d) Draw.io como ferramenta online para a criação dos diagramas de classe e atividade;
e) Java como linguagem de programação do *back-end* da aplicação;
f) Play, Spring, Bootstrap e AngularJS utilizados como *frameworks* para a base completa do sistema, sendo estes um: servidor, integrador das linguagens de programação e modelador de técnicas do desenvolvimento completo do sistema;
g) HTML 5, CSS 3 e JavaScript utilizados como linguagem de programação para o desenvolvimento do *front-end* da aplicação;
h) GulpJS utilizado como automatizador de tarefas para os códigos do *front-end* da aplicação.

Alguns destes *frameworks* e tecnologias citados anteriormente são explicados e exemplificados a seguir.

3.4.1.1 Play *Framework*

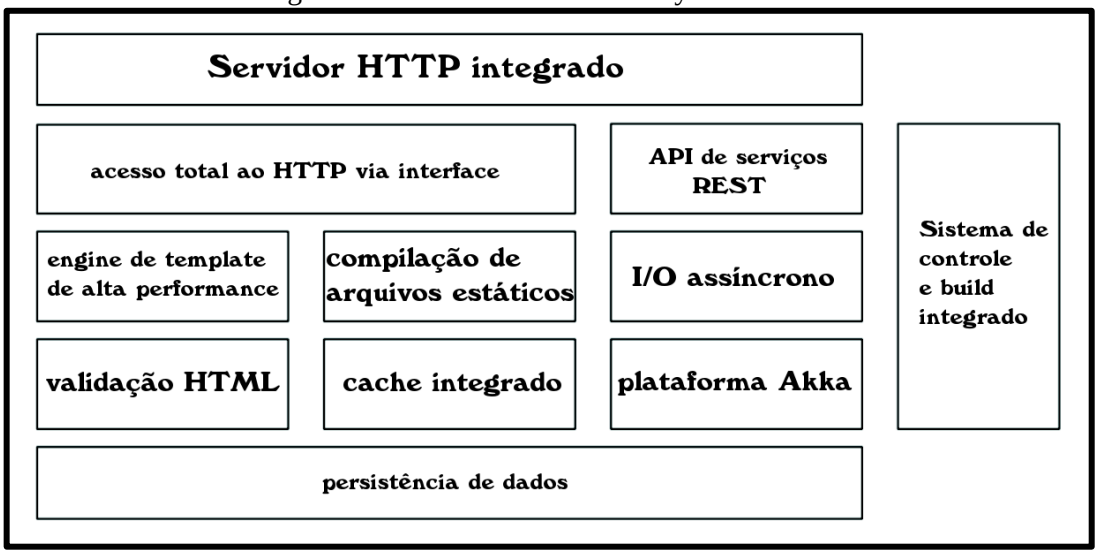
O Play é um *framework* que redefine o desenvolvimento web em Java. Este não é um *framework* padrão Java Enterprise Edition (JavaEE) como é o exemplo do Spring ou outros, neste caso o Play utiliza uma arquitetura extremamente simples. A diferença com os demais exemplos de *frameworks* é que ele possui apenas duas camadas que são: o próprio Play *framework* e o seu HyperText Transfer Protocol (HTTP) server embutido.

Para o desenvolvimento deste trabalho o Play foi optado, pois o seu foco é o desenvolvimento no qual a interface HTTP torna-se simples e flexível. Além de focar na produtividade do desenvolvedor Representational State Transfer (REST), ele também é uma solução completa que envolve persistência e muito mais recursos, como:

a) servidor HTTP integrado;
b) acesso completo à interface HTTP;
c) API de serviços REST;
d) manutenção de códigos apenas com a atualização na página;
e) compilação dos arquivos estáticos da aplicação;
f) persistência de dados.

Na Figura 15 é possível verificar outros recursos disponibilizados pelo Play *framework.*

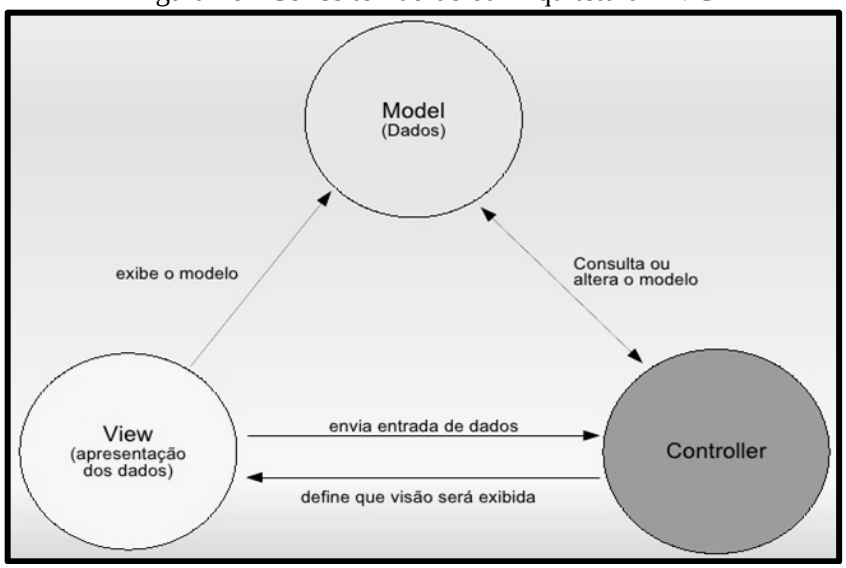
Figura 15 - Funcionalidades do Play *Framework*



Fonte: Apostila Play Framework (2016).

A partir destas funcionalidades e recursos normalmente utilizadas pelos desenvolvedores atuais, o Play foi estudado e aplicado neste trabalho. O foco do desenvolvimento foi Java e o próprio Play mantém o padrão de arquitetura Model View Controller (MVC). Na Figura 16 pode-se visualizar e entender melhor o funcionamento da comunicação entre as classes do projeto por meio do conceito do MVC.

Figura 16 - Conceito Padrão da Arquitetura MVC



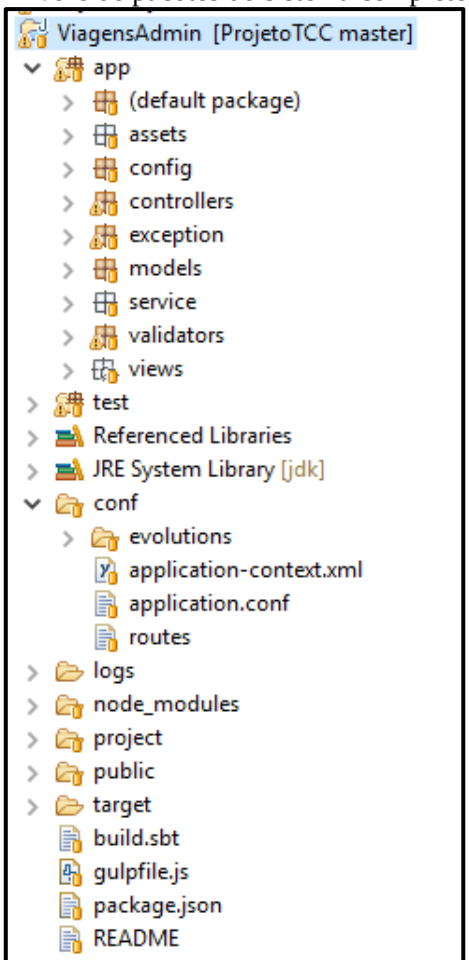
Fonte: elaborado pela autora (2016).

Para compreender o nível dos detalhes o desenvolvimento deste trabalho no padrão da arquitetura MVC, recomenda-se entender os conceitos e tipos de desenvolvimento *front-end* e *back-end*. Estes podem ser resumidos de modo simples, o *front-end* também é conhecido por linguagem de frente, é o responsável por coletar a entrada do usuário de várias formas por meio da interface do sistema e processá-la de uma maneira que adéque o código para que o *back-end* possa usufruir do mesmo. Desta forma, o *back-end*, também conhecido por linguagem de trás, dinamiza e controla todas as informações recebidas e retorna as respostas necessárias ao *front-end*.

Neste trabalho o conceito MVC foi utilizado de forma diferenciada da convencional, mas conhecida por desenvolvedores atualizados no mercado. As classes `models` fazem parte apenas do *back-end* na teoria, mas são utilizadas tanto pelo *front-end* quanto pelo *back-end*, com o auxílio e integração dos *frameworks* AngularJS e Spring. As classes `controllers` fazem parte apenas do *back-end* e a partir destas são chamadas as classes `services`, também conhecidas por Data Access Object (DAO). Todo o *back-end* do projeto é mantido em linguagem Java. No *front-end* do projeto, conhecido como `views` ou interfaces do sistema, é possível aplicar praticamente a mesma teoria do MVC, pois neste caso para cada tela, existem

as interfaces em HTML, e os `controllers` e `services` em *JavaScript*. Na Figura 17 pode-se analisar a árvore de pacotes do sistema completo.

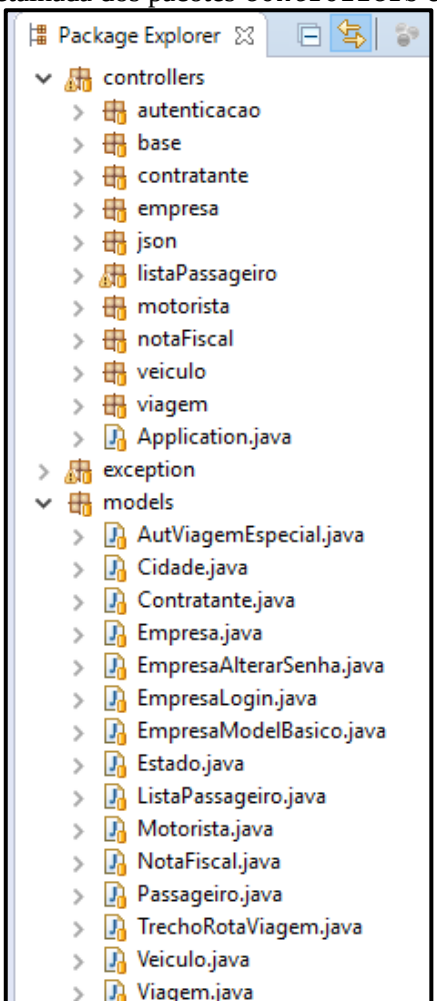
Figura 17 - Árvore de pacotes do sistema completo no Eclipse



Fonte: elaborado pela autora (2016).

Na Figura 17 pode-se verificar a maneira em que os pacotes do sistema são divididos e compreender melhor as divisões de cada pacote já citado anteriormente. O `app` é o pacote que comanda todos os códigos do sistema, sendo eles *front-end* ou *back-end*. O pacote `assets` mantém todos os códigos *front-end* e os demais pacotes fazem parte do *back-end*. A partir dos `models` cria-se também a base de dados PostgreSQL com o auxílio do Hibernate. O Hibernate disponibilizado pelo próprio Play é um *framework* que realiza o mapeamento do objeto relacional, neste caso os `models`. O seu principal objetivo é diminuir a complexidade envolvida no desenvolvimento de aplicações que trabalham com banco de dados relacionais e realizar a intermediação entre o banco de dados e o sistema proposto. Na Figura 18 pode-se verificar também a árvore detalhada com as classes `controllers` e `models` do sistema.

Figura 18 - Árvore detalhada dos pacotes `controllers` e `models` do sistema



Fonte: elaborado pela autora (2016).

Pode-se analisar na Figura 18 as classes e pacotes do *back-end* da aplicação. Como citado anteriormente os `models` possuem extrema importância, pois além de servirem de base e padrão para todo o sistema, criam as dependências do Hibernate para integrar-se ao banco de dados. Para exemplificar este funcionamento pode-se analisar o Quadro 5 que demonstra o `model Empresa` do sistema. A partir deste quadro, verifica-se que o comando `@Entity` (linha 12) mostra ao Play que esta classe estendida do `model` será uma entidade principal. Com o comando `@Table(name = "Empresa")` (linha 13) o Hibernate a partir do próprio *framework* Play, consegue compreender e compilar que a entidade é uma tabela no banco de dados com o nome `Empresa`. Observa-se também que acima das variáveis existem comandos como o `@Id` (linha 25), que informa ao Hibernate que a variável abaixo é a *primary key* da tabela, e o comando `@Column` (linha 30), que pode manter informações importantes como o nome, tamanho e valores das variáveis.

Quadro 5 - Exemplificando o `model Empresa` do sistema

Empresa.java

```
1  package models;
2
3  import javax.persistence.Column;
11
12 @Entity
13 @Table(name = "Empresa")
14 public class Empresa extends Model {
15
16     private static final long serialVersionUID = 1L;
17
18     public static final String ID_EMPRESA = "id_empresa";
19     private static final String NM_EMPRESA = "nm_empresa";
20     private static final String EM_EMPRESA = "email_empresa";
21     private static final String CNPJ_EMPRESA = "cnpj_empresa";
22     private static final String PW_EMPRESA = "pw_empresa";
23     private static final String LOGO_EMPRESA = "logo_empresa";
24
25     @Id
26     @GeneratedValue(strategy=GenerationType.IDENTITY)
27     @Column(name = ID_EMPRESA)
28     private Integer codigo;
29
30     @Column(name = NM_EMPRESA, nullable=false, length=80)
31     private String nome;
32
33     @Column(name = EM_EMPRESA, nullable=false, length=100)
34     private String email;
35
36     @Column(name = CNPJ_EMPRESA, nullable=false, length=20)
37     private String cnpj;
38
39     @Column(name = PW_EMPRESA, nullable=false, length=20)
40     private String senha;
```

Fonte: elaborado pela autora (2016).

Existem outros tipos de comandos importantes para descrever às funcionalidades do Hibernate, uma delas é o `@GenerateValue` (linha 26), o qual informa que a variável seguinte será reconhecida como `AUTO_INCREMENT` no banco de dados. Outro comando importante, mas que não consta no Quadro 5 é o que informa ao Hibernate que a variável é uma *foreign key*, este comando pode ser descrito por `@ManyToOne` ou `@OneToOne`, e também está sendo utilizado neste trabalho em `models` como `Viagem`, `Veiculo`, `Motorista`, `Rotas`, etc. A partir destes modelos cria-se o banco de dados completo do sistema. Por este e outros motivos o Play foi considerado uma ótima opção dentre os `frameworks` do mercado escolhidos para realizar o desenvolvimento deste sistema.

A partir do que foi mostrado anteriormente, pode-se compreender o funcionamento do sistema focado na estruturação dos dados. Para haver uma compreensão detalhada de como o *front-end* se comunica com o back-*end* do sistema, deve-se analisar a classe `routes` no pacote `conf` da estruturação do Play Framework. No Quadro 6 pode-se conferir o funcionamento das rotas no arquivo `routes` do sistema proposto.

Quadro 6 - Arquivo `routes` do sistema

routes

```
1  # Página pricipal
2  GET     /login                      @controllers.Application.telaLogin()
3  GET     /                           @controllers.Application.inicio()
4
5  # Login
6  POST    /login                      @controllers.empresa.EmpresaController.logar()
7
8
9  # Map static resources from the /public folder to the /assets URL path
10 GET     /assets/*file               controllers.Assets.at(path="/public", file)
11
12 # Empresa
13 POST    /empresa/salvar                                  @controllers.empresa.EmpresaController.salvar()
14 GET     /empresa/sessao                                  @controllers.empresa.EmpresaController.buscarEmpresaD
15 POST    /empresa/sessao/atualizar                        @controllers.empresa.EmpresaController.atualizarEmpre
16 POST    /empresa/sessao/atualizar/senha                  @controllers.empresa.EmpresaController.atualizarSenha
17
18 #Motorista
19 POST    /motorista/salvar                                @controllers.motorista.MotoristaController.salvar()
20 GET     /motorista/buscar                                @controllers.motorista.MotoristaController.buscar()
21 GET     /motorista/buscar/:codMotorista                  @controllers.motorista.MotoristaController.buscarPelo
22 DELETE  /motorista/remover/:codMotorista                 @controllers.motorista.MotoristaController.remover(co
```

Fonte: elaborado pela autora (2016).

Como se pode analisar no Quadro 6, o padrão do arquivo `routes` é citar primeiramente o método HTTP da rota (exemplo na linha 2), sendo os principais métodos `GET`, `POST` ou `DELETE`. Após este deve-se informar o caminho Uniform Resource Identifier (URI) da rota HTTP e o próximo passo é informar para a rota o determinado método `controller` responsável por realizar os serviços desejados. Um dos exemplos iniciais e importantes é no momento em que o usuário acessa a página inicial da aplicação. Neste instante o arquivo `routes` informa para a rota que o método `controller` responsável por tomar esta ação é o `@controllers.Application.inicio()`. No Quadro 7 é possível verificar o `controller Application.java`.

Quadro 7 - Exemplificando classe `controller Application.java`

Application.java

```
1  package controllers;
2
3  import org.springframework.stereotype.Service;
10
11 @Service
12 public class Application extends Controller {
13
14     @Autenticado
15     public Result inicio() {
16         String nomeEmpresa = session(SessionHelper.SESSION_EMPRESA_NOME);
17         return ok(views.html.menuPrincipal.render(nomeEmpresa));
18     }
19
20     public Result telaLogin() {
21         session().clear();
22         Content page = views.html.login.render();
23         if (possuiErroDeValidacao()) {
24             return unauthorized(page);
25         }
26         return ok(page);
27     }
28
29     private boolean possuiErroDeValidacao() {
30         return flash("erro") != null;
31     }
```

Fonte: elaborado pela autora (2016).

No Quadro 7 visualiza-se o método `inicio()` (linha 15) citado no último parágrafo. Nele pode-se verificar que antes mesmo da sua execução é realizada uma autenticação por meio do comando `@Autenticado` (linha 14), que analisa se algum usuário ou empresa consta na sessão do sistema. Caso não possua uma `Empresa` na sessão, existe um método que envia o a rota `/login` para o arquivo `routes` e este método `controller` faz com que a página de `Login` seja aberta para o usuário. Caso o usuário esteja autenticado no sistema o método `inicio()` adicionará os dados necessários na sessão e abrirá a página inicial referente a `Empresa` desejada.

Para abrir qualquer página HTML no Play é necessário repassar pela classe padrão `main.scala.html` localizada no pacote `views`. Esta classe é importante para a compreensão do funcionamento do Play, pois a partir dela serão importados todos os *frameworks* utilizados no sistema proposto. No Quadro 8 é possível visualizar a classe `main.scala.html`.

Quadro 8 - Exemplificação da classe `main.scala.html` do sistema Viagens.Adm

main.scala.html

```
@(angularModuleName : String)(content: Html)
<!DOCTYPE html>

<html>

<head>
    <meta charset="utf-8">
    <meta http-equiv="X-UA-Compatible" content="IE=edge">
    <meta name="viewport" content="width=device-width, initial-scale=1">
    <title>Viagens.Adm</title>
    <link rel="stylesheet" href="@routes.Assets.at("bower_components/font-awesome-4.6.3/css/font-awesome.min.css")">
    <link rel="stylesheet" href="@routes.Assets.at("bower_components/bootstrap/dist/css/bootstrap.min.css")">
    <link rel="stylesheet" href="@routes.Assets.at("bower_components/angular-bootstrap-datetimepicker/src/css/dateti
    <link rel="stylesheet" href="@routes.Assets.at("bower_components/angular-chart.js/dist/angular-chart.css")">
    <link rel="stylesheet" href="@routes.Assets.at("bower_components/sweetalert/lib/sweet-alert.css")">
    <link rel="stylesheet" href="@routes.Assets.at("bower_components/swal-forms-master/swal-forms.css")">
    <link rel="stylesheet" href="@routes.Assets.at("stylesheets/main.css")">

    <script src="@routes.Assets.at("bower_components/jquery/dist/jquery.min.js")"></script>
    <script src="@routes.Assets.at("bower_components/bootstrap/dist/js/bootstrap.min.js")"></script>
    <script src="@routes.Assets.at("bower_components/angular/angular.min.js")"></script>
    <script src="@routes.Assets.at("bower_components/angular-i18n/angular-locale_pt-br.js")"></script>
    <script src="@routes.Assets.at("bower_components/angular-route/angular-route.min.js")"></script>
    <script src="@routes.Assets.at("bower_components/sweetalert/lib/sweet-alert.min.js")"></script>
    <script src="@routes.Assets.at("bower_components/swal-forms-master/swal-forms.js")"></script>
    <script src="@routes.Assets.at("bower_components/moment/moment.js")"></script>
    <script src="@routes.Assets.at("bower_components/angular-bootstrap-datetimepicker/src/js/datetimepicker.js")"></
    <script src="@routes.Assets.at("bower_components/Chart.js/Chart.min.js")"></script>
    <script src="@routes.Assets.at("bower_components/angular-chart.js/dist/angular-chart.min.js")"></script>

</head>

<body ng-app="@angularModuleName">
    @content
</body>

</html>
```

Fonte: elaborado pela autora (2016).

Como já citado anteriormente, todas as páginas HTML possuem a obrigatoriedade de compilarem primeiramente a página do Quadro 8 antes mesmo do seu próprio código. Dessa forma possuirão também todas as importações dos *frameworks* utilizados nesta classe. Pode-se visualizar no código da página `main.sacala.html` que os `links` e `scripts` são importados da pasta `public/bower_components`, esta pasta mantém todos os *frameworks*

externos utilizados no projeto Play `Viagens.Adm`. A partir disto verifica-se novamente os principais *frameworks front-end* utilizados no projeto, sendo eles:

a) AngularJS e jQuery, realizam controle e auxílio nas sintaxes *front-end* da aplicação;
b) Bootstrap, focado na facilidade e auxílio das interfaces da aplicação;
c) ChartJS e AngularChart, focados na construção do gráfico inicial da aplicação;
d) Swal e SweetAlert, focados nos avisos e mensagens de alerta ou erros da aplicação.

Após a chamada da página `main.scala.html` serão indicadas as páginas iniciais do sistema, estas dependem para qual endereço os métodos da classe `Aplication.java` enviarão a solicitação. O primeiro método enviará uma solicitação para a página do menu inicial do sistema e o segundo método enviará a solicitação para o menu de *login* do sistema. Estes dois exemplos de páginas HTML, o `menuPrincipal.scala.html` e o `login.scala.html` trabalham da mesma forma e podem ser compreendidos exemplificando apenas o primeiro exemplo citado. No Quadro 9 é possível analisar o código inicial da página `menuPrincial.scala.html`.

Quadro 9 - Exemplificação da página `menuPrincipal.scala.html`

menuPrincipal.scala.html

```
@(nomeEmpresa : String)

@main("viagensAdm"){
    <meta http-equiv="Content-Type" content="text/html;charset=UTF-8">
    <nav class="navbar navbar-inverse navbar-fixed-top">
      <div class="container-fluid">
        <div class="navbar-header">
          <button type="button" class="navbar-toggle collapsed" data-toggle="collapse" data-target="#navbar-collapse">
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
            <span class="icon-bar"></span>
          </button>
          <a class="navbar-brand" href="#/">
          <i class="fa fa-road" aria-hidden="true"></i>
              Viagens.Adm
           </a>
        </div>
        <div class="collapse navbar-collapse" id="navbar-collapse">
            <ul class="nav navbar-nav">
               <li>
                   <a href="#/listagem/viagens">
                       Viagens
```

Fonte: elaborado pela autora (2016).

A partir da página `menuPrincipal.scala.html` os restantes das páginas do sistema são chamados para a visualização e manutenção completa por parte do usuário. Esta mesma lógica ocorre com o `login.scala.html` do sistema. Este conceito vem do *framework* AngularJS e é conhecido como *single-page-application* ou aplicações com páginas únicas ou simples. A seguir é possível compreender melhor o conceito e o funcionamento do *front-end* do projeto baseado no AngularJS.

3.4.1.2 AngularJS

O AngularJS é um *framework JavaScript* com código aberto mantido pela empresa Google e, como já citado anteriormente, auxilia no conceito *single-page-application*, também conhecido por aplicativos com páginas únicas ou simples. Um dos seus objetivos é aumentar a utilização de aplicativos que podem ser acessados por um navegador web. Neste projeto o AngularJS é utilizado com o padrão Model View (MV) e o funcionamento deste inicia-se pela compilação do código HTML com as *tags* especiais e então executa-se a diretiva na qual essa *tag* pertence. A partir daí é feita a conexão entre a apresentação e o seu `model`, representado por variáveis *JavaScript* comuns, os valores destas variáveis podem ser mantidos manualmente ou via um recurso JSON.

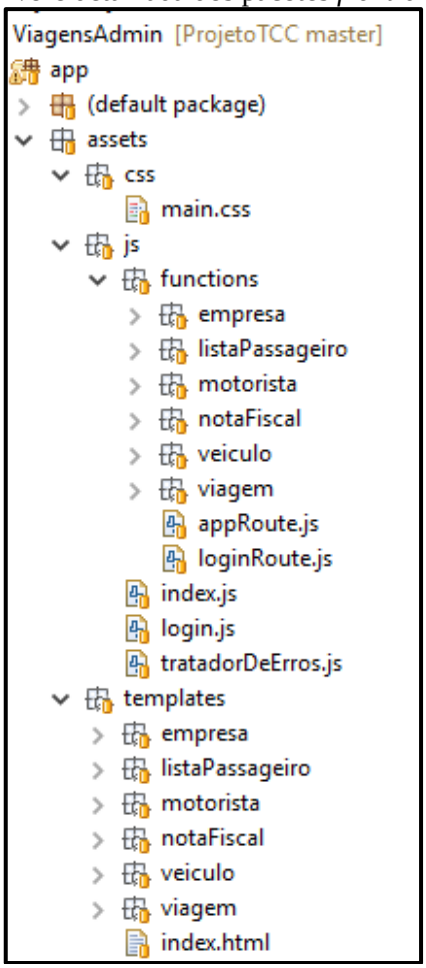
Neste trabalho o recurso JSON é utilizado para manter a comunicação entre os dados do *back-end* com o *front-end* da aplicação, e vice-versa. Mais exemplos de códigos do *back-end* referentes às funcionalidades do JSON serão analisados no subseção 3.4.2. Como citado anteriormente, o AngularJS possui *tags* especiais que facilitam na comunicação de uma aplicação no padrão MVC. Algumas destas *tags* especiais utilizadas neste desenvolvimento são:

a) `ng-app`: declara um elemento como o elemento raiz do aplicativo, ocasionando a mudança do comportamento padrão da *tag*;
b) `ng-model`: permite ligação direta entre a `view` e o escopo do aplicativo;
c) `ng-click`: permite instanciar o evento de click, semelhante ao *onclick*;
d) `ng-controller`: especifica um `controller` *JavaScript* para determinada página HTML;
e) `ng-repeat`: instancia um elemento por item de uma lista;
f) `ng-show`/`ng-hide`: mostra ou esconde um elemento HTML de acordo com o resultado de uma expressão *booleana*;
g) `ng-switch`: instancia um *template*, em uma lista de escolhas, dependendo do valor obtido pela expressão;
h) `ng-view`: a diretiva base para manipulação de rotas resolvendo um JSON antes de renderizar os modelos acionados por controladores especificados;
i) `ng-if`: declaração básica de um `if` que permite mostrar um elemento se a condição for verdadeira.

Antes da exemplificação das *tags* acima citadas e do funcionamento do AngularJS na prática, recomenda-se estar ciente do escopo e funcionamento deste trabalho, dando

continuidade na lógica iniciada no subitem 3.4.1.1 - Play Framework. Desta forma, a Figura 19 demonstra a árvore detalhada do pacote `assets` do sistema.

Figura 19 - Árvore detalhada dos pacotes *front-end* do sistema



Fonte: elaborado pela autora (2016).

O pacote `assets` mantém todos os códigos *front-end* do projeto `Viagens.Adm`. Dentro deste, existem divisões do próprio *front-end* da aplicação, sendo o pacote `css` com os códigos em linguagem Cascade Style Sheets (CSS), os pacotes `js` e `functions` com os códigos em *JavaScript* e o pacote `templates` que mantém os códigos em HTML. Dando continuidade à lógica iniciada no subcapítulo anterior, para que estes códigos sejam manipulados é necessário que se mantenha a página do `menuPrincipal.scala.html` e apenas altere-se as páginas dentro deste arquivo. Este conceito é aplicado devido ao *framework* AngularJS e por isso é citado dentro deste subcapítulo. O Quadro 10 demonstra a exemplificação da continuação do código da interface `menuPrincipal.scala.html`.

Quadro 10 - Continuação da exemplificação do código `menuPrincipal.scala.html`

menuPrincipal.scala.html

```
73            </div>
74          </div>
75        </nav>
76      <div class="container-fluid">
77          <div class="row">
78              <div class="col-md-12">
79                  <div ng-view></div>
80              </div>
81          </div>
82        </div>
83        <script src="@routes.Assets.at("dist/js/index.js")"></script>
84  }
```

Fonte: elaborado pela autora (2016).

No Quadro 10 pode-se verificar o comando dentro da *tag* `<script>` (linha 83) do código, este informa que o arquivo `routes` deve redirecionar para o arquivo *JavaScript* no caminho `dist/js/index.js`. No Quadro 11 é possível visualizar o arquivo do *front-end* `index.js`.

Quadro 11 - Exemplificação do código do arquivo `index.js`

index.js | appRoute.js

```
1  var viagensAdm = angular.module("viagensAdm", ['ngRoute', 'ui.bootstrap.datetimepicker', 'chart.js']);
2  viagensAdm.config(require('./functions/appRoute'));
3
4  viagensAdm.controller("MinhaContaFunction", require('./functions/empresa/minha-conta-function'));
5  viagensAdm.controller("AlterarSenhaFunction", require('./functions/empresa/alterar-senha-function'));
6  viagensAdm.controller("ViagemFunction", require('./functions/viagem/viagem-function'));
7  viagensAdm.controller("VeiculoFunction", require('./functions/veiculo/veiculo-function'));
8  viagensAdm.controller("MotoristaFunction", require('./functions/motorista/motorista-function'));
9  viagensAdm.controller("ListagemViagensFunction", require('./functions/viagem/listagem-viagens-function'));
10 viagensAdm.controller("ListagemVeiculosFunction", require('./functions/veiculo/listagem-veiculos-function'));
11 viagensAdm.controller("ListagemMotoristasFunction", require('./functions/motorista/listagem-motoristas-function'))
12 viagensAdm.controller("GraficoStatusViagensFunction", require('./functions/viagem/grafico-status-viagens-function
13 viagensAdm.controller("ListagemViagensInicialFunction", require('./functions/viagem/listagem-viagens-inicial-funct
14 viagensAdm.controller("NotaFiscalFunction", require('./functions/notaFiscal/notaFiscal-function'));
15 viagensAdm.controller("ListaPassageiroFunction", require('./functions/listaPassageiro/listaPassageiro-function'))
16
17 viagensAdm.factory("MotoristaFunctionHelp", require("./functions/motorista/motorista-function-help"));
18 viagensAdm.factory("VeiculoFunctionHelp", require("./functions/veiculo/veiculo-function-help"));
19 viagensAdm.factory("ViagemFunctionHelp", require("./functions/viagem/viagem-function-help"));
20 viagensAdm.factory("NotaFiscalFuntionHelp", require("./functions/notaFiscal/notaFiscal-function-help"));
21 viagensAdm.factory("ListaPassageiroFuntionHelp", require("./functions/listaPassageiro/listaPassageiro-function-he
22 viagensAdm.factory("LabelGraficoGenerator", require('./functions/viagem/label-grafico-generator'));
```

Fonte: elaborado pela autora (2016).

Este é um arquivo importante para a compreensão do funcionamento do *front-end* do sistema. Ele mapeia os arquivos `controllers` e `factorys` do *front-end* que são os próprios `functions` dos *templates* em HTML. Para obter-se um controle completo da manutenção do *front-end*, o arquivo `index.js` encaminha essa funcionalidade para o arquivo `appRoute`. A função deste arquivo é muito parecida com o funcionamento do arquivo `routes` do *back-end* do sistema. A principal diferença é que neste caso o `appRoute` mantém as interfaces e funções dos arquivos *front-end*, já o `routes` mantém as funcionalidades necessárias para realizar de fato as funções que o sistema permite, sendo estas: cadastro, buscas, etc. No Quadro 12 pode-se verificar a exemplificação do código do arquivo `appRoute.js`.

Quadro 12 - Exemplificação do código do arquivo `appRoute.js`

index.js | appRoute.js

```
1  var AppRouteProvider = function ($routeProvider) {
2      'use strict';
3
4      $routeProvider.
5      when('/', {
6          templateUrl: '/assets/dist/templates/index.html'
7      }).
8      when('/minha-conta', {
9          templateUrl: '/assets/dist/templates/empresa/minha-conta.html',
10         controller: 'MinhaContaFunction'
11     }).
12     when('/minha-conta/alterar-senha', {
13         templateUrl: '/assets/dist/templates/empresa/alterar-senha.html',
14         controller: 'AlterarSenhaFunction'
15     }).
16     when('/motorista/:codMotorista', {
17         templateUrl: '/assets/dist/templates/motorista/motorista.html',
18         controller: 'MotoristaFunction'
19     }).
20     when('/viagem/:codViagem', {
21         templateUrl: '/assets/dist/templates/viagem/viagem.html',
22         controller: 'ViagemFunction'
23     })
```

Fonte: elaborado pela autora (2016).

No Quadro 12, visualiza-se as funcionalidades do arquivo `appRoute`. Este comanda e administra todas as rotas que a aplicação *front-end* possui, o seu funcionamento é simples, primeiramente é realizado um método `when()`(linha 5) para cada possibilidade de rota de páginas, após isto é informado o endereço da rota e então, dependendo deste endereço, é redirecionado um determinado `template` e um `controller` para o browser. Estes códigos podem ser visualizados no quadro anterior e são descritos pelos comandos `templateUrl: 'caminho\do\html'` e `controller: 'NomeDoFunctionComandaHtml'`. Desta forma as páginas do *front-end* da aplicação são organizadas. A seguir será exemplificado de que maneira a comunicação do *front-end* com o *back-end* se relaciona.

Dando continuidade na explicação do *framework* AngularJS, as *tags* especiais acima citadas podem ser visualizadas no Quadro 13 que exemplifica o código da página `nova-empresa.html`.

Quadro 13 - Exemplificação do código da página `nova-empresa.html`

nova-empresa.html

```
3    <h1>Preencha o formulário de cadastro da sua empresa</h1>
4  </div>
5  <form class="form-horizontal">
6      <div class="form-group">
7        <div class="col-sm-12">
8          <input type="text" class="form-control" ng-model="empresa.nome" placeholder="Digite o nome da sua empre
9        </div>
10     </div>
11     <div class="form-group">
12       <div class="col-sm-12">
13         <input type="cnpj" class="form-control" ng-model="empresa.cnpj" placeholder="Digite o CNPJ da sua empre
14       </div>
15     </div>
16     <div class="form-group">
17       <div class="col-sm-12">
18         <input type="email" class="form-control" ng-model="empresa.email" placeholder="Digite o e-mail da sua e
19       </div>
20     </div>
21     <div class="form-group">
22       <div class="col-sm-12">
23         <input type="password" class="form-control" ng-model="empresa.senha" placeholder="Crie e digite uma sen
24       </div>
25     </div>
26     <div class="form-group">
27       <div class="col-sm-12">
28         <input type="password" class="form-control" ng-model="empresa.senha" placeholder="Digite a senha novame
29       </div>
30     </div>
31     <div class="form-group">
32       <div class="col-sm-12">
33         <label>Selecionar logo da empresa</label><br/>
34         <input type="file" class="form-control" ng-model="empresa.logo">
35       </div>
36     </div>
37     <div class="form-group">
38       <div class="col-sm-12">
39         <button type="submit" class="btn btn-primary btn-lg btn-block" ng-click="cadastrar(empresa)">Cadastrar<
40       </div>
```

Fonte: elaborado pela autora (2016).

No código da página `nova-empresa.html` pode-se obter maior compreensão do funcionamento do *framework* AngularJS. Neste verifica-se dentro do `<input>` o comando `ng-model="empresa.variavel"` (exemplo linha 8), este comando informa ao *framework* que o `model Empresa.java`, juntamente com as suas determinadas variáveis, está em utilização nesta página. Logo abaixo no código pode-se visualizar a *tag* `ng-click="cadastrar(empresa)"` indicada no comando `<button>` da página. Essa *tag* envia ao determinado *JavaScript* `nova-empresa-function.js`, no qual comanda a página HTML, o `model` da `Empresa.java` com os determinados dados inseridos.

Neste ponto, por meio da manutenção dos arquivos *JavaScripts* `index.js` e `appRoutes.js`, que serão analisados a seguir é chamada a página `nova-empresa-function.js` que controla o HTML visualizado no Quadro 13. No Quadro 14 é possível analisar a exemplificação do código da página `nova-empresa-function.js`.

Quadro 14 - Exemplificação do código da página `nova-empresa-function.js`

nova-empresa-function.js

```
var CadastrarNovaEmpresaFunction = function ($scope,  $http, $location){
    'use strict';

    var tratadorDeErros = require("../../tratadorDeErros");

    $scope.cadastrar = function (empresa){
            $http.post("/empresa/salvar", empresa)
                .success(function (inputSalvo){
                    swal({
                            title: "Empresa cadastrada com sucesso!",
                            text : inputSalvo.nome,
                            type : "success",
                            allowOutsideClick : true
                    },function (){
                        $location.path("/");
                        $scope.$apply();
                    });
                })
                .error(tratadorDeErros(function (){
                         $scope.empresa = {};
                         $scope.$apply();
                 }));
    };
};

module.exports = ['$scope', '$http', '$location', CadastrarNovaEmpresaFunction];
```

Fonte: elaborado pela autora (2016).

Após a seleção do botão cadastrar da página HTML, a função `$scope.cadastrar` é chamada na página *JavaScript* exemplificada no Quadro 14. Essa função encaminha uma URI juntamente com o método `HTTP` do tipo `post` para o arquivo `routes` do Play, visto no Quadro 6 e encaminha também o modelo `Empresa.java`, visto no Quadro 5, juntamente com os dados inseridos na página HTML. A partir disto, são chamadas as classes que controlam e realizam o serviço de cadastro no banco de dados. Essas classes podem ser visualizadas nos Quadros 15, 16 e 17.

Quadro 15 - Exemplificação do código da página `EmpresaController.java`

EmpresaController.java

```
1  package controllers.empresa;
2
3  import java.sql.SQLException;
25
26 @Service
27 public class EmpresaController extends BaseController<Empresa> {
28     @Autowired
29     private SessionHelper sessionHelper;
30     @Autowired
31     protected JsonParser jsonParser;
32     @Autowired
33     private EmpresaService empresaService;
34     @Autowired
35     private TratadorExcecao tratadorDeExcecaoSQL;
36     @Autowired
37     private AlterarSenhaValidator alterarSenhaValidator;
38
39     @Override
40     protected BaseService<Empresa> getService() {
41         return empresaService;
42     }
43     @Override
44     protected Class<Empresa> getModelClass() {
45         return Empresa.class;
46     }
47     @Override
48     public Result salvar() {
49         // TODO Auto-generated method stub
50         return super.salvar();
51     }
```

Fonte: elaborado pela autora (2016).

No Quadro 15 é possível verificar o método `salvar()` (linha 48) da classe `EmpresaController.java` chamado pelo arquivo `routes` do Play a partir da URI enviada anteriormente. Este método é estendido de uma classe `controller` base e padrão para todas as outras classes `controllers` do sistema. No Quadro 16 pode-se visualizar a classe e o método `BaseController.salvar()`.

Quadro 16 - Exemplificação do código da classe `BaseController.java`

EmpresaController.java | BaseController.java

```
1  package controllers.base;
2
3  import java.util.List;
16 public abstract class BaseController<T extends Model> extends Controller {
17     @Autowired
18     private SessionHelper sessionHelper;
19     @Autowired
20     protected JsonParser jsonParser;
21     @Autenticado
22     @TratarExcecao
23     public Result buscar() {
24         int codigoEmpresa = sessionHelper.buscarDaSessao().getCodigo();
25         List<T> resultados = getService().buscar(codigoEmpresa);
26         return ok(Json.toJson(resultados));
27     }
28     @Autenticado
29     @TratarExcecao
30     public Result buscarPeloId(Integer id) {
31         T resultado = getService().buscarPeloId(id);
32         return ok(Json.toJson(resultado));
33     }
34     @Autenticado
35     @TratarExcecao
36     public Result salvar() {
37         T input = jsonParser.bindFromRequest(getModelClass());
38         T inputSalvo = getService().salvar(input);
39         return ok(Json.toJson(inputSalvo));
40     }
41     @Autenticado
42     @TratarExcecao
43     public Result remover(Integer id) {
44         getService().remover(id);
45         return buscar();
46     }
47     protected abstract BaseService<T> getService();
48     protected abstract Class<T> getModelClass();
49 }
```

Fonte: elaborado pela autora (2016).

A classe `BaseController.java` é estendida para todos os `controllers` do sistema. Pode-se verificar no Quadro 16 o método `salvar()` (linha 36) da classe `BaseController.java`, nesta primeira ação é realizada o encontro direto do componente JSON com o `model` repassado do *front-end* para o *back-end* com o comando `T input = jsonParser.bindFromRequest(getModelClass())` (linha 37). Desta forma, por meio da ferramenta JSON o *back-end* realiza a manipulação do `model` padrão recebido da biblioteca do AngularJS, enviado anteriormente pelo método `HTTP POST`. Seguindo a lógica do exemplo da empresa, este componente JSON será o `model Empresa.java` preenchido com as devidas informações vindas do *front-end*. Posteriormente é encaminhada uma solicitação para o método `salvar()` da classe `EmpresaService.java`, este realizará de fato o cadastro no banco de dados. No Quadro 17 pode-se visualizar a exemplificação de parte do código da classe `EmpresaService.java`.

Quadro 17 - Exemplificação do código da classe `EmpresaService.java`

*EmpresaService.java

```
1  /**..
4  package service.empresa;
5
6  import java.sql.Connection;..
25
26 @Service
27 public class EmpresaService extends BaseService<Empresa> {
28
29     public static final long CODIGO_EMPRESA_NAO_ENCONTRADO = -1L;
30
31     @Autowired
32     private LoginValidator loginValidator;
33
34     @Override
35     public Empresa buscarPeloId(Integer codigo) {
36         return Empresa.findUnique(codigo);
37     }
38
39     public Empresa buscarPeloEmail(String email) {
40         return Empresa.findUnique(email);
41     }
42     @Override
43     @Transactional(rollbackFor = Throwable.class)
44     public Empresa salvar(Empresa empresa) {
45         validarCampos(empresa);
46         empresa.save();
47         return buscarPeloId(empresa.getCodigo());
48     }
```

Fonte: elaborado pela autora (2016).

O Quadro 17 demonstra o exemplo dos códigos `services`, com base na classe `EmpresaService.java`, dos métodos que fazem o serviço direto de conexão com o banco de dados relacional do sistema, neste caso o PostgreSQL. Desta maneira, é utilizado novamente nas classes `services` as funcionalidades do Hibernate e também do Spring *framework*. No código do método `salvar()`(linha 44) no comando `empresa.save()`(linha 46) é feito a conexão com a biblioteca do Hibernate e é realizado o cadastro do `model Empresa` no banco de dados, agilizando o processo de inserção.

Com isso, a partir das classes `services` os dados são inseridos, excluídos, alterados e consultados no banco de dados. Para realizar uma manutenção e dependência mais ágil destas classes é utilizado o Spring *framework*. Nos Quadros 18 e 19 pode-se verificar a maneira em que o Spring está inserido neste trabalho.

Quadro 18 - Exemplificação do funcionamento das classes do Spring *framework*

AppConfig.java | Global.java

```
1
2  import org.springframework.context.ApplicationContext;
3  import org.springframework.context.annotation.AnnotationConfigApplicationContext;
4
5  import play.Application;
6  import play.GlobalSettings;
7  import config.AppConfig;
8
9  public class Global extends GlobalSettings {
10
11     private ApplicationContext ctx;
12
13     @Override
14     public void onStart(Application app) {
15         ctx = new AnnotationConfigApplicationContext(AppConfig.class);
16     }
17
18     @Override
19     public <A> A getControllerInstance(Class<A> clazz) {
20         return ctx.getBean(clazz);
21     }
```

Fonte: elaborado pela autora (2016).

No Quadro anterior pode-se analisar as bibliotecas e métodos importados para o funcionamento completo do Spring *framework* e suas dependências. No Quadro 19 considera-se a classe `AppConfig.java` com a lista de componentes utilizados e mantidos pelo Spring.

Quadro 19 - Exemplificação da classe `AppConfig.java` do Spring *framework*

AppConfig.java | Global.java

```
1  package config;
2
3  import org.springframework.context.annotation.ComponentScan;
4  import org.springframework.context.annotation.Configuration;
5
6  @Configuration
7  @ComponentScan(basePackages = { "controllers", "service", "validators", "exception"})
8  public class AppConfig {
9
10 }
```

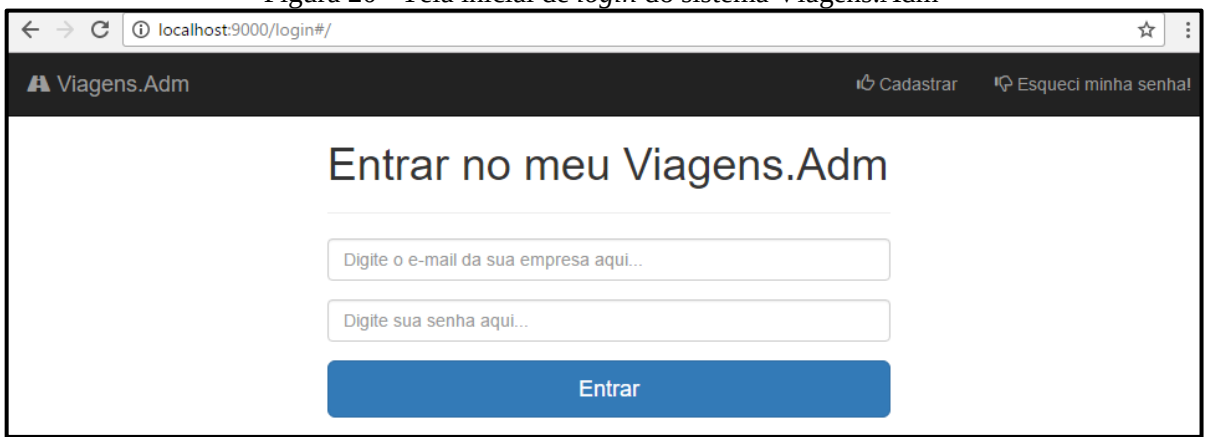
Fonte: elaborado pela autora (2016).

Até este ponto analisou-se o completo funcionamento da aplicação, compreendendo a comunicação do *back-end* com o *front-end* por meio das técnicas utilizadas. Na subseção 3.4.2 visualizam-se as interfaces do sistema, como também a exemplificação de alguns códigos essenciais para a compreensão deste.

3.4.2 Operacionalidade da implementação

Nesta subseção será detalhado a operacionalidade da implementação do sistema, juntamente com as figuras das interfaces e códigos explicativos das principais funcionalidades. Na Figura 20 é possível verificar a tela inicial de login do sistema Viagens.Adm.

Figura 20 - Tela inicial de *login* do sistema Viagens.Adm



Fonte: elaborado pela autora (2016).

A tela inicial de *login* aparecerá para os usuários, donos de empresas de turismo, que já possuem ou não a sua empresa cadastrada no sistema. Desta forma, caso o usuário não possua o cadastro de sua empresa no sistema, para realizar o *login* é preciso acessar a tela de cadastro da empresa e proceder com as devidas informações necessárias. Na Figura 21 observa-se a tela de cadastro inicial da empresa. Caso o usuário já possua a sua empresa cadastrada no sistema ele poderá inserir o devido e-mail e a senha cadastrada anteriormente.

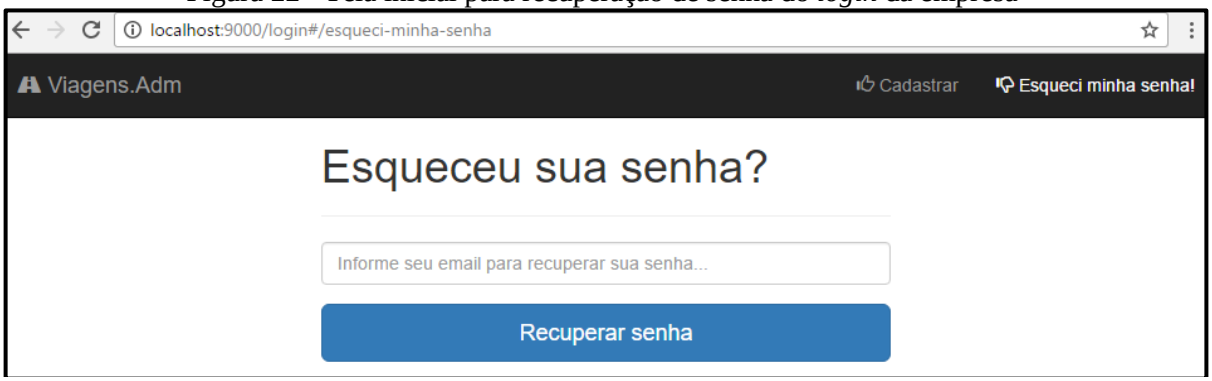
Figura 21 - Tela de cadastro inicial da empresa

Fonte: elaborado pela autora (2016).

Para realizar definitivamente o cadastro é necessário informar o `nome` da empresa, o `CNPJ`, o `e-mail` válido da empresa e uma `senha` com no mínimo seis dígitos. Após inserir todos os campos válidos a empresa será cadastrada no banco de dados sistema. A partir disto é

possível realizar o primeiro *login* no sistema. Caso o usuário esqueça a senha informada no cadastro da empresa, este poderá acessar a tela de recuperação de senha e informar o e-mail cadastrado. Sua nova senha será enviada ao e-mail informado. Na Figura 22 pode-se visualizar a tela inicial para a recuperação de senha do *login* da empresa.

Figura 22 - Tela inicial para recuperação de senha do *login* da empresa



Fonte: elaborado pela autora (2016).

Caso a recuperação de senha seja desnecessária para o usuário, para concluir com a efetivação do *login* da empresa é realizado no sistema uma série de validações. Essas validações podem ser verificados no Quadro 20 que exemplifica o método `logar()` da classe `EmpresaController.java`.

Quadro 20 - Exemplificação de método da classe `EmpresaController.java`

EmpresaController.java | EmpresaService.java

```
57    @TratarExcecao
58    public Result logar() {
59        session().clear();
60        try {
61            EmpresaLogin empresaLogin = jsonParser.bindFromRequest(EmpresaLogin.class);
62            return fazerLogin(empresaLogin);
63        } catch (ViagensAdminException e) {
64            return redirecionarParaLoginComErro(e.getMessage());
65        } catch (SQLException e) {
66            return redirecionarParaLoginComErro(e.getMessage());
67        }
68    }
69    private Result fazerLogin(EmpresaLogin empresaLogin) throws ViagensAdminException, SQLException {
70        if (empresaService.autenticar(empresaLogin)) {
71            Empresa empresa = empresaService.buscarPeloEmail(empresaLogin.getEmail());
72            sessionHelper.colocarNaSessao(empresa);
73            return redirect("/");
74        }
75        return redirecionarParaLoginComErro("Empresa Inválida...");
76    }
77    private Result redirecionarParaLoginComErro(String erro) {
78        flash("erro", erro);
79        return redirect("/login");
80    }
```

Fonte: elaborado pela autora (2016).

No método demonstrado pode-se verificar primeiramente que a sessão do sistema é limpada e após isto, caso não surjam erros, o e-mail e a senha anteriormente informados na página de *login* são redirecionados por meio de um `model EmpresaLogin.java` para o método chamado `fazerLogin()`(linha 69). Este método possui um `if()` que autentica o

`model EmpresaLogin` e após autenticar insere a empresa na sessão e redireciona o usuário para a tela inicial do sistema. Esta autenticação é realizada pelo método `autenticar()` (linha 70) da classe `EmpresaService.java`. No Quadro 21 é possível visualizar a exemplificação do código deste método.

Quadro 21 - Exemplificação de método classe `EmpresaService.java`

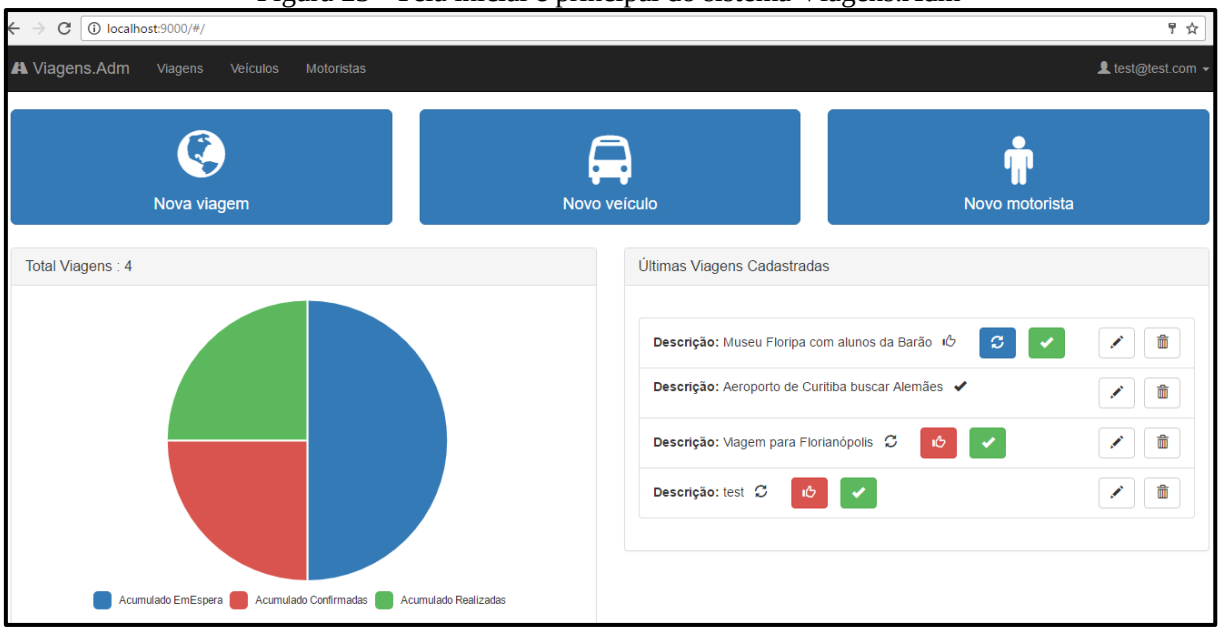
aController.java | EmpresaService.java

```
public boolean autenticar(EmpresaLogin empresaLogin) throws ViagensAdminException, SQLException {
    loginValidator.validarLogin(empresaLogin);
    Connection connection = DB.getConnection();
    String sql = "select * from empresa where email_empresa = '" + empresaLogin.getEmail() + "' and pw_empresa = '"
    Statement statement = connection.createStatement();
    ResultSet resultSet = statement.executeQuery(sql);
    boolean estaAutenticado = resultSet.next();
    connection.close();
    return estaAutenticado;
}
```

Fonte: elaborado pela autora (2016).

O método acima visualizado verifica primeiramente se os campos da página *login* não estão vazios. Após isto é realizada uma conexão com o bando de dados e é verificado, por meio de um `select`, se os dados informados na tela estão de fato cadastrados no sistema. Caso o retorno de todos esses métodos sejam positivos o usuário será redirecionado a página inicial do sistema, esta pode ser visualizada na Figura 23.

Figura 23 - Tela inicial e principal do sistema Viagens.Adm



Fonte: elaborado pela autora (2016).

Na Figura 23 pode-se visualizar a tela inicial do sistema proposto, este possui um menu com opções de Viagens, Veículos e Motoristas. Estas três opções possuem o propósito de visualizar, buscar, alterar ou excluir as determinadas informações dos dados já cadastrados no sistema e será demonstrado a seguir. Além disso, ainda no menu é possível clicar no nome da empresa e alterar a senha ou os dados da mesma. Na página principal logo a baixo pode-se

visualizar os três botões principais do sistema: `Nova viagem`, `Novo veículo` e `Novo motorista`. A partir destes é possível cadastrar os dados no sistema. Logo abaixo pode-se verificar também um gráfico e uma tabela inicial das viagens cadastradas do sistema. Estes dois componentes iniciais acabam facilitando o acesso às viagens necessárias para realizar os serviços de determinadas empresas.

Para realizar o gráfico dos status das viagens foi utilizado uma biblioteca própria do *framework* AngularJS, chamada Angular Chart ou ChartJS. Estes facilitaram na interface como também em toda a lógica do gráfico. No Quadro 22 pode-se visualizar o código da página *JavaScript* do gráfico acima demonstrado.

Quadro 22 - Exemplificação da página `grafico-status-viagens-function.js`

*grafico-status-viagens-function.js | ViagemService.java | label-grafico-generator.js

```
var GraficoStatusViagensFunction = function ($scope, $filter, ViagemFunctionHelp, LabelGraf
    'use strict';

    var itensNavegados = [];
    atualizarGraficoDinamicamente("viagens");

    function atualizarGraficoDinamicamente(item) {
        if (item === "viagens") {
            ViagemFunctionHelp.buscarViagensResumo().success(function (viagensResumo) {
                $scope.valorAcumulado = "Total Viagens : " + (viagensResumo.totalViagens);
                $scope.labels = [LabelGraficoGenerator.acumulado("EmEspera"),
                                 LabelGraficoGenerator.acumulado("Confirmadas"),
                                 LabelGraficoGenerator.acumulado("Realizadas")];
                $scope.data = [viagensResumo.acumuladoStatusEmEspera,
                               viagensResumo.acumuladoStatusConfirmadas,
                               viagensResumo.acumuladoStatusRealizadas];
                $scope.colours = ['#337ab7', '#d9534f', '#5cb85c'];
            });
        }
    }

    $scope.atualizarGrafico = function (points, event) {
        if (points.length > 0) {
            var item = points[0].label;
            atualizarGraficoDinamicamente(item);
        }
    };
```

Fonte: elaborado pela autora (2016).

O funcionamento do gráfico é simples, primeiramente é feita uma busca da quantidade de viagens inseridas no sistema, estes acumulados são divididos pelos três tipos de status de viagens, sendo estes em espera, confirmadas e realizadas. Após isto e inserido em um vetor e carregado no gráfico por meio do código `JavaScript` anteriormente verificado.

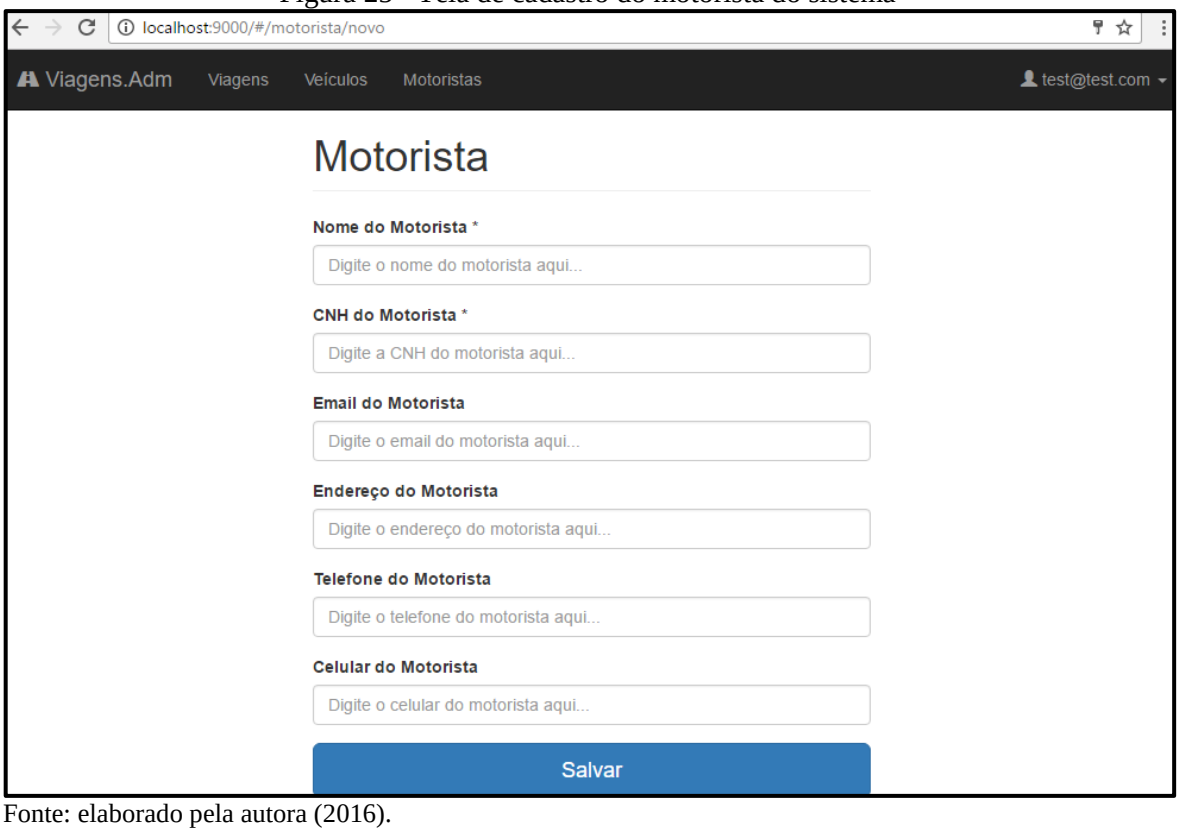
Ainda na página inicial, o usuário pode cadastrar no sistema todos os veículos que possui antes mesmo de cadastrar uma viagem. Na Figura 24 pode-se visualizar a tela de cadastro do veículo.

Figura 24 - Tela de cadastrado do veículo do sistema

Fonte: elaborado pela autora (2016).

Na Figura 24 é necessário inserir todas as informações do veículo para que este seja cadastrado. As informações essenciais são a `placa`, o `modelo`, a `marca`, o `ano` e a `lotação` do veículo. Antecipadamente ao cadastro da viagem pode-se também cadastrar no sistema os motoristas que a empresa possui. Na Figura 25 pode-se visualizar a tela de cadastro destes.

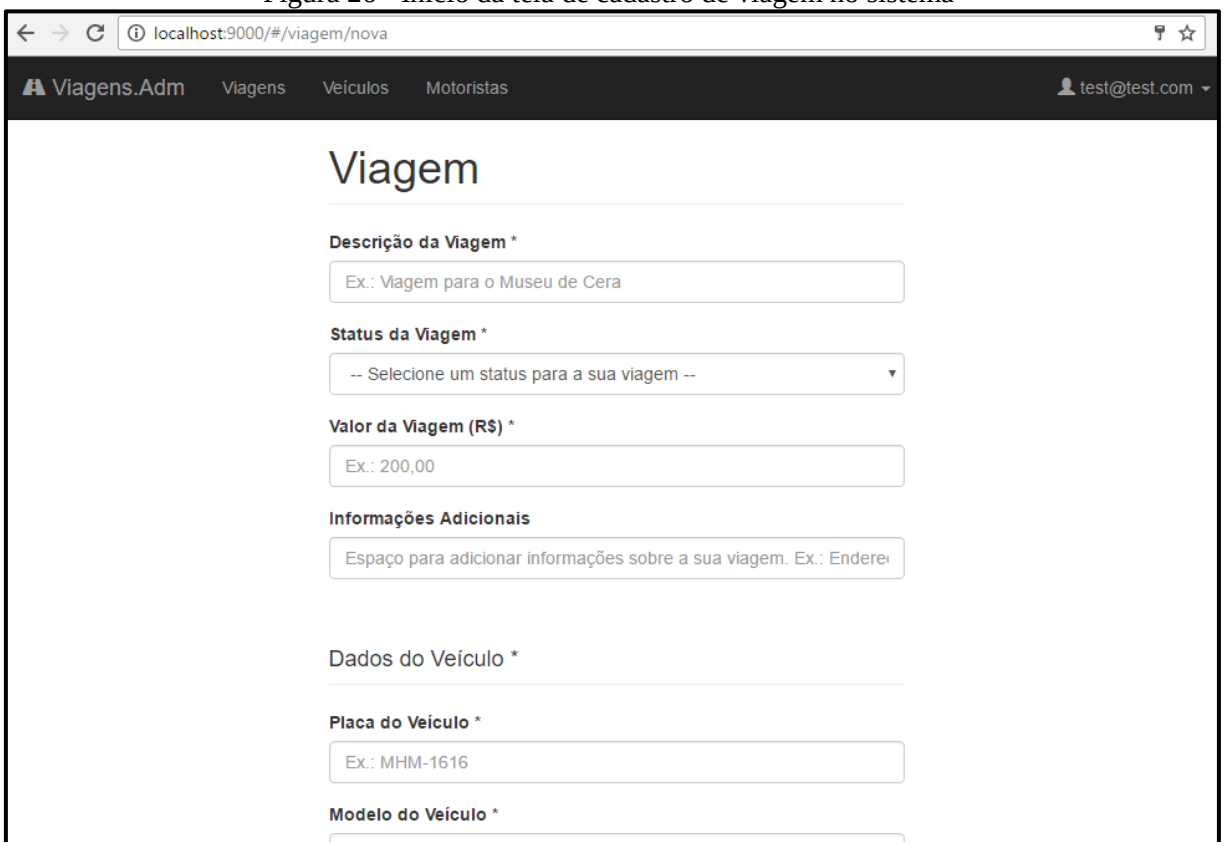
Figura 25 - Tela de cadastro do motorista do sistema



Fonte: elaborado pela autora (2016).

No cadastro de um motorista é necessário informar obrigatoriamente o `nome` e o número da `CNH`. Após inserir os dados no sistema pode-se prosseguir ao cadastro de uma viagem. Na Figura 26 pode-se visualizar o início da tela de cadastro de viagem no sistema.

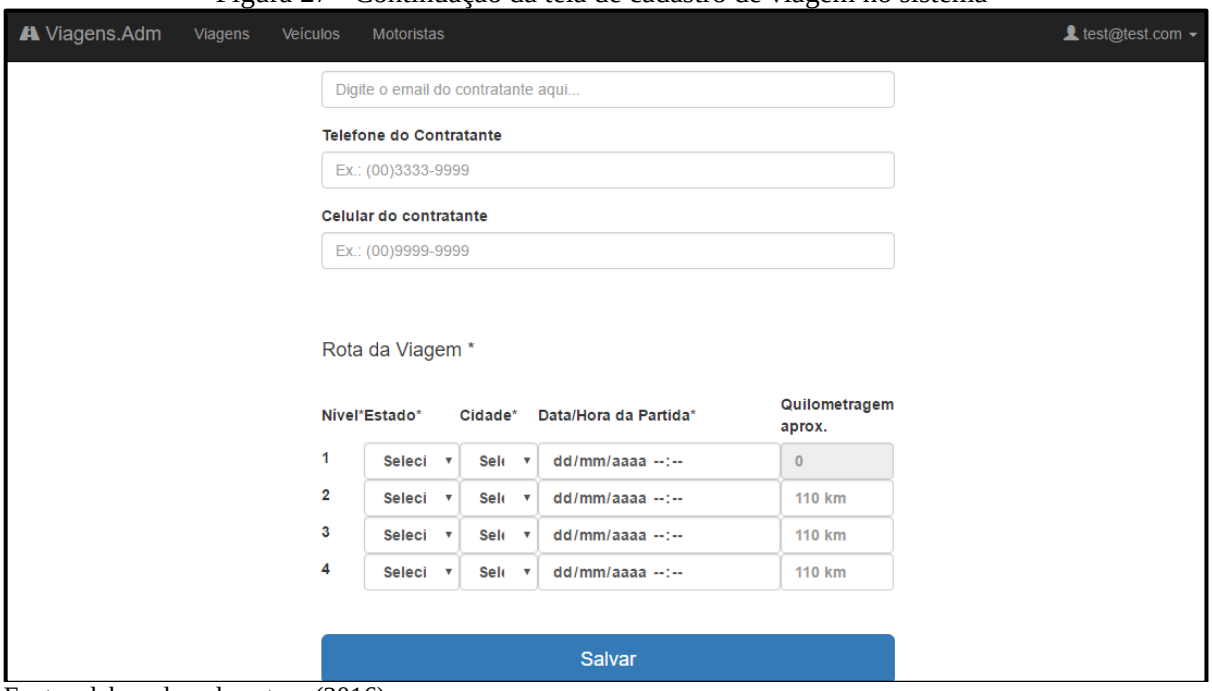
Figura 26 - Início da tela de cadastro de viagem no sistema



Fonte: elaborado pela autora (2016).

O cadastro da viagem é uma das etapas mais importantes do sistema, pois nela é possível cadastrar a própria viagem, com seus devidos campos, como também é possível selecionar um motorista e veículo já cadastrados ou cadastrar novos a partir desta página. Além disso, é necessário proceder com o cadastro do contratante da viagem como também das rotas. Na Figura 27 pode-se visualizar a continuação da tela de cadastro da viagem, onde mostra parte dos campos necessários para cadastrar o contratante e as rotas. Nas `rotas` é necessário selecionar um `estado` e uma `cidade`, após isso a `data` e a `hora de partida` daquela determinada `cidade` selecionada, e caso seja necessário a `quilometragem aproximada` do trecho da rota.

Figura 27 - Continuação da tela de cadastro de viagem no sistema



Fonte: elaborado pela autora (2016).

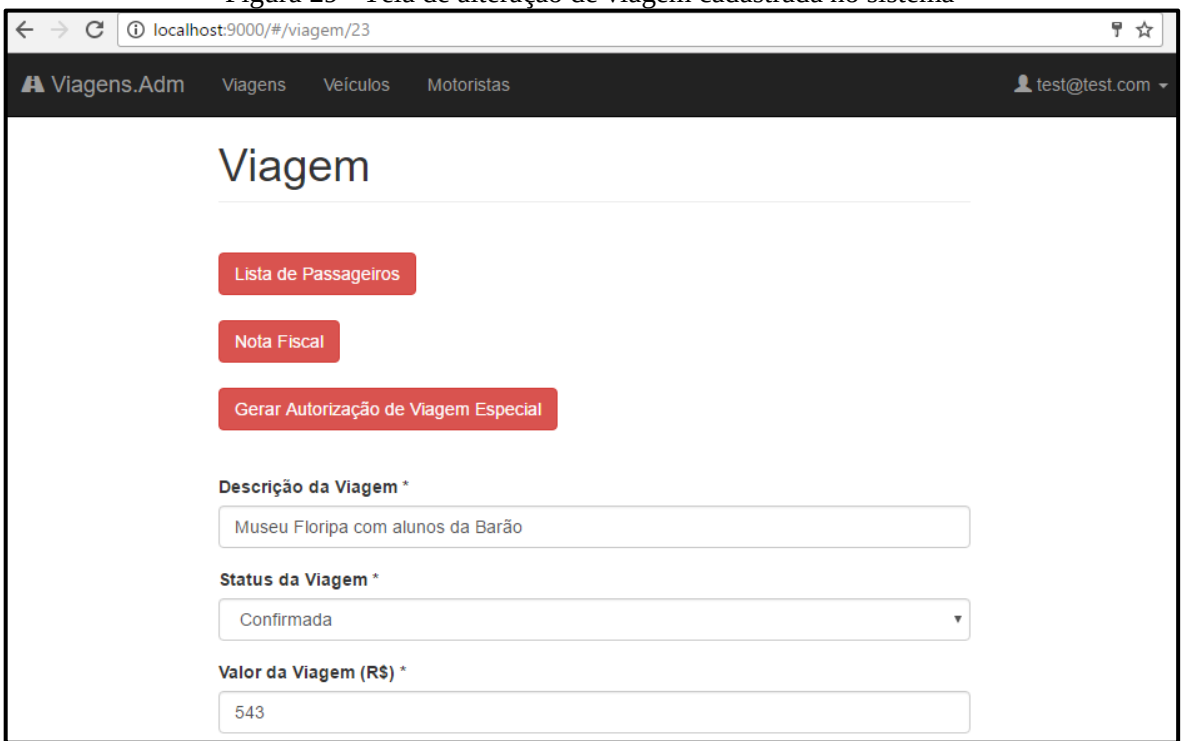
Após inserir todas as informações uma `viagem` poderá ser cadastrada no sistema, juntamente com seu respectivo `contratante`, `motorista`, `veículo` e `rotas` relacionados. A Figura 28 demonstra a tela para buscar, visualizar, alterar ou excluir determinada `viagem` cadastrada. Este mesmo padrão de tela se repete para as `viagens`, `veículos` e `motoristas`.

Figura 28 - Padrão de tela de consulta de viagens cadastradas

Fonte: elaborado pela autora (2016).

Na Figura 28 pode-se visualizar as viagens cadastradas no sistema. Essa tela também disponibiliza um campo `Procurar Viagens`, sendo possível pesquisar por todos os dados cadastrados nesta tela. É possível também remover uma determinada viagem como também editar a mesma clicando sobre a descrição. Na Figura 29 é possível visualizar a tela de alteração de uma viagem.

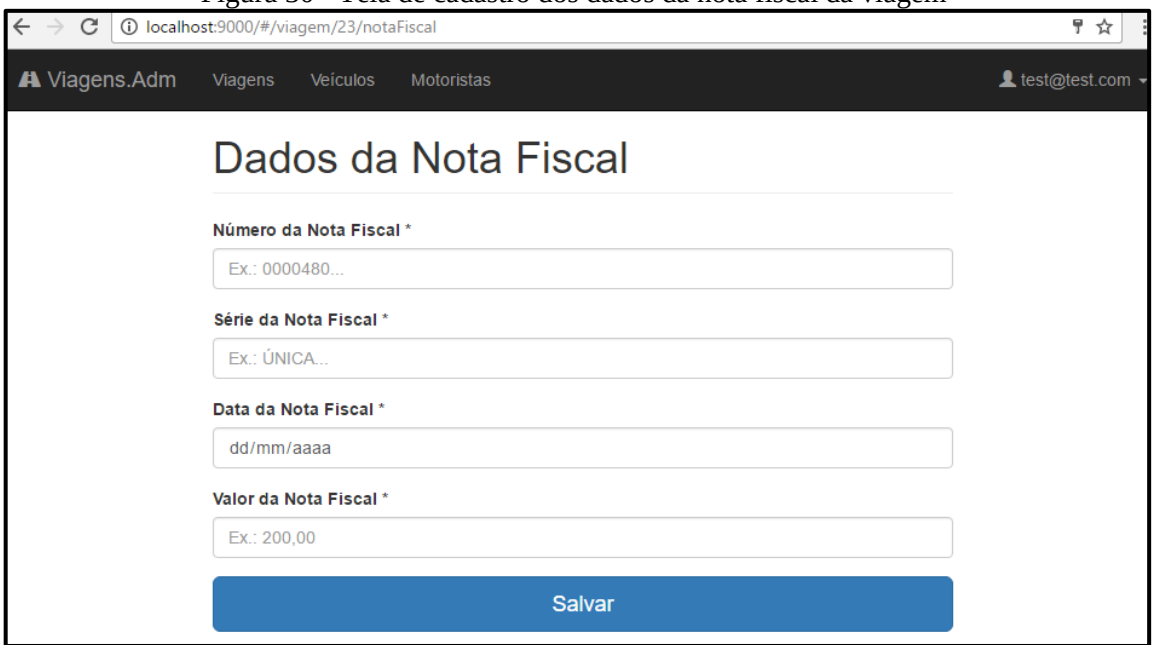
Figura 29 - Tela de alteração de viagem cadastrada no sistema



Fonte: elaborado pela autora (2016).

Nesta tela pode-se perceber certa diferença de interface, pois a partir do momento em que uma `viagem` está cadastrada no sistema é possível cadastrar a `Lista de Passageiros` e os dados da `Nota Fiscal`. Após o cadastro destes, pode-se então prosseguir com a futura emissão da `Autorização de Viagem Especial`, que será disponibilizada a partir do momento em que a API do Deter-SC for concluída. Na Figura 30 pode-se visualizar a tela de cadastro dos dados da `nota fiscal`.

Figura 30 - Tela de cadastro dos dados da nota fiscal da viagem



Fonte: elaborado pela autora (2016).

Para o cadastro dos dados da nota fiscal é necessário informar o número, a série a data e o valor da nota. Estes dados são essenciais para a futura integração com o sistema do Deter-SC, pois para realizar a emissão da autorização de viagem especial é necessário possuir todos os dados cadastrados no sistema, inclusive os dados financeiros da nota fiscal. No seção 3.5 serão tratadas e analisadas as questões quanto aos resultados e discussões finais da disponibilidade e funcionalidade do sistema proposto.

3.5 RESULTADOS E DISCUSSÕES

Nesta sessão são apresentados os resultados e as discussões do trabalho. Todos os objetivos e requisitos do trabalho foram concluídos com êxito e os resultados foram satisfatórios. Conforme esses objetivos previstos, o sistema web desenvolvido propôs aumentar a produtividade e organização, como também manter a gestão e administração das empresas do ramo turístico de transporte de passageiros. Este sistema permite às empresas cadastrarem seus veículos, motoristas, viagens, contratantes, listas de passageiros e dados das notas fiscais de uma maneira organizada que aumente a produtividade no dia a dia dos processos das agências de viagens.

A análise e avaliação da qualidade de software do sistema foi realizada pelas empresas do ramo. Desta forma, foi feita uma reunião presencial com quatro empresas de transporte e turismo de Blumenau-SC. Nestas reuniões o sistema foi demonstrado para os donos e funcionários, foram realizadas perguntas voltadas à qualidade de software e qualidade de uso. Os principais pontos levados em consideração foram com base nas características da qualidade de software e de uso, neste contexto foram determinadas as seguintes:

a) eficiência: capacidade do produto de software de permitir ao usuário atingir metas específicas como completude, em um contexto de uso específico;
b) produtividade: capacidade do produto de software de permitir que seus usuários empreguem quantidade adequada de recursos em relação à efetividade alcançada em um contexto de uso específico;
c) segurança: capacidade do produto de software de apresentar níveis aceitáveis de riscos de danos a pessoas, negócios, software, propriedade ou ambiente em um contexto de uso específico;
d) satisfação: capacidade do produto de software de satisfazer usuários em um contexto de uso específico;
e) usabilidade: capacidade do software possuir uma interface agradável, amigável e

de fácil utilização, sendo ao mesmo tempo eficiente e eficaz.

Neste contexto determinados pontos destas cinco características foram explicadas e exemplificadas antecipadamente ao entrevistado, desta forma o usuário possuía conhecimentos adequados para responder o questionário aplicado. No Quadro 23 visualizam-se as perguntas e respostas realizadas com as quatro empresas do ramo entrevistadas.

Quadro 23 – Resultados das entrevistas realizadas quanto a qualidade do software

Nome da empresa entrevistada:	Continental Turismo	SkyTour	Turisblu	Blumentur
Possui um sistema voltado para a gestão da produtividade? (Sim/Não)	Não	Não	Não	Não
Seria interessante possuir um sistema que auxilie na administração da gestão? (Sim/Não)	Sim	Sim	Sim	Sim
O sistema Viagens.Adm seria útil na sua empresa atualmente? (Sim/Não)	Sim	Sim	Sim	Sim
Seria interessante que o sistema integrasse a emissão da autorização de viagem especial com o sistema do Deter-SC? (Sim/Não)	Sim	Sim	Sim	Sim
Dê uma avaliação quanto a eficiência do sistema Viagens.Adm. (Nota de 0 À 10)	10	10	9	9
Dê uma avaliação quanto a produtividade do sistema Viagens.Adm. (Nota de 0 À 10)	9	8	9	10
Dê uma avaliação quanto a segurança do sistema Viagens.Adm. (Nota de 0 À 10)	9	9	9	10
Dê uma avaliação quanto a sua satisfação do sistema Viagens.Adm. (Nota de 0 À 10)	10	10	10	10
Dê uma avaliação quanto a usabilidade do sistema Viagens.Adm. (Nota de 0 À 10)	10	10	10	10

Fonte: elaborado pela autora (2016).

Os resultados foram satisfatórios de acordo com as entrevistas na visão geral das quatro empresas abordadas. Nenhuma das empresas entrevistadas possui um sistema voltado para as suas efetivas funcionalidades, por isso as quatro empresas aprovaram o sistema e concordaram que a utilização deste agilizará os processos, como também, os históricos das viagens serão extremamente úteis para as atividades e produtividade diária. Os gráficos e a tabela inicial do sistema também foram citados em algumas entrevistas, todas as empresas aprovaram de forma positiva a utilização destes. A usabilidade do sistema foi testada e aprovada, o design responsivo do sistema foi citado por várias empresas como um diferencial, podendo assim ser acessado com facilidade pelos smartphones e tablets dos usuários.

Uma das limitações do sistema foi a funcionalidade da emissão da autorização de viagem especial, pois essa depende da finalização da API por parte do Deter-SC, parte essa que, infelizmente, não foi concluída até a entrega deste trabalho. Essa dificuldade, porém, foi contornada com a alternativa de deixá-lo preparado para a integração, a qual já foi aprovada pelo DETER-SC e sinalizada que será feita tão breve quanto possível.

Realizando uma comparação com os trabalhos correlatos na seção 2.5, como já citado, existe uma escassez de sistemas voltados para a área da gestão dos transportes e turismo de passageiros e, com isso, não foram encontrados sistemas que abrangessem funcionalidades parecidas para os mesmos tipos de empresa. Desta forma, o único sistema encontrado nas pesquisas realizadas para este trabalho que mais se aproximasse aos requisitos do sistema aqui proposto foi o sistema Transporte Escolar 6.5. No Quadro 24 pode-se verificar a comparação de determinados pontos avaliados entre os dois sistemas.

Quadro 24 - Comparação do sistema desenvolvido com os trabalhos correlatos

	Sistema desenvolvido (Viagens.Adm)	Sistema Transporte Escolar 6.5
Plataforma	Web	Desktop
Layout responsivo	Sim	Não se aplica
Gráficos e Buscas rápidas	Sim	Não
Manutenção de Viagens	Sim	Não
Manutenção de Veículos	Sim	Sim
Manutenção de Motoristas	Sim	Sim
Controle Financeiro	Sim (Por meio de gráficos)	Sim
Relatórios	Sim	Sim (em pdf)

Fonte: elaborado pela autora (2016).

Analisando o quadro comparativo é possível concluir que o sistema desenvolvido neste trabalho abrange uma quantidade maior de funcionalidades, além de possuir a plataforma em ambiente web e layout responsivo, facilitando e agilizando o acesso do sistema em qualquer navegador web. Lembra-se ainda que os sistemas possuem focos de empresas diferenciadas, por isso a aplicação da comparação dos sistemas foi realizada apenas em enfoques funcionais e técnicos, e não em enfoques de negócio. O controle financeiro não é o foco do sistema desenvolvido neste trabalho, mas os gráficos e as buscas inteligentes auxiliam nas emissões de relatórios e visualização ágil do que é necessário para as empresas do ramo.

Contudo, pode-se verificar que o sistema desenvolvido atende a maioria das características neste seção enumeradas.

4 CONCLUSÕES

Este trabalho propôs o desenvolvimento de um sistema web que permitisse realizar a gestão, administração e manutenção de dados de empresas do ramo de transporte turístico de passageiros. Essa gestão envolve o controle e a administração por meio de viagens, motoristas, veículos, contratantes, lista de passageiros e dados de notas fiscais. Para o desenvolvimento do sistema web foi utilizada a linguagem de programação Java no ambiente Eclipse Luna, juntamente com a utilização de *frameworks* como o Play e AngularJS. Não foram encontrados problemas na utilização das ferramentas.

Os resultados alcançados foram satisfatórios. O sistema atingiu o objetivo geral do trabalho, o qual se propusera a disponibilizar para donos de empresas de transporte de passageiros uma solução que mantivesse, organizasse, auxiliasse e agilizasse as tarefas rotineiras das organizações. A possibilidade de o usuário manter e organizar viagens, listas de passageiros, motoristas, veículos, contratantes e dados das notas fiscais foi concluída. A funcionalidade de gerar gráficos quanto as informações das viagens e faturamentos da empresa foi realizada. Como também, as notificações via e-mail para os usuários de tarefas pendentes foram finalizadas.

Como limitação, destacam-se os problemas encontrados na integração com a API do sistema do Deter-SC, pois acredita-se que essa integração facilitaria e agilizaria o serviço das empresas do ramo de turismo de Santa Catarina. Desta forma, após realizadas duas visitas a sede do Deter-SC, notou-se a possibilidade de realizar uma API de integração para executar a funcionalidade de emissão de autorização de viagens especiais. Infelizmente, por parte do Deter-SC, essa API não foi finalizada até o encerramento do tempo deste trabalho, assim, a intensão desta integração é citada em vários capítulos. Em contrapartida, o sistema aqui proposto encontra-se com a base pronta para a realização desta integração.

É possível concluir que este trabalho possibilitou alternativas para a gestão de empresas do ramo turístico e de transportes de passageiros. Essa gestão vem sendo considerada cada vez mais importante no dia a dia das empresas e é de grande conhecimento de todas que um sistema próprio, no qual mantenha e organize todas as informações necessárias, o que é muito bem-vindo para agilizar os processos e oferecer mais produtividade. Por fim, este trabalho serve como base para futuros trabalhos em áreas do ramo turístico e fomenta ideias adjacentes a esse tema.

4.1 EXTENSÕES

Para as extensões deste trabalho propõe-se:

a) conclusão da integração com a futura API do Deter-SC para a emissão da autorização de viagem especial;
b) realização de um aplicativo Android e/ou IOS que mantenha os mesmo dados do sistema web;
c) incluir um espaço web específico para que os passageiros cadastrem as suas solicitações de viagens para cada empresa específica;
d) inserir máscara validadora de campos para CNPJ, CPF e CNH e incluir regras especificas para as datas das rotas das viagens.

REFERÊNCIAS

ACERENZA, Miguel Ángel. **Administração do turismo:** planejamento e direção. Bauru SP. EDUSC, 2003.

ANDRADE, Jonas Pereira de. **Planejamento dos transportes.** João Pessoa: Editora Universitária - USPB, 1994.

BEMATECH. **Sistema Bematech.** 2006. Disponivel em: <http://www.bematech.com.br>. Acesso em: 30 mar. 2016.

CHRISTOPHER, M. **Logistics and supply chain management**. New York. Irwin, 1994.

COLLIER, A. **Principles of tourism.** Ed. 3. Auckland: Longman Paul, 2004.

GSBUS. **Sistema GSbus.** 2011. Disponivel em: <http://www.systemsat.com.br/Solucoes /GSbus/)>. Acesso em: 30 mar. 2016.

HEPWORTH, M; DUCATEL, K. **Transport in the information age:** Wheels and Wires. London: Belhaven, 1992.

HUTCHINSON, B. G. **Princípios de planejamento dos sistemas de transporte urbano**. São Paulo: Guanabara Dois, 1979.

LAMB, B.; DAVIDSON, S. **Tourism and transportation in Ontario.** Canadá; Chichester. 1996.

LAWS, E. **Tourism marketing:** Service Quality and Management Perspectives. Cheltenham: Stanley Thornes, 1991.

LEIPER, N. **Tourism systems:** An Interdiciplinary Perspective. Palmerston North: NZ, 1990.

LUMSDON, L. PAGE, S.J. **Tourism and transport:** Issues and Agenda for the New Millennium. Oxford: Elsevier, 2004.

MILL, R.C. MORRISON, A.M. **The tourism system:** An Introductory Text. Englewood Cliffs. NJ: Prentince-Hall, 1992.

PAGE, Stephen J. **Transportes e turismo:** perspectivas globais. Ed. 2. Porto Alegre: Bookman, 2008.

PANIZIO, Jean Carlos. **Transpescolar:** Software para Controle e Administração de Transporte Escolar. 2006. Disponível em: <http://www.transpescolar.com>. Acesso em: 03 abr. 2016.

PENA, Rodolfo Alves. **Transportes:** Geografia. Brasil Escola. Minas Gerais, 2010. Disponível em: <http://brasilescola.uol.com.br/geografia/transportes.htm>. Acesso em: 25 mar. 2016.

PRIDEAUX, B. **The role of the transport system in destination development**. Tourism management, 2000.

QUAYLE, M. **Logistics:** An Integrated Approach. Kent: Hodder & Stoughton, 1993.

RODOSOFT. **Sistema Rodosoft.** 2007. Disponível em: <http://www.rodosoft.com.br/e-sipe/>. Acesso em: 30 mar. 2016.

SHELDON, P. **Tourism information technology.** Wallingford: CAB International, 1997.

TOTVS. **Sistema Totvs.** 2005. Disponível em: <https://www.totvs.com/software-de-gestao/servicos/>. Acesso em: 30 mar. 2016.

VOSS, Alidor. **Proprietário da empresa Continental Turismo**. Blumenau. SC. 2016.

APÊNDICE A – Descrição dos Casos de Uso

Esta seção apresenta a descrição dos casos de uso conforme previstos nos diagramas apresentados na seção 3.3.1.

Quadro 25 - Descrição do caso de uso 01 - Manter Empresa

Nome:	**UC01 - Manter Empresa**
Descrição:	Permite ao usuário cadastrar os dados de sua empresa no sistema, bem como editar ou excluir informações da mesma. Além disso, o usuário pode consultar suas informações já cadastradas.
Ator:	Usuário.
Pré-condições:	Cadastro: Não necessário. Está área pode ser acessada por qualquer usuário.
	Edição: Usuário deve possuir cadastro e realizar login no sistema.
Fluxo principal:	Sistema apresenta a tela destinada ao cadastro ou edição de empresas.
Fluxo alternativo:	Campos obrigatórios não preenchidos.
	Mensagem de alerta “Favor preencher todos os campos obrigatórios!” é apresentada.
Cenários:	Cadastro:
	a) Usuário acessa a página de cadastro de novas empresas no sistema; b) Usuário preenche todos os campos obrigatórios referentes a sua empresa; c) Usuário seleciona a opção “Começar a utilizar o sistema...”; d) Sistema inclui a empresa e apresenta a mensagem “Empresa cadastrada com sucesso. Seja bem-vindo!”; e) Sistema direciona o usuário para a área principal do seu novo sistema.
	Edição:
	a) Usuário acessa a página de edição da sua empresa; b) Usuário preenche novamente todos os campos que ele deseja alterar; c) Usuário seleciona a opção “Salvar”; d) Sistema inclui as alterações e apresenta a mensagem “Edição realizada com sucesso.”; e) Sistema direciona o usuário para a área principal do seu sistema.
Pós condições:	Usuário incluiu ou alterou dados de sua empresa.

Fonte: elaborado pela autora (2016).

Quadro 26 - Descrição do caso de uso 02 - Efetuar login

Nome:	**UC02 - Efetuar login**
Descrição:	Através da identificação por nome de usuário e senha, permite ao usuário conectar-se a área pessoal de sua empresa do sistema.
Ator:	Usuário.
Pré-condições:	Usuário deve estar cadastrado no banco de dados.
Fluxo principal:	a) Usuário preenche seu nome de usuário e senha; b) Sistema valida os dados do usuário; c) Sistema direciona o usuário para a sua página principal.
Fluxo alternativo:	a) Nome de usuário ou senha inválidos; b) Mensagem de alerta “Usuário ou senha estão incorretos” é apresentada.
Pós condições:	Usuário possuirá uma sessão a todas as áreas de seu sistema.

Fonte: elaborado pela autora (2016).

Quadro 27 - Descrição do caso de uso 03 - Alterar senha

Nome:	**UC03 – Alterar senha**
Descrição:	Permite ao usuário acessar uma tela específica à alteração de senhas informando seu *login*, senha atual, nova senha desejada e confirmando a nova senha como garantia.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada à alteração de senha.
Fluxo alternativo:	a) Campos obrigatórios não preenchidos; b) Mensagem de alerta "Favor preencher todos os campos obrigatórios!" é apresentada; c) Nome de usuário e/ou senha inválido (s); d) Mensagem de alerta "Usuário ou senha estão incorretos!" é apresentada; e) A nova senha e a sua confirmação não coincidem; f) Mensagem de alerta "As senhas digitadas não coincidem!" é apresentada;
Cenário	Edição:
	a) Usuário acessa a página de alteração de senha; b) Usuário preenche o nome de usuário, senha atual, nova senha e a confirmação da nova senha; c) Usuário seleciona a opção "Salvar"; d) Sistema inclui a alteração e apresenta a mensagem "Alteração concluída."; e) Sistema direciona o usuário para a área principal do seu sistema.
Pós condições:	Usuário alterou a senha de acesso.

Fonte: elaborado pela autora (2016).

Quadro 28 - Descrição do caso de uso 04 - Manter Viagem

Nome:	**UC04 - Manter Viagem**
Descrição:	Permite ao usuário cadastrar os dados da viagem no sistema, bem como editar ou excluir informações da mesma. Além disso, o usuário pode visualizar suas informações já cadastradas.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada ao cadastro, edição ou exclusão de viagens.
Fluxo alternativo:	a) Campos obrigatórios não preenchidos; b) Mensagem de alerta "Favor preencher todos os campos obrigatórios!" é apresentada.
Cenários:	Cadastro:
	a) Usuário acessa a página de cadastro de novas viagens; b) Usuário preenche todos os campos obrigatórios referentes a viagem; c) Usuário seleciona a opção "Salvar"; d) Sistema inclui a viagem e apresenta a mensagem "Cadastro efetuado com sucesso.".
	Edição:
	a) Usuário acessa a página de edição de determinada viagem; b) Usuário preenche novamente todos os campos que ele deseja alterar; c) Usuário seleciona a opção "Salvar"; d) Sistema inclui as alterações e apresenta a mensagem "Edição efetuada com sucesso.".
	Exclusão:
	a) Usuário acessa a página de exclusão de determinada viagem; b) Usuário seleciona a opção "Excluir"; c) Sistema apresenta uma mensagem de confirmação da ação selecionada. d) Caso a resposta seja positiva, o sistema exclui a viagem e apresenta a mensagem "A viagem foi excluída com sucesso.".
Pós condições:	Usuário incluiu, alterou ou excluiu dados da viagem.

Fonte: elaborado pela autora (2016).

Quadro 29 - Descrição do caso de uso 05 - Manter veículo

Nome:	**UC05 – Manter veículo**
Descrição:	Permite ao usuário cadastrar os dados de seu veículo no sistema, bem como editar ou excluir informações do mesmo. Além disso, o usuário pode visualizar os veículos já cadastrados.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada ao cadastro, edição ou exclusão de veículos.
Fluxo alternativo:	a) Campos obrigatórios não preenchidos; b) Mensagem de alerta “Favor preencher todos os campos obrigatórios!” é apresentada.
Cenários:	Cadastro:
	a) Usuário acessa a página de cadastro de novos veículos; b) Usuário preenche todos os campos obrigatórios referentes ao veículo; c) Usuário seleciona a opção “Salvar”; d) Sistema inclui o veículo e apresenta a mensagem “Cadastro efetuado com sucesso.”.
	Edição:
	a) Usuário acessa a página de edição de determinado veículo; b) Usuário preenche novamente todos os campos que ele deseja alterar; c) Usuário seleciona a opção “Salvar”; d) Sistema inclui as alterações e apresenta a mensagem “Edição efetuada com sucesso.”.
	Exclusão:
	a) Usuário acessa a página de exclusão de determinado veículo; b) Usuário seleciona a opção “Excluir”; c) Sistema apresenta uma mensagem de confirmação da ação selecionada. d) Caso a resposta seja positiva, o sistema exclui a viagem e apresenta a mensagem “O veículo foi excluído com sucesso.”.
Pós condições:	Usuário incluiu, alterou ou excluiu dados do veículo.

Fonte: elaborado pela autora (2016).

Quadro 30 - Descrição do caso de uso 06 - Manter motorista

Nome:	**UC06 – Manter motorista**
Descrição:	Permite ao usuário cadastrar os dados de um motorista no sistema, bem como editar ou excluir informações do mesmo. Além disso, o usuário pode visualizar os motoristas já cadastrados.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada ao cadastro, edição ou exclusão de motoristas.
Fluxo alternativo:	a) Campos obrigatórios não preenchidos; b) Mensagem de alerta “Favor preencher todos os campos obrigatórios!” é apresentada.
Cenários:	Cadastro:
	a) Usuário acessa a página de cadastro de novos motoristas; b) Usuário preenche todos os campos obrigatórios referentes ao motorista; c) Usuário seleciona a opção “Salvar”; d) Sistema inclui o motorista e apresenta a mensagem “Cadastro efetuado com sucesso.”.
	Edição:
	a) Usuário acessa a página de edição de determinado motorista; b) Usuário preenche novamente todos os campos que ele deseja alterar; c) Usuário seleciona a opção “Salvar”; d) Sistema inclui as alterações e apresenta a mensagem “Edição efetuada com sucesso.”.
	Exclusão:
	a) Usuário acessa a página de exclusão de determinado motorista; b) Usuário seleciona a opção “Excluir”; c) Sistema apresenta uma mensagem de confirmação da ação selecionada. d) Caso a resposta seja positiva, o sistema exclui a viagem e apresenta a mensagem “O motorista foi excluído com sucesso.”.
Pós condições:	Usuário incluiu, alterou ou excluiu dados do motorista.

Fonte: elaborado pela autora (2016).

Quadro 31 - Descrição do caso de uso 07 - Manter contratante

Nome:	**UC07 – Manter contratante**
Descrição:	Permite ao usuário cadastrar os dados de seu contratante no sistema, bem como editar ou excluir informações do mesmo. Além disso, o usuário pode visualizar os contratantes já cadastrados.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada ao cadastro, edição ou exclusão de contratantes.
Fluxo alternativo:	a) Campos obrigatórios não preenchidos; b) Mensagem de alerta “Favor preencher todos os campos obrigatórios!” é apresentada.
Cenários:	Cadastro:
	a) Usuário acessa a página de cadastro de novos contratantes; b) Usuário preenche todos os campos obrigatórios referentes ao contratante; c) Usuário seleciona a opção “Salvar”; d) Sistema inclui o contratante e apresenta a mensagem “Cadastro efetuado com sucesso.”.
	Edição:
	a) Usuário acessa a página de edição de determinado contratante; b) Usuário preenche novamente todos os campos que ele deseja alterar; c) Usuário seleciona a opção “Salvar”; d) Sistema inclui as alterações e apresenta a mensagem “Edição efetuada com sucesso.”.
	Exclusão:
	a) Usuário acessa a página de exclusão de determinado contratante; b) Usuário seleciona a opção “Excluir”; c) Sistema apresenta uma mensagem de confirmação da ação selecionada. d) Caso a resposta seja positiva, o sistema exclui o contratante e apresenta a mensagem “O contratante foi excluído com sucesso.”.
Pós condições:	Usuário incluiu, alterou ou excluiu dados do contratante.

Fonte: elaborado pela autora (2016).

Quadro 32 - Descrição do caso de uso 08 - Manter lista de passageiros

Nome:	**UC08 – Manter lista de passageiros**
Descrição:	Permite ao usuário cadastrar os dados da lista de passageiros sistema, bem como editar ou consultar informações da mesma.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* e possuir a viagem cadastrada no sistema.
Fluxo principal:	Sistema apresenta a tela destinada ao cadastro ou edição de lista de passageiros.
Fluxo alternativo:	a) Campos obrigatórios não preenchidos; b) Mensagem de alerta "Favor preencher todos os campos obrigatórios!" é apresentada.
Cenários:	Cadastro:
	a) Usuário acessa a página de cadastro de novas listas de passageiros; b) Usuário preenche todos os campos obrigatórios referentes a lista; c) Usuário seleciona a opção "Salvar"; d) Sistema inclui a lista de passageiros e apresenta a mensagem "Cadastro efetuado com sucesso.".
	Edição:
	a) Usuário acessa a página de edição de determinada lista de passageiros; b) Usuário preenche novamente todos os campos que ele deseja alterar; c) Usuário seleciona a opção "Salvar"; d) Sistema inclui as alterações e apresenta a mensagem "Edição efetuada com sucesso.".
Pós condições:	Usuário incluiu ou alterou dados da lista de passageiros.

Fonte: elaborado pela autora (2016).

Quadro 33 - Descrição do caso de uso 09 - Emitir autorização de viagem especial

Nome:	**UC09 – Emitir autorização de viagem especial**
Descrição:	Permite ao usuário emitir a autorização de viagem especial por meio dos dados já cadastrados anteriormente no sistema.
Ator:	Usuário, Sistema do Deter-SC.
Pré-condições:	Usuário deve realizar *login* e possuir todos os dados necessários cadastrados no sistema. Cadastro de dados necessários para a emissão: viagem, veículo, motorista, contratante, nota fiscal e lista de passageiros.
Fluxo principal:	Sistema apresenta um local para a seleção da emissão da autorização de determinada viagem especial.
Fluxo alternativo:	a) Dados obrigatórios não cadastrados; b) Mensagem de alerta "Favor preencher todos os dados obrigatórios para realizar a emissão da autorização de viagem especial!" é apresentada.
Cenários:	Emissão:
	a) Usuário acessa a página de busca de viagens confirmadas; b) Usuário seleciona determinada viagem; c) Usuário seleciona a opção "Emitir autorização de viagem especial"; d) Sistema acessa o site do Deter-SC, emite a autorização e apresenta a mensagem "Emissão efetuada com sucesso."; e) Sistema abre uma nova janela do navegador com a autorização da viagem especial e a lista de passageiros geradas em PDF.
Pós condições:	Usuário emitiu a autorização de viagem especial. Sistema cadastrou a autorização de viagem especial e lista de passageiros em seu banco de dados.

Fonte: elaborado pela autora (2016).

Quadro 34 - Descrição do caso de uso 10 - Registrar dados da nota fiscal

Nome:	**UC10 – Registrar dados da nota fiscal**
Descrição:	Permite ao usuário cadastrar os dados da nota fiscal de determinada viagem no sistema, bem como editar e consultar informações da mesma.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* e possuir uma viagem cadastrada no sistema.
Fluxo principal:	Sistema apresenta a tela destinada ao cadastro ou edição de dados da nota fiscal.
Fluxo alternativo:	a) Campos obrigatórios não preenchidos; b) Mensagem de alerta "Favor preencher todos os campos obrigatórios!" é apresentada.
Cenários:	Cadastro:
	a) Usuário acessa a página de cadastro de nota fiscal de determinada viagem; b) Usuário preenche todos os campos obrigatórios referentes aos dados da nota fiscal; c) Usuário seleciona a opção "Salvar"; d) Sistema inclui os dados da nota fiscal e apresenta a mensagem "Cadastro efetuado com sucesso.".
	Edição:
	a) Usuário acessa a página de edição de determinados dados da nota fiscal; b) Usuário preenche novamente todos os campos que ele deseja alterar; c) Usuário seleciona a opção "Salvar"; d) Sistema inclui as alterações e apresenta a mensagem "Edição efetuada com sucesso.".
Pós condições:	Usuário incluiu ou alterou dados da nota fiscal.

Fonte: elaborado pela autora (2016).

Quadro 35 - Descrição do caso de uso 11 - Emitir relatório de viagens em aberto

Nome:	**UC11 – Emitir relatório de viagens em aberto**
Descrição:	Permite ao usuário emitir relatórios de viagens com o status "confirmado".
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada a emissão de relatórios.
Pós condições:	Usuário emitiu um novo relatório.

Fonte: elaborado pela autora (2016).

Quadro 36 - Descrição do caso de uso 12 - Emitir relatório de viagens confirmadas

Nome:	**UC12 – Emitir relatório de viagens confirmadas**
Descrição:	Permite ao usuário emitir relatórios de viagens com o status "em aberto".
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada a emissão de relatórios.
Pós condições:	Usuário emitiu um novo relatório.

Fonte: elaborado pela autora (2016).

Quadro 37 - Descrição do caso de uso 13 - Consultar histórico

Nome:	**UC13 – Consultar históricos de viagens, veículos, motoristas e contratantes**
Descrição:	Permite ao usuário consultar históricos de viagens, veículos, motoristas e contratantes.
Ator:	Usuário.
Pré-condições:	Usuário deve realizar *login* no sistema.
Fluxo principal:	Sistema apresenta a tela destinada a busca e consulta de históricos.
Pós condições:	Usuário consultou histórico de algum dado.

Fonte: elaborado pela autora (2016).

Quadro 38 - Descrição do caso de uso 14 – Notificar tarefas pendentes

Nome:	**UC14 – Notificar tarefas pendentes**
Descrição:	O sistema notifica o usuário por e-mail referente a alguma tarefa em atraso. As notificações serão referentes as viagens ou emissões de autorização de nota fiscal.
Fluxo principal:	Usuário deve possui o seu e-mail pessoal cadastrado no sistema.
Pós condições:	Sistema notificou o usuário por alguma tarefa em atraso.

Fonte: elaborado pela autora (2016).

APÊNDICE B – Dicionário de Dados

Este apêndice apresenta o detalhamento das entidades conforme previsto do diagrama entidade relacionamento apresentados na seção 3.3.3.

Quadro 39 - Dicionário de dados da entidade `Empresa`

Entidade: Empresa					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_empresa	INT	Numérico	-	Não Nulo - Auto increment	PK
nm_empresa	VARCHAR	Texto	80	Não Nulo	-
email_empresa	VARCHAR	Texto	100	Não Nulo	-
cnpj_empresa	VARCHAR	Texto	20	Não Nulo	-
pw_empresa	VARCHAR	Texto	20	Não Nulo	-
logo_empresa	MEDIUMBLOB	Lista	-	-	-

Fonte: elaborado pela autora (2016).

Quadro 40 - Dicionário de dados da entidade `Motorista`

Entidade: Motorista					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_motorista	INT	Numérico	-	Não Nulo - Auto increment	PK
nm_motorista	VARCHAR	Texto	45	Não Nulo	-
cnh_motorista	VARCHAR	Texto	15	Não Nulo	-
email_motorista	VARCHAR	Texto	100	-	-
ds_end_motorista	VARCHAR	Texto	200	-	-
tel_motorista	VARCHAR	Texto	12	-	-
cel_motorista	VARCHAR	Texto	12	-	-
id_empresa	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).

Quadro 41 - Dicionário de dados da entidade `Veículo`

Entidade: Veiculo					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_veiculo	INT	Numérico	-	Não Nulo - Auto increment	PK
placa_veiculo	VARCHAR	Texto	50	Não Nulo	-
marca_veiculo	VARCHAR	Texto	50	Não Nulo	-
ano_veiculo	INT	Numérico	-	Não Nulo	-
lotacao_veiculo	INT	Numérico	-	-	-
id_empresa	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).

Quadro 42 - Dicionário de dados da entidade `Contratante`

Entidade: Contratante					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_contratante	INT	Numérico	-	Não Nulo - Auto increment	PK
nm_contratante	VARCHAR	Texto	100	Não Nulo	-
cpf_contratante	VARCHAR	Texto	20	Não Nulo	-
email_contratante	VARCHAR	Texto	100	-	-
tel_contratante	VARCHAR	Texto	12	-	-
cel_contratante	VARCHAR	Texto	12	-	-

Fonte: elaborado pela autora (2016).

Quadro 43 - Dicionário de dados da entidade `TrechoRotaViagem`

Entidade: TrechoRotaViagem					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_trecho	INT	Numérico	-	Não Nulo - Auto increment	PK
nivel_trecho	INT	Numérico	-	Não Nulo	-
hr_partida_trecho	TIME	Hora	-	Não Nulo	-
dt_partida_trecho	DATE	Data	-	Não Nulo	-
km_trecho	FLOAT	Numérico	-	-	-
Viagem_id_viagem	INT	Numérico	-	Não Nulo	FK
Cidade_id_cidade	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).

Quadro 44 - Dicionário de dados da entidade `Estado`

Entidade: Estado					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_estado	INT	Numérico	-	Não Nulo - Auto increment	PK
nm_estado	VARCHAR	Texto	30	Não Nulo	-
uf_estado	CHAR	Texto	2	Não Nulo	-

Fonte: elaborado pela autora (2016).

Quadro 45 - Dicionário de dados da entidade `Cidade`

Entidade: Cidade					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_cidade	INT	Numérico	-	Não Nulo - Auto increment	PK
nm_cidade	VARCHAR	Texto	45	Não Nulo	-
Estado_id_estado	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).

Quadro 46 - Dicionário de dados da entidade `Viagem`

Entidade: Viagem					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_viagem	INT	Numérico	-	Não Nulo - Auto increment	PK
ds_viagem	VARCHAR	Texto	70	Não Nulo	-
st_viagem	CHAR	Texto	1	Não Nulo	-
vl_viagem	FLOAT	Numérico	-	Não Nulo	-
inf_viagem	VARCHAR	Texto	200	-	-
Veiculo_id	INT	Numérico	-	Não Nulo	FK
Empresa_id	INT	Numérico	-	Não Nulo	FK
Motorista_id	INT	Numérico	-	Não Nulo	FK
Contratante_id	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).

Quadro 47 - Dicionário de dados da entidade `Passageiro`

Entidade: Passageiro					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_passageiro	INT	Numérico	-	Não Nulo - Auto increment	PK
nm_passageiro	VARCHAR	Texto	45	Não Nulo	-
cpf_passageiro	VARCHAR	Texto	11	Não Nulo	-

Fonte: elaborado pela autora (2016).

Quadro 48 - Dicionário de dados da entidade `ListaPassageiros`

Entidade: ListaPassageiros					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
Viagem_id	INT	Numérico	-	Não Nulo	FK
Passageiro_id	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).

Quadro 49 - Dicionário de dados da entidade `NotaFiscal`

Entidade: NotaFiscal					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
id_nota	INT	Numérico	-	Não Nulo - Auto increment	PK
nr_nota	INT	Numérico	-	Não Nulo	-
sr_nota	VARCHAR	Texto	20	Não Nulo	-
dt_nota	DATE	Data	-	Não Nulo	-
vl_nota	FLOAT	Numérico	-	-	-
Viagem_id	INT	Numérico	-	Não Nulo	FK
Empresa_id	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).

Quadro 50 - Dicionário de dados da entidade `AutorizacaoViagemEspecial`

Entidade: AutorizacaoViagemEspecial					
Atributo	**Tipo**	**Domínio**	**Tamanho**	**Restrições**	**PK/FK**
nr_lacre_aut	INT	Numérico	-	Não Nulo - Auto increment	PK
st_aut	BOOLEAN	Texto	-	Não Nulo	-
Viagem_id	INT	Numérico	-	Não Nulo	FK
Empresa_id	INT	Numérico	-	Não Nulo	FK

Fonte: elaborado pela autora (2016).