

**FACULDADE DE TECNOLOGIA DE SÃO JOSÉ DOS CAMPOS  
FATEC PROFESSOR JESSEN VIDAL**

**JOÃO VITOR FERREIRA GARCIA**

**SISTEMA LOGÍSTICO DE ROTEIRIZAÇÃO**

São José dos Campos  
2018

**JOÃO VITOR FERREIRA GARCIA**

## **SISTEMA LOGÍSTICO DE ROTEIRIZAÇÃO**

Trabalho de Graduação apresentado à Faculdade de Tecnologia de São José dos Campos, como parte dos requisitos necessários para a obtenção do título de Tecnólogo em Banco de Dados.

**Orientador: Me. Lucas Gonçalves Nadalete**

São José dos Campos  
2018

**Dados Internacionais de Catalogação-na-Publicação (CIP)**  
**Divisão de Informação e Documentação**

GARCIA, João Vitor Ferreira  
Sistema Logístico de Roteirização  
São José dos Campos, 2018.  
999f. (número total de folhas do TG)

Trabalho de Graduação – Curso de Tecnologia em Banco de Dados.  
FATEC de São José dos Campos: Professor Jessen Vidal, 2018.  
Orientador: Prof. Lucas Gonçalves Nadalete.

1. Roteirização. 2. Logística. 3. Software. I. Faculdade de Tecnologia. FATEC de São José dos Campos: Professor Jessen Vidal. Divisão de Informação e Documentação. II. Título

**REFERÊNCIA BIBLIOGRÁFICA**

GARCIA, João Vitor Ferreira. **Sistema Logístico de Distribuição de Produtos**. 2018. 999f. Trabalho de Graduação - FATEC de São José dos Campos: Professor Jessen Vidal.

**CESSÃO DE DIREITOS**

NOME(S) DO(S) AUTOR(ES): João Vitor Ferreira Garcia  
TÍTULO DO TRABALHO: Sistema Logístico de Roteirização  
TIPO DO TRABALHO/ANO: Trabalho de Graduação/2018.

É concedida à FATEC de São José dos Campos: Professor Jessen Vidal permissão para reproduzir cópias deste Trabalho e para emprestar ou vender cópias somente para propósitos acadêmicos e científicos. O autor reserva outros direitos de publicação e nenhuma parte deste Trabalho pode ser reproduzida sem a autorização do autor.

---

João Vitor Ferreira Garcia  
Rua do Porto 718, Caçapava – São Paulo

**JOÃO VITOR FERREIRA GARCIA**

## **SISTEMA LOGÍSTICO DE ROTEIRIZAÇÃO**

Trabalho de Graduação apresentado à Faculdade de Tecnologia de São José dos Campos, como parte dos requisitos necessários para a obtenção do título de Tecnólogo em Banco de Dados

---

**Me. Lucas Gonçalves Nadalete - Fatec SJC**

---

**Titulação, Nome do Componente da Banca - Sigla da Instituição**

---

**Titulação, Nome do Componente da Banca - Sigla da Instituição**

\_\_\_\_/\_\_\_\_/\_\_\_\_

**DATA DA APROVAÇÃO**

Dedicatória (opcional)

O autor oferece a obra (elemento sem título e sem indicativo numérico), ou presta homenagem a alguém, de forma clara e breve em folha única.

## **AGRADECIMENTOS**

Na página de agradecimentos o autor dirige palavras de reconhecimento àqueles que contribuíram para a elaboração do trabalho. O conteúdo não deve ultrapassar uma página e por isso, é necessário que ele seja sucinto e objetivo.

O texto deve ser escrito em Times New Roman, Tamanho 12, Alinhamento Justificado, Espaçamento entre linhas de 1,5 linhas e com recuo de parágrafo de 1,25 cm.

Epígrafe (opcional)

“É citada uma sentença escolhida pelo autor (elemento sem título e sem indicativo numérico), que deve guardar coerência com o tema abordado na obra.”

Nome do autor

## RESUMO

Apresentação concisa dos pontos relevantes do documento deve ser exposta no resumo. No presente caso o resumo será informativo, assim deverá ressaltar o objetivo, a metodologia, os resultados e as conclusões do documento. A ordem desses itens depende do tratamento que cada item recebe no documento original. O resumo deve ser composto por uma sequência de frases concisas, afirmativas e não em enumeração de tópicos. Deve ser escrita em parágrafo único e espaçamento de 1,5 linhas. A primeira frase deve ser significativa, explicando o tema principal do documento. Deve-se usar o verbo na voz ativa e na terceira pessoa do singular. Quanto a sua extensão, o resumo deve possuir de 150 a 500 palavras.

**Palavras-Chave:** Roteirização, Logística, Software, Programação.



## ABSTRACT

O abstract é o resumo da obra em língua estrangeira, que basicamente segue o mesmo conceito e as mesmas regras que o texto em português. Recomenda-se que para o texto do abstract o autor traduza a versão do resumo em português e faça, se necessário, os ajustes referentes à conversão dos idiomas. É importante observar que o título e texto NÃO DEVEM estar em itálico.

**Keywords:** Recomenda-se que o autor traduza para o inglês as Palavras-Chave em português e faça, se necessário, os ajustes referentes à conversão dos idiomas.

## LISTA DE FIGURAS

Figura 1 . Era do Transporte Moderno, isolamento das empresas.....	17
Figura 2 . Era da Logística Empresarial.....	17
Figura 3 . Era da Cadeia de Suprimentos.....	18
Figura 4 . Era das redes de Suprimentos.....	18
Figura 5 . Participação do Modal Rodoviário nas Empresas.....	19
Figura 6 . Distribuição dos Custos Logísticos nas Empresas.....	20
Figura 7 . Grau de Priorização das Empresas na Redução de Custos logísticos.....	21
Figura 8 . Arquitetura da Solução - BackEnd.....	46
Figura 9 . Arquitetura da Solução: FrontEnd.....	46
Figura 10 . Arquitetura da Solução: BackEnd.....	48
Figura 11 . Arquitetura da Solução: Implantação de Projeto.....	49
Figura 12 . Diagrama de Componentes.....	50
Figura 13 . Diagrama de Classes: Ceps.....	51
Figura 14 . Diagrama de Classes: Consulta Ceps.....	52
Figura 15 . Diagrama de Classes: Pessoa.....	53
Figura 16 . Diagrama de Classes: Rota.....	54
Figura 17 . Definição de Controlador e URL.....	54
Figura 18 . Definição de Acesso ao Método usando @RequestMapping.....	55
Figura 19 . Definição Injeção de Dependências.....	56
Figura 20 . Definição Injeção de Serviço.....	56
Figura 21 . Fragmento da Classe PessoaService.....	56
Figura 22 . Interface Pessoa Repository.....	57
Figura 23 . Entidade Pessoa.....	58
Figura 24 . Página de Geração de Rotas.....	59
Figura 25 . Cabeçalho da Página de Geração de Rotas.....	60
Figura 26 . Fragmento da Página de Geração de Rotas.....	60
Figura 27 . Fragmento de Código, Controlador Pagina de Geração de Rotas.....	61
Figura 28 . Fragmento da Classe CepService.....	61
Figura 29 . Tela para Geração e Rotas, com Endereços Inseridos.....	62
Figura 30 . Tela para Geração e Rotas, com Endereços Inseridos.....	62
Figura 31 . Método para Remoção de Endereço da Lista.....	63
Figura 32 . Tela de Geração de Rotas após executada a Geração.....	63
Figura 33 . Tela de Geração de Rotas após executada a Geração.....	64
Figura 34 . Modelo Entidade Relacionamento.....	65
Figura 35 . Propriedade do Liquibase em application.properties.....	71
Figura 36 . Fragmento do Arquivo liquibase-changelog.xml.....	72
Figura 37 . Fragmento do Arquivo liquibase-changelog.xml.....	72
Figura 38 . Classe UserPrincipal e Alguns Métodos.....	73
Figura 39 . Método Generate Token de JWTTokenProvider.....	74
Figura 40 . Classe JWTAuthenticationEntryPoint.....	75
Figura 41 . Método FilterInternal de JWTAuthenticationFilter.....	75
Figura 42 . Diagrama exemplificando Implementação de Segurança.....	76
Figura 43 . Bean para Configuração de Cors.....	76
Figura 44 . Método Configure da Classe SecurityConfig.....	77
Figura 45 . Diagrama exemplificando Implementação de Segurança.....	77
Figura 46 . Utilização da Anotação @PreAuthorize.....	78
Figura 47 . Tela de Login.....	79

Figura 48 . Tela de Cadastro.....	80
Figura 49 . Menu Principal.....	80
Figura 50 . Listagem de Rotas Criadas.....	81
Figura 51 . Página de detalhamento da Rota.....	81
Figura 52 . Rota Criada Aberta no Google Maps.....	82
Figura 53 . Página para Gerar a Rota.....	82
Figura 54 . Página Após Rota Ser Gerada.....	83
Figura 55 . Página de Endereço.....	83
Figura 56 . Página Empresa.....	84
Figura 57 . Página de Filiais da Empresa.....	84
Figura 58 . Página de Listagem Funcionários da Empresa.....	85
Figura 59 . Página de Região.....	85
Figura 60 . Página para Alterar a Região.....	86
Figura 61 . Página para Alterar a Região.....	87
Figura 62 . Página de Detalhamento do Usuário.....	87
Figura 63 . Página de Alteração do Usuário.....	88
Figura 64 . Botões para Salvar e Cancelar alteração do Usuário.....	88

## LISTA DE TABELAS

Tabela 1 . Comparativo de produtividade dos veículos com e sem a roteirização.....	21
Tabela 2 . Atendimentos no Prazo de um Determinado Período Com e Sem a Roteirização..	22
Tabela 3 . Requisitos Funcionais do Projeto.....	27
Tabela 4 . Requisitos Não-Funcionais do Projeto.....	27
Tabela 5 . Lista de Personas com seus comportamentos, necessidades e objetivos.....	28
Tabela 6 . User Story - Otimização de Rota.....	29
Tabela 7 . User Story - Recuperar Rota.....	29
Tabela 8 . User Story - Identificar Entregas Fora da Região de Distribuição da Empresa.....	29
Tabela 9 . User Story - Solicitar Geração de Rotas a partir de Outro Sistema.....	30
Tabela 10 . User Story - Excluir rota gerada.....	30
Tabela 11 . User Story - Cadastrar Usuário.....	30
Tabela 12 . User Story - Alterar Usuário.....	30
Tabela 13 . User Story - Pesquisar Usuários.....	31
Tabela 14 . User Story - Deletar Usuário.....	31
Tabela 15 . User Story - Consultar Cep.....	31
Tabela 16 . User Story - Abrir Rota no Maps.....	31
Tabela 17 . User Story - Consultar Ceps.....	32
Tabela 18 . User Story - Cadastrar Pessoa.....	32
Tabela 19 . User Story - Alterar Pessoa.....	32
Tabela 20 . User Story - Pesquisar Pessoa.....	33
Tabela 21 . User Story - Cadastrar Empresa.....	33
Tabela 22 . User Story - Alterar Empresa.....	33
Tabela 23 . User Story - Pesquisar Empresa.....	33
Tabela 24 . User Story - Cadastrar Funcionário.....	34
Tabela 25 . User Story - Alterar Funcionário.....	34
Tabela 26 . User Story - Deletar Funcionário.....	34
Tabela 27 . User Story - Pesquisar Funcionário.....	34
Tabela 28 . User Story - Pesquisar Região.....	35
Tabela 29 . User Story - Alterar Região.....	35
Tabela 30 . User Story - Deletar Região.....	35
Tabela 31 . User Story - Efetuar Login.....	35
Tabela 32 . User Story - Efetuar Login.....	36
Tabela 33 . Dicionário de Dados: Tabela Cargo.....	66
Tabela 34 . Dicionário de Dados: Tabela Cep.....	66
Tabela 35 . Dicionário de Dados: Tabela Cidade.....	66
Tabela 36 . Dicionário de Dados: Tabela Empresa.....	67
Tabela 37 . Dicionário de Dados: Tabela Endereço.....	67
Tabela 38 . Dicionário de Dados: Tabela Estado.....	67
Tabela 39 . Dicionário de Dados: Tabela Funcionário.....	68
Tabela 40 . Dicionário de Dados: Tabela Map_config.....	68
Tabela 41 . Dicionário de Dados: Tabela Pessoa.....	68
Tabela 42 . Dicionário de Dados: Tabela Região.....	69
Tabela 43 . Dicionário de Dados: Tabela Roles.....	69
Tabela 44 . Dicionário de Dados: Tabela Telefone.....	69
Tabela 45 . Dicionário de Dados: Tabela Tipo_Empresa.....	70
Tabela 46 . Dicionário de Dados: Tabela Tipo_Pessoa.....	70
Tabela 47 . Dicionário de Dados: Tabela User.....	70
Tabela 48 . Dicionário de Dados: Tabela User_Role.....	71

## LISTA DE ABREVIATURAS E SIGLAS

CEL	Centro de estudos em Logística
CD	Centro de Distribuição
Copeead	Instituto de Pós-Graduação e Pesquisa em Administração
FK	<i>Foreign Key</i>
HTML	Hypertext Markup Language
ID	Número identificador
ILOS	Instituto de Logística e <i>Supply Chain</i>
JVM	<i>Java Virtual Machine</i>
MVC	<i>Model View Controller</i>
PK	<i>Primary Key</i>
TI	Tecnologia da Informação

## SUMÁRIO

<b>1. INTRODUÇÃO.....</b>	<b>16</b>
1.1. Problema em estudo.....	23
1.2. Relevância do Trabalho.....	23
1.3. Objetivo do Geral.....	23
1.4. Objetivos Específicos.....	23
1.5. Proposta Metodológica.....	24
1.6. Conteúdo do Trabalho.....	25
<b>2. REQUISITOS IDENTIFICADOS E CONTEXTUALIZAÇÃO TECNOLÓGICA....</b>	<b>26</b>
2.1. Especificação de requisitos.....	26
2.1.1. Requisitos Funcionais:.....	27
2.1.2. Requisitos Não-Funcionais:.....	27
2.2. Especificações baseadas em User Stories.....	28
2.2.1. BackLog.....	36
2.3. Tecnologias Aplicadas .....	37
2.3.1. BackEnd.....	37
2.3.1.1. Linguagem.....	37
2.3.1.2. Formato para Transmissão de Dados.....	38
2.3.1.3. Maven.....	38
2.3.1.4. Spring.....	38
2.3.1.5. Banco de Dados.....	39
2.3.1.6. Plugins para Base de Dados.....	39
2.3.1.7. Demais Dependências utilizadas.....	40
2.3.1.8. Softwares utilizados.....	40
2.3.1.9. Recursos Externos.....	41
2.3.2. FrontEnd.....	41
2.3.2.1. NPM.....	41
2.3.2.2. Ionic.....	41
2.3.2.3. HTML5.....	42
2.3.2.4. CSS.....	42
2.3.2.5. TypeScript.....	43
2.3.2.6. AngularJS.....	43
2.3.2.7. Cordova.....	43
2.3.2.8. Softwares utilizados.....	43
2.3.3. Versionamento.....	44
<b>3. DESENVOLVIMENTO.....</b>	<b>45</b>
3.1. Padrão do Projeto.....	45
3.2. Arquitetura da Solução.....	45
3.2.1. Arquitetura da Solução - FrontEnd.....	46
3.2.2. Arquitetura da Solução - BackEnd.....	47
3.2.3. Arquitetura da Solução - Implantação.....	48
3.3. Arquitetura do Software.....	49
3.3.1. Diagrama de Componentes.....	50
3.3.2. Diagramas de Classes.....	51
3.3.3. Exemplificação de Funcionamento do BackEnd.....	54
3.3.4. Exemplificação de Funcionamento do FrontEnd.....	59

3.4. Modelagem e Gestão dos Dados.....	65
3.4.1. Modelo de Entidade Relacionamento.....	65
3.4.2. Dicionário de Dados.....	66
3.4.3. Liquibase.....	71
3.5. Segurança.....	73
3.5.1. Visão Geral - Segurança.....	77
3.6. Visão geral do Sistema.....	79
<b>4. VALIDAÇÃO E ANÁLISE DOS DOS RESULTADOS OBTIDOS.....</b>	<b>89</b>
4.1. Métricas do sistema.....	89
4.2. Técnicas de Verificação e Validação aplicadas.....	89
4.3. Resultados de Verificação e Validação.....	89
<b>5. CONCLUSÃO.....</b>	<b>90</b>
5.1. Principais contribuições.....	90
5.2. Considerações Gerais, Limitações e Dificuldades.....	90
5.3. Sugestões de trabalhos futuros.....	90
<b>REFERÊNCIAS BIBLIOGRÁFICAS.....</b>	<b>91</b>

## 1. INTRODUÇÃO

No cenário atual do mercado, empresas têm que trabalhar com prazos reduzidos, gerenciar seus estoques, reduzir o valor da produção de seus produtos e economizar com o transporte de suas mercadorias. Devido ao grande número e complexidade de tarefas à serem realizadas pelas empresas, cada vez mais elas buscam terceirizar o transporte de produtos (BRANSKI, 2008), pois com a terceirização surge a oportunidade de redução de custos logísticos (PERÇIN; MIN, 2013).

A logística é uma área vital e de extrema importância para as empresas, pois com o ambiente globalizado em que ocorrem mudanças constantes, as operações logísticas acabam tornando-se cada vez mais complexas, onerosas e importantes sob o ponto de vista estratégico (FLEURY, 1999).

Empresas visam maximizar cada vez mais seu lucro, portanto faz-se necessário atentarem-se as atividades logísticas de planejamento, abastecimento, mão de obra, e entrega final. Pois, planejando adequadamente estes itens, a empresa consegue reduzir custos e consequentemente repassar um menor valor a seus clientes, maximizando seu lucro e tornando-se mais competitiva frente a seus concorrentes. (DORNIER et al, 2000).

Desde a antiguidade, a logística já fazia parte das guerras, devido ao deslocamento de tropas, suprimentos e armamentos, por grandes distâncias além do longo período de duração das guerras. Por conta destes fatores precisava-se de um planejamento, organização e execução de tarefas.

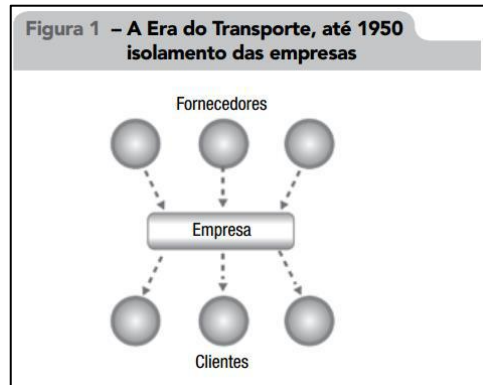
Até os anos 50, os mercados eram muito limitados, funções logísticas estavam dispersas entre os diversos departamentos da organização. A área industrial geria o planejamento e o controle, já a área administrativa comandava o estoque, o departamento de vendas que cuidava dos pedidos e o departamento financeiro que era responsável pelas compras (ROMERO; SOUZA, 2015).

Esse procedimento ocasionava conflitos e descontrole, já que departamentos que não tinham uma base de conhecimento específica para lidar com essas tarefas acabavam tornando-se responsáveis. Dessa forma as corporações acabavam sendo afetadas negativamente, com perda de vantagem competitiva e prejudicando os processos de entrega de valor para o cliente. (POZO, 2010 p. 3).



A Figura 1 mostra graficamente o isolamento das empresas na Era do Transporte até 1950.

**Figura 1. Era do Transporte Moderno, isolamento das empresas.**



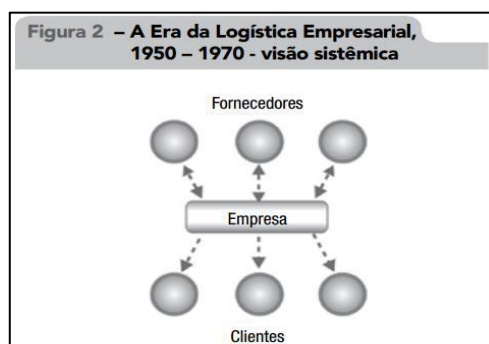
Fonte:

Entre os anos de 1950 e 1970, a área da logística empresarial, que até esta época era apenas teoria, começou a ser realmente utilizada na prática, com o intuito de melhorar o resultado das empresas como evidenciado por POZO (2010, p. 5):

“A nova situação econômica do pós-guerra, e principalmente no início dos anos 50, era um forte instrumental para fomentar o interesse em Logística. O crescimento econômico substancial que decorreu das novas atitudes e concepções após a Segunda Guerra foi seguido de recessão e um período de prolongada pressão nos ativos das empresas e de seus lucros. Os novos conceitos logísticos, que começavam a aflorar na mente dos administradores, ofereciam a oportunidade de melhorar os resultados das empresas.” (POZO, 2010 p. 5).

Pozo (2010, p. 7) destaca que com o desenvolvimento da tecnologia, os problemas logísticos tornam-se cada vez mais complexos, exigindo uma visão sistêmica da organização e do mercado devido ao relacionamento bidirecional entre empresa e fornecedores, conforme demonstra a Figura 2, neste cenário.

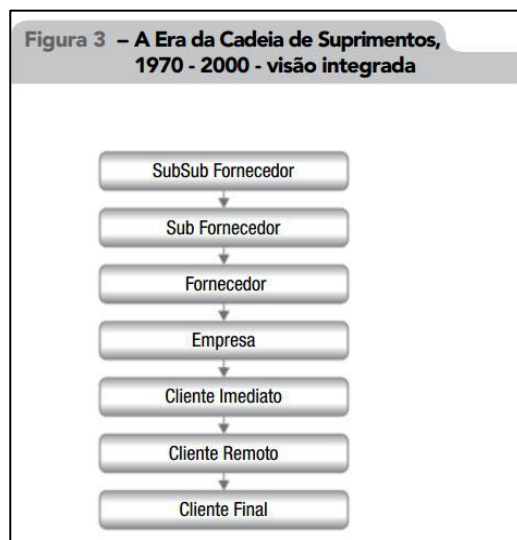
**Figura 2. Era da Logística Empresarial**



Fonte:

Machline (2011) através de uma visão integrada denominou no período entre os anos de 1970 a 2000 como a Era da Cadeia de Suprimentos, conforme apresentado na Figura 3.

**Figura 3. Era da Cadeia de Suprimentos**

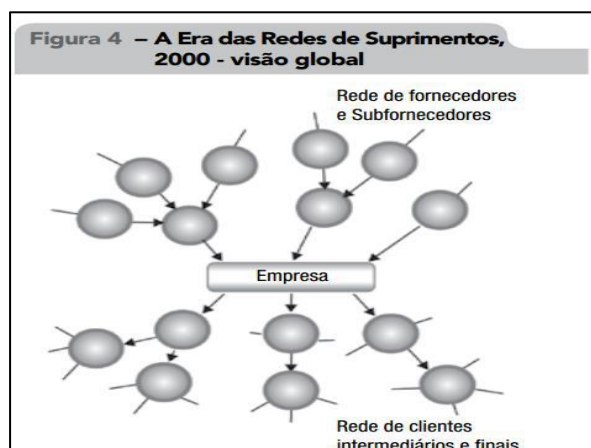


Fonte:

De modo geral, Francischini e Amaral Gurgel (2002, p. 262) alegam que a cadeia de suprimento é uma “integração dos processos que formam um determinado negócio, desde os fornecedores originais até o usuário final, proporcionando produtos, serviços e informações que agregam valor para o cliente”.

Machline (2011) denominou a Era das Redes de Suprimentos desde o ano 2000 até a atualidade, com uma visão global como mostra na Figura 4.

**Figura 4. Era das redes de Suprimentos**



Fonte:

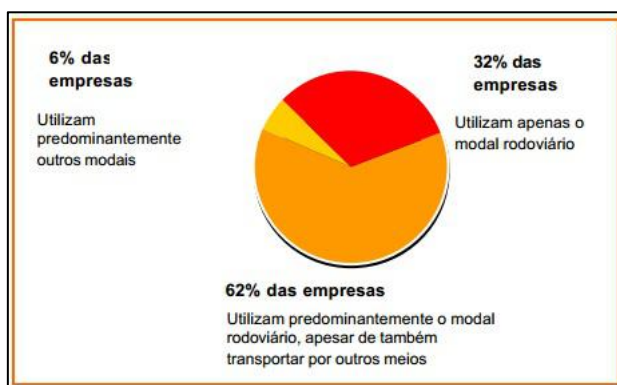
Chopra e Meindl (2011, p. 154), afirmam que devido ao atual cenário globalizado, as cadeias de suprimentos estão sujeitas a muito mais fatores de risco, do que as cadeias locais do passado. Subestimar os fatores de risco pode acarretar péssimos resultados como aconteceu na crise financeira de 2008, quando devido a grande recessão, o número de exportações e importações caiu drasticamente em diversos países, devido à diminuição do consumo.

Uma área que conecta os diversos estágios da cadeia de suprimentos, permitindo a coordenação das atividades e contribuindo para a mesma, é a Tecnologia da informação (TI), pois com softwares para gestão adequados, informações gerenciais e dados estatísticos estão disponíveis em tempo real para serem analisados e compartilhados dentro das empresas, facilitando todo o planejamento logístico (CHOPRA; MEINDL, 2003).

Sem os aplicativos de TI, a troca de informações seria limitada ao papel (NAZÁRIO, 1999), gerando um grande descontrole e prejudicando qualquer tipo de operação ou procedimento. As aplicações de TI na Logística são várias, e englobam tanto os equipamentos como os sistemas de informações. Combinados, o uso destas tecnologias permite o gerenciamento integrado e eficiente dos estoques, armazéns, transportes, processamento de pedidos, compras e manufaturas (FLEURY et al, 2000).

De todos os meios de transporte Logístico, ou modais Logísticos o mais utilizado no Brasil é o Rodoviário (RIBEIRO; FERREIRA, 2002). Em 2006 uma pesquisa do CEL (Centro de Estudos em Logística) /Copead (Instituto de Pós-Graduação e Pesquisa em Administração) apontou que 88,3% das empresas transportam cargas por rodovia, aproximadamente um terço das empresas entrevistadas usam somente o modal rodoviário e apenas 6% das empresas não utilizam modal rodoviário conforme apresentado na Figura 5:

**Figura 5. Participação do Modal Rodoviário nas Empresas**



Fonte: Panorama logístico CEL/COOPEAD – Gestão do transporte rodoviário de carga nas empresas – Práticas e Tendências - 2007

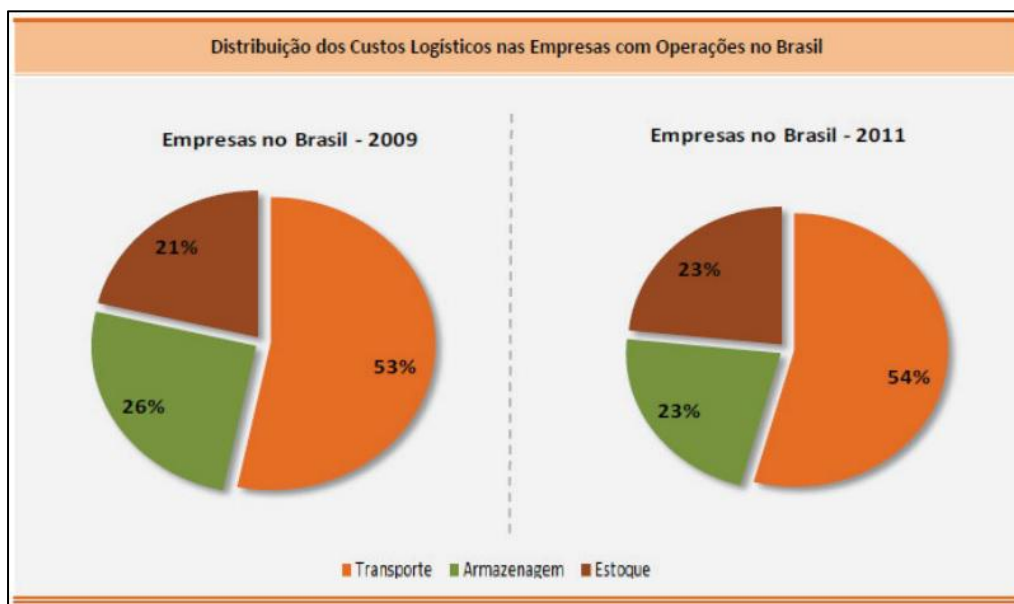
Os pontos fortes do modal rodoviário, tais como disponibilidade para embarques urgentes, rapidez em curtas distâncias, capacidade de atingir grande extensão no território brasileiro, custos fixos baixos, dentre outros, acabam por fazer dele o mais utilizado (DEMARIA; MARJORY, 2004).

Atualmente o país conta com um programa de Investimento em Logística que visa criar uma infraestrutura intermodal mais integrada, promover concessões em rodovias a fim de proporcionar uma melhor infraestrutura para assim aumentar a agilidade e diminuir custos. Cerca de R\$ 198,4 bilhões serão investidos no setor. Esses investimentos em infraestrutura se fazem necessários considerando que o mau estado das rodovias provoca uma média de 46% de aumento no custo operacional dos veículos (ROSA, 2007).

Em pesquisa realizada pelo CEL/Coopead cerca de 7,5% da receita líquida das empresas brasileiras é gasta com custos logísticos, englobando gastos com armazenagem, transporte e estoque. Em seu Livro Balou (2006) mostra que o transporte Logístico representa entre um ou dois terços dos custos totais de Logística das empresas.

Em estudo desenvolvido entre 2009 e 2011 pela empresa ILOS (Instituto de Logística e Supply Chain) o percentual de custos com logística derivados do transporte ocorreram na proporção apresentada na Figura 6:

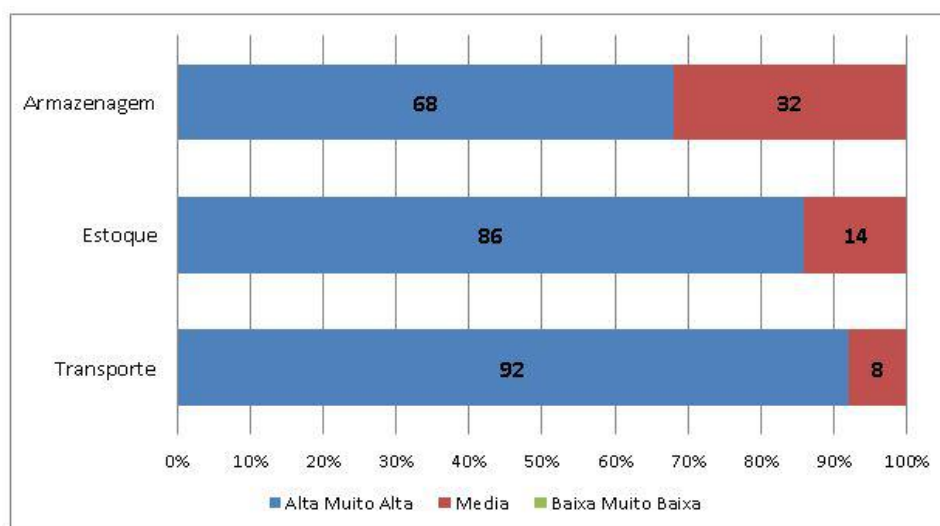
**Figura 6. Distribuição dos Custos Logísticos nas Empresas**



Fonte: Panorama Instituto ILOS - Custos Logísticos no Brasil, 2012

Devido ao grande percentual sobre os custos logísticos, as empresas buscam cada vez mais otimizar suas rotas de transporte. Na pesquisa realizada pelo CEL/Coopead cerca de 92% das empresas responderem que tem alta prioridade em buscar redução de custos com transporte dentro dos custos logísticos, conforme demonstrado na Figura 7:

**Figura 7. Grau de Priorização das Empresas na Redução de Custos logísticos**



Fonte: Adaptado de Panorama logístico CEL/COOPEAD – Gestão do transporte rodoviário de carga nas empresas – Práticas e Tendências, 2007.

Em estudo realizado por Matos Junior (2013), a roteirização foi capaz de reduzir as devoluções em média 1,57%, o que consequentemente reduz gastos operacionais dos veículos. Aumentou a taxa de ocupação em cerca de 7%, reduzindo o custo por quilo e na entrega conforme demonstrado na Tabela 1:.

**Tabela 1. Comparativo de produtividade dos veículos com e sem a roteirização**

Id da Rota	Distância	Custo Quilo	Custo por Parada	Custo Total
MYC - 8306	40,8	R\$ 0,14	R\$ 23,88	R\$ 191,00
MYC - 8306	72,7	R\$ 0,16	R\$ 27,38	R\$ 219,00
<b>Redução</b>	<b>43,88%</b>	<b>12,50%</b>	<b>12,78%</b>	<b>12,79%</b>
MYC - 8356	83,4	R\$ 0,15	R\$ 26,63	R\$ 229,00
MYC - 8356	100,3	R\$ 0,16	R\$ 30,50	R\$ 244,00
<b>Redução</b>	<b>16,85%</b>	<b>6,25%</b>	<b>12,69%</b>	<b>6,15%</b>

Fonte: Adaptado de Dados cedidos pela indústria cearense (2013)

No mesmo estudo de Matos Junior (2013), foi evidenciado um aumento médio de 7.32% nas entregas realizadas com sucesso, gerando uma melhoria no desempenho operacional, conforme mostrado a seguir na Tabela 2.

**Tabela 2. Atendimentos no Prazo de um Determinado Período Com e Sem a Roteirização**

Unidade	Atendimentos no Prazo com roteirização em %				Atendimentos no Prazo sem roteirização em %				Diferença em %			
	Janeiro	Fevereiro	Março	Abril	Janeiro	Fevereiro	Março	Abril	Janeiro	Fevereiro	Março	Abril
EUSÉBIO	92,20%	95,93%	94,36%	98,57%	86,40%	88,60%	87,60%	91,60%	5,80%	7,33%	6,76%	6,97%
BHO	98,65%	98,80%	98,46%	93,83%	91,08%	90,35%	92,41%	85,40%	7,57%	8,45%	6,05%	8,43%
RECIFE	99,21%	96,31%	93,48%	90,84%	92,60%	88,48%	86,60%	83,58%	6,61%	7,83%	6,88%	7,26%
TERESINA	91,47%	95,55%	95,11%	86,67%	78,90%	87,50%	88,64%	79,64%	12,57%	8,05%	6,47%	7,03%
RIO	95,27%	89,35%	86,12%	89,96%	87,25%	83,58%	79,61%	82,50%	8,02%	5,77%	6,51%	7,46%
TIMOM	100,00%	100,00%	100,00%	87,59%	95,00%	93,68%	91,80%	79,50%	5,00%	6,32%	8,40%	8,09%

Fonte: Adaptado de Dados cedidos pela indústria cearense (2013)

Após a apresentação desses dados, percebe-se que uma roteirização bem preparada gera economia para as empresas, melhorando o serviço de transporte entregue aos seus clientes.

### **1.1. Problema em estudo**

Em estudo realizado pela Empresa ILOS entre os anos de 2009 e 2011, cerca 54 % dos custos logísticos das empresas estudadas, são derivados do transporte. Já o Panorama Logístico(CEL/COOPEAD) mostra que 92% das empresas classificam a redução de custos com transporte com prioridade alta. Ao analisar esses dois fatores, faz-se necessário uma ferramenta que proporcione a redução de custos logísticos com transporte.

### **1.2. Relevância do Trabalho**

Uma plataforma de roteirização é importante pois, como apresentado pelo estudo de Matos Junior, uma roteirização adequada, reduz até 43% nas distâncias percorridas, além diminuir o custo por quilo em 7,3%.

Existem atualmente no mercado empresas que comercializam softwares para roteirização, no entanto grande parte delas faz necessária a utilização de uma plataforma muito mais ampla, incluindo mais módulos não dedicados à roteirização.

O desenvolvimento de um *software* de roteirização independente, auxiliará empresas de transporte proporcionando uma rota de entregas otimizada, não sendo necessário que as empresas tenham que adquirir uma nova plataforma de gestão completa.

### **1.3. Objetivo do Geral**

O objetivo geral deste trabalho é desenvolver uma aplicação, dedicada a criação de rotas, para o transporte de produtos entre centros de distribuição e também transporte de centro de distribuição para cliente final.

### **1.4. Objetivos Específicos.**

Este trabalho tem como objetivos específicos:

- Desenvolver um algoritmo para roteirização.
- Realizar um estudo comparativo entre aplicar uma roteirização manual, por ordem de inserção dos pontos de entrega, e realizar a roteirização utilizando algoritmo desenvolvido.

### 1.5. Proposta Metodológica

Para desenvolver o projeto, primeiramente foi realizado um trabalho de pesquisa com caráter exploratório, para compreender melhor o setor de logística, seu funcionamento, o quanto ele é vital para as empresas, o custo que ele gera para as empresas, e o interesse que elas têm em economizar com este setor. Realizado essa pesquisa, foi observado que desenvolvimento de um Software para roteirização de entregas poderia auxiliar este setor nas empresas reduzindo seus custos de operação.

Na pesquisa exploratória foi lido diversos artigos sobre roteirização, trechos de livros, de autores como: Paulo F. Fleury, Sunil Chopra, Peter Meindl. Autores estes sempre utilizados como referência, em estudos do Setor Logístico. Outro aspecto analisado, foi de artigos e estudos que comprovam o interesse das empresas em reduzir seus custos com operações logísticas como o Panorama Logístico Desenvolvido pelo CEL/COOPEAD.

No que compreende ao levantamento de requisitos, utilizar-se-á de *User Stories*, pois seu formato auxilia a codificação, proporcionando de maneira simples e direta a funcionalidade a ser desenvolvida.

Como metodologia de desenvolvimento, utilizar-se-á de Kanban, essa metodologia pode ser aplicada ao desenvolvimento de Softwares e combina muito bem com as *User Stories*. O Kanban proporciona uma visão geral do projeto e viabiliza a classificação das tarefas, organizando o andamento do projeto. O Trello foi escolhido como ferramenta de Kanban para o Projeto.

O desenvolvimento realizar-se-á por meio de duas etapas. Na primeira, desenvolver-se-á o *BackEnd* do projeto, que compreende toda a camada que disponibiliza os serviços para gestão dos cadastros, criação de rotas e consulta de ceps. A segunda etapa responsabilizar-se-á pelo desenvolvimento do *FrontEnd* do projeto, que é a *Interface Web* e também *Mobile*, na qual o usuário utilizará a aplicação.

Os resultados serão mensurados de maneira quantitativa analisando o tempo que uma empresa economizará utilizando o aplicativo para definir uma rota de entregas em relação a montar uma roteirização sem nenhum tipo de planejamento.



## 1.6. Conteúdo do Trabalho

O presente trabalho está estruturado em seis Capítulos, cujo conteúdo é sucintamente apresentado a seguir:

Capítulo 1, é o capítulo atual, composto pela Introdução, Objetivos e metodologia Aplicada.

No Capítulo 2 é apresentada etapa de engenharia de requisitos, apresentando as *User Stories* e seu detalhamento. Neste mesmo capítulo, as tecnologias utilizadas no desenvolvimento do projeto serão elencadas .

O Capítulo 3 compreende o Desenvolvimento do Trabalho, sendo composto por: Arquitetura da Solução, Modelagem e Gestão dos Dados, Arquitetura do Software, Segurança e para concluir a apresentação de uma Visão Geral do Sistema.

No Capítulo 4 serão apresentadas as experimentações e a análise dos resultados obtidos.

Finalmente, o Capítulo 5 apresenta a conclusão deste trabalho com base nos resultados obtidos com o software desenvolvido.

## 2. REQUISITOS IDENTIFICADOS E CONTEXTUALIZAÇÃO TECNOLÓGICA

O presente capítulo dedica-se a Engenharia de Requisitos, ramo da Engenharia de Software, que segundo Thayer e Dorfman(2004) é responsável por analisar e documentar requisitos, incluindo análise das necessidades do software e a especificação dos requisitos.

### 2.1. Especificação de requisitos

Requisitos podem ser definidos como o que o sistema oferece, o que o sistema deve fazer, suas entradas e restrições. Os requisitos são um reflexo da necessidade do cliente para um sistema com função determinada, exemplo: buscar informações, armazenar um dado, controlar um equipamento(SOMMERVILLE, 2011). O processo de descoberta e documentação destes requisitos é denominada Levantamento de Requisitos.

Os requisitos podem ser classificados em dois tipos:

- i. Requisitos Funcionais: Descrevem as funcionalidades do sistema, suas ações, entradas , saídas e exceções
- ii. Requisitos Não-Funcionais: São requisitos que não estão diretamente ligados com serviços e/ou funcionalidades específicas oferecidas pelo *Software*. Eles são frequentemente mais críticas como: Definir restrições sobre a implantação do sistema, confiabilidade, tempo de resposta dentro outros aspectos.

### 2.1.1. Requisitos Funcionais:

Os requisitos funcionais do projetos foram analisados, elencados e categorizados conforme mostrado na Tabela 3:

**Tabela 3. Requisitos Funcionais do Projeto**

Requisitos Funcionais		
Requisito	Nível de Priorização	Legenda
Consultar CEP	10	Imprescindível
Gerar Rota a Partir de Lista de CEPs	10	Imprescindível
Apresentar Rota Gerada com o Google Maps	10	Imprescindível
Gerenciar Usuário	10	Imprescindível
Gerenciar Pessoa	8	Importante
Gerenciar Empresa	8	Importante
Gerenciar Região	9	Obrigatório
Gerenciar Filiais	8	Importante
Gerenciar Cargos	4	Desejável
Gerenciar Funcionários	7	Importante
Disponibilizar <i>Web Service</i> de geração de Rotas	8	Importante

Fonte: O autor

### 2.1.2. Requisitos Não-Funcionais:

Os requisitos não funcionais elencados serão mostrados a seguir na Tabela 4:

**Tabela 4. Requisitos Não-Funcionais do Projeto**

Requisitos Não-Funcionais		
Requisito	Nível de Priorização	Legenda
Consultar Cep não Cadastrado externamente	8	Importante
Utilizar autenticação básica para a Aplicação	10	Imprescindível
Desenvolver para Plataforma <i>Web</i> e Android	8	Importante
Desenvolver para plataforma IOS	2	Baixa
Padrão de comunicação <i>BackEnd</i> - <i>FrontEnd</i> via Http	10	Imprescindível
JSON como formato do arquivo de comunicação	10	Imprescindível
Comunicação constante com servidor de Internet	10	Imprescindível

Fonte: O autor

## 2.2. Especificações baseadas em User Stories

O formato escolhido para a especificação dos requisitos é o de *User Stories*, por conta de ser uma forma sucinta e direta de apresentar a funcionalidade à ser desenvolvida, os critérios de aceitação e os fluxos de exceções. Esse formato prioriza o processo de desenvolvimento do código e entrega do software.

Para que uma *User Story* possa ser descrita, primeiramente devem ser elencadas as Personas, ferramenta que utiliza estereótipos de grupos de pessoas fictícias para representar usuários. As Personas envolvidas no projeto estão relacionadas na Tabela 5, apresentada abaixo:

**Tabela 5. Lista de Personas com seus comportamentos, necessidades e objetivos**

Personas		
Persona	Comportamentos	Necessidade/Objetivos
Motorista	<ul style="list-style-type: none"> <li>- Não utiliza Sistemas Gerências</li> <li>- Não é familiarizado com computadores</li> <li>- Usa <i>Smartphone</i></li> </ul>	<ul style="list-style-type: none"> <li>- Que as rotas já estejam otimizadas quando ele sair para realizar as entregas</li> <li>- Ferramenta simples e fácil</li> </ul>
Gerente	<ul style="list-style-type: none"> <li>- Usa Sistemas Gerenciais apenas para consulta</li> <li>- Dificilmente realiza algum procedimento operacional</li> <li>- Está interessado em resultados</li> </ul>	<ul style="list-style-type: none"> <li>- Redução de custos com transporte</li> <li>- Aumentar a quantidade de entregas no Prazo</li> <li>- Não precisar investir em um novo Sistema de Gestão</li> </ul>
Operador Logístico (Engloba Analistas, técnicos auxiliares) e	<ul style="list-style-type: none"> <li>- Realiza muitos procedimentos operacionais ao longo do dia no Sistema Gerencial.</li> <li>- São familiarizados com computadores e Sistemas</li> <li>- Tem que checar manualmente as entregas para verificar quais não são entregues pela sua respectiva empresa</li> </ul>	<ul style="list-style-type: none"> <li>- Sistema simples e Objetivo para realizar cadastros e alterações</li> <li>- Interface minimalista, com fácil navegação</li> <li>- Identificar automaticamente as entregas à serem transferidas a outros centros de distribuição</li> <li>- Otimizar a rota de entregas automaticamente</li> <li>- Não necessitar refazer os lançamentos que estão no Sistema de gestão</li> </ul>
TI	<ul style="list-style-type: none"> <li>- Alto conhecimento sobre sistemas e tecnologias</li> <li>- Preza pela facilidade, sendo contra o retrabalho</li> <li>- É o responsável pelo cadastro de usuários</li> </ul>	<ul style="list-style-type: none"> <li>- Integração entre o sistema de otimização de rotas e o sistema de gestão</li> <li>- Gestão adequada dos usuários</li> </ul>

Fonte: O autor

Após definida as Personas envolvidas com o projeto, as *User Stories* podem ser apresentadas. A primeira *User Story* a ser descrita é relacionada a criação de uma Rota otimizada, sendo mostrada na tabela 6:

**Tabela 6. User Story - Otimização de Rota**

User Story	Otimização de Rota
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Gostaria que a melhor rota de entrega fosse gerada Automaticamente
<b>Por que?</b>	Para reduzir custos operacionais e realizar as entregas mais rapidamente
<b>Critérios de aceitação</b>	Gerar rotas lançando endereços manualmente
	Gerar rotas a partir de uma lista de endereços já pronta

Fonte: O autor

Na próxima *User Story*, apresentada pela Tabela 7 é apresentada a necessidade do motorista recuperar uma rota gerada pelos operadores logísticos:

**Tabela 7. User Story - Recuperar Rota**

User Story	Recuperar Rota
<b>Quem ?</b>	Motorista
<b>O que?</b>	Quero conseguir encontrar uma rota gerada pelos operadores
<b>Por que?</b>	Para poder visualizar a rota otimizada e realizar as entregas
<b>Critérios de aceitação</b>	Quero visualizar as rotas que foram geradas e que ainda não tenham sido percorridas

Fonte: O autor

A *User Story* a seguir, relacionada a Tabela 8 mostra o interesse por parte dos Operadores Logísticos em identificar as entregas à endereços que não são área de atuação de sua empresa.

**Tabela 8. User Story - Identificar Entregas Fora da Região de Distribuição da Empresa**

User Story	Identificar entregas fora da região de distribuição
<b>Quem ?</b>	Operadores logísticos
<b>O que?</b>	Preciso que seja mostrada as entregas que não correspondem a área de atuação da empresa
<b>Por que?</b>	Para que as entregas sejam devidamente encaminhadas aos seus centros de distribuição respectivos.
<b>Critérios de aceitação</b>	A partir de uma Lista de Entregas identificar as entregas que não são realizadas pela empresa
	Apresentar a empresa que atende o endereço

Fonte: O autor

A *User Story* retratada na Tabela 9 apresenta a necessidade de utilização do Serviço de roteirização a partir de um outro sistema:

**Tabela 9. User Story - Solicitar Geração de Rotas a partir de Outro Sistema**

User Story	Solicitar geração de rotas a partir de outro sistema
<b>Quem ?</b>	Operadores logísticos
<b>O que?</b>	Gostaríamos de Gerar a rota a partir do Sistema e Gestão Logística Utilizado
<b>Por que?</b>	Para que não precisemos ficar utilizando outro sistema além do nosso sistema de gestão
<b>Critérios de aceitação</b>	A partir de um outro Sistema que não a Interface do Projeto, solicitar a criação da rota

Fonte: O autor

A Tabela 10 apresenta a *User Story* relacionada a necessidade de excluir uma rota criada:

**Tabela 10. User Story - Excluir rota gerada**

User Story	Excluir rota gerada
<b>Quem ?</b>	Operadores logísticos
<b>O que?</b>	Gostaríamos excluir rotas que foram geradas
<b>Por que?</b>	Para que rotas geradas por engano, ou com algum endereço errado não sejam mais visualizadas
<b>Critérios de aceitação</b>	A partir da lista de rotas geradas, poder selecionar uma e exclui-la desde que ja não esteja sendo utilizada

Fonte: O autor

Na Tabela 11 é apresentada a *User Story* onde novos usuários poderão ser cadastrados:

**Tabela 11. User Story - Cadastrar Usuário**

User Story	Cadastrar Usuário
<b>Quem ?</b>	TI
<b>O que?</b>	Precisamos cadastrar usuários
<b>Por que?</b>	Para que mais pessoas possam utilizar a aplicação
<b>Critérios de aceitação</b>	Passando um email e senha, seja cadastrado um usuário para utilizar a aplicação

Fonte: O autor

Na Tabela 12 é mostrada a *User Story* relacionada a necessidade de alteração de usuário.

**Tabela 12. User Story - Alterar Usuário**

User Story	Alterar Usuário
<b>Quem ?</b>	TI
<b>O que?</b>	Precisamos Alterar usuários
<b>Por que?</b>	Para ajustar suas permissões quando necessário
<b>Critérios de aceitação</b>	Ao pesquisar um usuário e seleciona-lo, o Sistema permita que eu altere suas permissões

Fonte: O autor

A Tabela 13 Apresenta a necessidade de Listar Consultar Usuários cadastrados.

**Tabela 13. User Story - Pesquisar Usuários**

User Story	Pesquisar Usuário
<b>Quem ?</b>	TI
<b>O que?</b>	Precisamos Pesquisar usuários
<b>Por que?</b>	Para poder visualizar os usuários e saber quem está usando o Sistema
<b>Critérios de aceitação</b>	Ao clicar em pesquisar, trazer todos os usuários
	Ao clicar no email, retornar o usuário

Fonte: O autor

A Tabela 14 Apresenta a necessidade de Excluir Usuários cadastrados caso seja necessário.

**Tabela 14 - User Story - Deletar Usuário**

User Story	Excluir Usuário
<b>Quem ?</b>	TI
<b>O que?</b>	Precisamos Excluir usuários
<b>Por que?</b>	Para que usuários que não estejam mais na empresa não utilizem o sistema
<b>Critérios de aceitação</b>	Ao pesquisar um usuário e seleciona-lo, o Sistema permita que eu o exclua

Fonte: O autor

A Tabela 15 Apresenta a necessidade de consultar CEP.

**Tabela 15. User Story - Consultar Cep**

User Story	Consultar Cep
<b>Quem ?</b>	Motorista
<b>O que?</b>	Precisamos consultar cep e que mostrando a localização dele
<b>Por que?</b>	Para que possamos saber onde se localiza o endereço com esse CEP
<b>Critérios de aceitação</b>	Ao inserir um cep o sistema traga um mapa com a localização dele

Fonte: O autor

A Tabela 15 Apresenta a necessidade de consultar CEP.

**Tabela 16. User Story - Abrir Rota no Maps**

User Story	Abrir Rota no Google Maps
<b>Quem ?</b>	Motorista
<b>O que?</b>	Preciso abrir a rota no gerada no google maps
<b>Por que?</b>	Para poder utilizar a rota com o GPS
<b>Critérios de aceitação</b>	Selecionar uma rota gerada, abrir o google maps com o itinerário.

Fonte: O autor

A Tabela 17 Apresenta a necessidade de consultar CEPs.

**Tabela 17. User Story - Consultar Ceps**

User Story	Consultar Cep
<b>Quem ?</b>	Operadores de Logística
<b>O que?</b>	Precisamos consultar ceps
<b>Por que?</b>	Para podermos saber a lista de ceps com base na busca para poder criar regiões e analisar demanda
<b>Critérios de aceitação</b>	Ao clicar em buscar cep ele traga todos os ceps cadastrados
	Ao inserir o nome de uma rua, o sistema deve trazer todas as ocorrências de ceps com o nome de rua inserido
	Ao inserir o nome de uma cidade e a sigla do estado, traga todos os ceps cadastrados para a cidade
	Ao inserir o nome de uma cidade e um bairro traga todos os Ceps cadastrados para a cidade

A Tabela 18 Apresenta a necessidade de cadastrar Pessoa.

**Tabela 18. User Story - Cadastrar Pessoa**

User Story	Cadastrar Pessoa
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos cadastrar uma usuários
<b>Por que?</b>	Para ter a informação da pessoa que utiliza o sistema e não apenas o email
<b>Critérios de aceitação</b>	Passando as informações de Pessoa Física ou Pessoa Jurídica, o sistema deve cadastra-las e vincula-las ao usuário

Fonte: O autor

A Tabela 19 Apresenta a necessidade de alterar uma Pessoa Cadastrada.

**Tabela 19. User Story - Alterar Pessoa**

User Story	Alterar Pessoa
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Alterar cadastro de pessoa
<b>Por que?</b>	Para corrigir possíveis erros de cadastro
<b>Critérios de aceitação</b>	Após pesquisar uma pessoa, o sistema deve permitir que eu altere seu cadastro

Fonte: O autor

A Tabela 20 Apresenta a necessidade de Pesquisar Pessoas cadastrados.



**Tabela 20. User Story - Pesquisar Pessoa**

User Story	Pesquisar Pessoa
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Pesquisar Pessoas cadastradas
<b>Por que?</b>	Para Verificar as informações pessoas que possuem cadastro
<b>Critérios de aceitação</b>	Ao clicar em buscar, buscar todos
	Ao inserir um tipo, buscar todas as pessoas do Tipo
	Ao inserir um CPF/CNPJ retornar a pessoas respectiva
	Ao inserir razão social, buscar uma lista de pessoas com a razão social buscada

Fonte: O autor

A Tabela 21 Apresenta a necessidade de Cadastrar uma Empresa.

**Tabela 21. User Story - Cadastrar Empresa**

User Story	Cadastrar Empresa
<b>Quem ?</b>	Gerente
<b>O que?</b>	Preciso cadastrar minha empresa
<b>Por que?</b>	Para que meus funcionários possam utilizar o Sistema e estarem vinculados a minha empresa
<b>Critérios de aceitação</b>	Após inserida as informações da empresa, o Sistema salva o cadastro

Fonte: O autor

A Tabela 22 Apresenta a necessidade de alterar uma Pessoa Cadastrada.

**Tabela 22. User Story - Alterar Empresa**

User Story	Alterar Empresa
<b>Quem ?</b>	Gerente
<b>O que?</b>	Precisamos Alterar cadastro de Empresa
<b>Por que?</b>	Caso alguma informação da empresa mude
<b>Critérios de aceitação</b>	Após pesquisar retornar a empresa, abrir a tela para poder altera-la.

Fonte: O autor

A Tabela 23 Apresenta a necessidade de alterar uma Pessoa Cadastrada.

**Tabela 23. User Story - Pesquisar Empresa**

User Story	Pesquisar Empresa
<b>Quem ?</b>	Gerente
<b>O que?</b>	Precisamos Pesquisar Empresas
<b>Por que?</b>	Verificar as filiais e os dados cadastrados
<b>Critérios de aceitação</b>	Após clicar em buscar, trazer todas as empresas filiais relacionadas
	Após selecionar transportadoras, trazer as empresas transportadoras

Fonte: O autor

A Tabela 24 Apresenta a necessidade de alterar uma Pessoa Cadastrada.

**Tabela 24. User Story - Cadastrar Funcionário**

User Story	Pesquisar Funcionários
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Cadastrar Funcionários
<b>Por que?</b>	Para que elas estejam vinculados a empresa e consigam utilizar o Sistemas
<b>Critérios de aceitação</b>	Após Inserir as informações do funcionário e clicar em salvar, o sistema deve cadastrar o funcionário

Fonte: O autor

A Tabela 25 Apresenta a necessidade de alterar um funcionário cadastrado.

**Tabela 25. User Story - Alterar Funcionário**

User Story	Pesquisar Funcionários
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Alterar o Cadastros dos Funcionários
<b>Por que?</b>	Por que podem haver mudanças, como cargo, ou mesmo de filial da empresa
<b>Critérios de aceitação</b>	Após pesquisar um funcionário, abrir uma tela para alterar as informações, realizando a alteração dos dados o sistema deve salvar essa alteração.

Fonte: O autor

A Tabela 26 Apresenta a necessidade de deletar um funcionário cadastrado.

**Tabela 26. User Story - Deletar Funcionário**

User Story	Pesquisar Funcionários
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Deletar um Funcionário
<b>Por que?</b>	Por que o funcionário pode sair da empresa
<b>Critérios de aceitação</b>	Após pesquisar um funcionário, seleciona-lo e exclui-lo

Fonte: O autor

Tabela 27 Apresenta a necessidade de pesquisar um funcionário cadastrado.

**Tabela 27. User Story - Pesquisar Funcionário**

User Story	Pesquisar Funcionários
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Pesquisar um Funcionário
<b>Por que?</b>	Para poder checar o quadro de funcionários cadastrados no Sistema
<b>Critérios de aceitação</b>	Ao inserir dados da pessoa, buscar o funcionário respectivo
	Ao inserir uma empresa, buscar os funcionários cadastrados na nela

Fonte: O autor

Tabela 28 Apresenta a necessidade de pesquisar uma região cadastrada.

**Tabela 28. User Story - Pesquisar Região**

User Story	Pesquisar Região
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Pesquisar região cadastrada
<b>Por que?</b>	Para poder ver os ceps que compõem a região
<b>Critérios de aceitação</b>	Ao inserir a empresa, buscar as regiões dela
	Ao inserir uma empresa e a descrição da rota, retornar as regiões que satisfaçam as restrições de pesquisa
	Ao inserir uma empresa matriz, trazer todas as regiões da matriz e de suas filiais.

Fonte: O autor

Tabela 29 Apresenta a necessidade alterar uma região cadastrada.

**Tabela 29. User Story - Alterar Região**

User Story	Alterar Região
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Alterar uma região cadastrada
<b>Por que?</b>	Pois os ceps que a compõem podem mudar.
<b>Critérios de aceitação</b>	Ao buscar a região de uma empresa e selecionar par alterar, abrir a tela de alteração e salvar as alterações

Fonte: O autor

Tabela 30 Apresenta a necessidade de deletar uma região cadastrada.

**Tabela 30. User Story - Deletar Região**

User Story	Deletar Região
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos Deletar uma região cadastrada
<b>Por que?</b>	Região pode entrar em desuso
<b>Critérios de aceitação</b>	Ao buscar a região de uma empresa e selecionar par deletar, Sistema deve deletar essa região

Fonte: O autor

Tabela 31 Apresenta a necessidade de pesquisar um funcionário cadastrado.

**Tabela 31. User Story - Efetuar Login**

User Story	Efetuar Login
<b>Quem ?</b>	Todos
<b>O que?</b>	Precisamos Deletar uma região cadastrada
<b>Por que?</b>	Necessário para que ninguém utilize o sistema sem estar identificado
<b>Critérios de aceitação</b>	Entrar na tela principal inserir email e senha para conectar ao sistema.

Fonte: O autor

Tabela 32 Apresenta a necessidade de pesquisar um funcionário cadastrado.

**Tabela 32. User Story - Efetuar Login**

User Story	Cadastrar Região
<b>Quem ?</b>	Operador Logístico
<b>O que?</b>	Precisamos cadastrar uma região
<b>Por que?</b>	Necessário para poder parametrizar as indicações de área de atuação
<b>Critérios de aceitação</b>	Ao inserir Bairro e Cidade, cadastrar todos os ceps do bairro e cidade passados
	Ao inserir uma Cidade e um Estado, cadastrar todos os ceps da cidade e estado respectivos
	Ao passar uma lista de ceps, efetuar o cadastro de uma região com essa lista

Fonte: O autor

Tabela 31 Apresenta a necessidade de pesquisar um funcionário cadastrado.

### 2.2.1. BackLog

Todas as tarefas elencadas foram organizadas com o Trello, ferramenta para Kanban escolhido para o Task Control do projeto:

Segue os links do Trello:

*FrontEnd:* <https://trello.com/b/AMaFiirR/frontend-sysrlog-trabalho-de-gradua%C3%A7%C3%A3o>

*BackEnd:* <https://trello.com/b/Ot6gDOSy/backend-sysrlog-trabalho-de-gradua%C3%A7%C3%A3o>

## 2.3. Tecnologias Aplicadas

Para atender as *User Stories* elencadas anteriormente, faz-se necessário a seleção de tecnologias à serem utilizadas para o desenvolvimento do *Software*. As tecnologias definidas foram divididas em duas categorias: *BackEnd* e *FrontEnd*.

O *BackEnd* é a camada do servidor que recebe as requisições, faz acesso ao banco de dados e envia as respostas. Também é a responsável pelas regras de negócio e por prover segurança a aplicação(MARQUES, 2017).

O *FrontEnd* é camada correspondente a interação com o usuário, é a camada que irá receber a entrada de dados, enviar ao *BackEnd*, receber a resposta e apresentar ao usuário(MARQUES, 2017).

### 2.3.1. BackEnd

A seguir serão apresentadas, as tecnologias aplicadas no desenvolvimento do BackEnd do software.

#### 2.3.1.1. Linguagem

A linguagem de programação escolhida foi a Java. Ela foi desenvolvida na década de 90 pela Sun Microsystems. No ano de 2008 a empresa foi adquirida pela Oracle. O diferencial da Linguagem Java frente as linguagens convencionais, é que ao invés de um programa ser compilado para o código nativo da plataforma, na qual o programa foi desenvolvido, ele é compilado para um bytecode(código da linguagem Java) que será interpretada pela JVM Java Virtual Machine ou Máquina virtual do Java.

A JVM é a responsável por fazer com que um código Java possa ser executado em diferentes plataformas que utilizam diferentes Sistemas Operacionais.

Java foi escolhida por conta de sua portabilidade e Frameworks para desenvolvimento, além de sua grande utilização. Segundo dados disponíveis no site da linguagem, 97% dos Desktops corporativos executam Java e cerca 89% dos computadores residências possuem Java instalado. Segundo (UDACITY,2018) Java aparece sempre nas primeiras posições, como linguagem mais utilizada no mercado.

### **2.3.1.2. Formato para Transmissão de Dados**

Utilizar-se-á do formato JSON (Java Script Object Notation) para a transmissão de dados no Projeto. No site oficial JSON é descrito como um formato leve para troca de dados, fácil para ser escrito e lido por humanos e fácil para serem convertidos e gerados por máquinas.

O formato foi escolhido por conta de além de ser o formato padrão utilizado pelo Spring Framework, pela API de distâncias do Google e pela API de consultas de ceps do viaCep, é também o formato utilizado pelos componentes que fazem parte do FrontEnd.

### **2.3.1.3. Maven**

Maven é uma ferramenta desenvolvida para facilitar o processo de desenvolvimento, Build (construir um programa executável a partir de código fonte) e gerenciar qualquer projeto baseado em Java.

Ele foi originalmente iniciado para facilitar o processo de Build , de um Projeto chamado Jakarta Turbine. Este projeto é composto por diversos subprojetos, cada um com suas próprias Builds e JARS (Arquivos Java).

A ferramenta baseou-se na utilização de Project Object Model (POM), um arquivo formato XML que contem todas as informações necessárias para a Build do projeto, configurações, formatos de compilação e o a gestão das dependências.

O Maven irá realizar a gestão de todas as dependências da parte de BackLog do projeto que serão listadas a seguir:

### **2.3.1.4. Spring**

O principal Framework e base para o BackEnd é o Spring. Ele foi lançado no Ano de 2002, hoje é mantido pela Pivotal Software. Suas principais características são a Inversão de Controle e a Injeção de dependências.

Inversão de controle é quando as chamadas da aplicação não podem ser controladas manualmente ou não possuem ordem definida para execução. Injeção de Dependências é um padrão de desenvolvimento que busca manter um nível de acoplamento baixo entre os módulos das aplicações, nesse caso o Container ou Framework que disponibiliza ou ‘injeta’ os componentes entre os módulos sempre que necessário (FOWLER, 2004).

O Spring é composto também de outros módulos, denominados no site como Projects . Para o desenvolvimento utilizar-se-á dos seguintes Projects:

- I. Spring ou Spring MVC: é um dos projetos iniciais do Spring com foco da Inversão de Controle e Injeção de dependências. Esse projeto provê as dependências necessárias para a criação de Serviços, Views e Controllers.
- II. Spring Boot DevTools - Tem a função e prover toda a configuração necessária para executar uma aplicação baseada Spring de Maneira automática, economizando tempo e trabalho.
- III. Spring Data JPA - Torna a comunicação entre a aplicação e o banco de dados mais simples, a partir de parametrizações é possível definir Classes que são a representam determinada tabela do banco de dados, e definir classes que executam funções de consulta, alteração, criação e exclusão a essas tabelas denominados repositórios. Outra grande vantagem dos repositórios é que eles evitam a criação de código repetitivo.
- IV. Spring Security - Toda a parte de autenticação e segurança é de responsabilidade deste Projeto. Ele é fornece configurações classes e métodos para serem utilizados de forma a controlar a segurança da aplicação. No entanto ele oferece autenticação básica, para complementar utilizar-se-á de de JSONWebToken o JWT que será enviado em todas as requisições realizadas.
- V. Spring Test - É responsável pela execução de testes Unitários utilizando o JUnit.

Informações detalhadas ou de outros Projects do Spring podem ser encontrados no seu próprio site, que também apresenta toda a documentação de cada uma dos Projects.

#### **2.3.1.5. Banco de Dados**

O Banco de Dados selecionado para o Projeto é o MySQL. É um banco de dados gratuito distribuído pela Oracle utilizados por diversas empresas ao redor do mundo. Algumas características que definiram o MySQL como Banco de Dados do projeto: são o excelente desempenho e escalabilidade, compatibilidade com a linguagem Java, baixo consumo de recursos do Host.

#### **2.3.1.6. Plugins para Base de Dados**

- a. H2Database: Banco de dados em memória, utilizar-se-á o H2 para Testes Unitários, verificação dos Scripts do Liquibase e para testes simples dos Serviços e Endpoints.

- b. MySQL Connertor: Plugin necessário para a conexão do Software em Java com o banco de Dados MySQL
- c. Liquibase: Biblioteca Java para de código fonte aberto para controle e gestão do Banco de Dados de uma aplicação a partir de Scripts Sql ou arquivos XML. O liquibase pode executar tanto comandos DDL Data Definition Language como DML Data Manipulation Language

#### **2.3.1.7. Demais Dependências utilizadas**

- a. Project Lombok: Biblioteca Java que gera métodos essenciais de forma automática, como métodos get, set, toString, equals, construtores sem atributos dentre outros.
- b. Javax.Json : Biblioteca utilizada para leitura, conversão e criação de arquivos JSON
- c. HttpClient : Cliente para realizar requisições HTTP para WebServices, faz-se necessária a utilização dessa biblioteca para realizar requisições à API de Distâncias do Google e ao Servidor do ViaCep.

#### **2.3.1.8. Softwares utilizados**

Spring Tool Suite: Conhecido como STS, é uma IDE customizada, que tem o Eclipse como base. É distribuída pela Pivotal Software, desenvolvedora do Projeto Spring. O STS disponibiliza um ambiente completo para criação e execução de uma aplicação baseada no Framework Spring. Contém inclusive um servidor Web embutido, que executa a aplicação desenvolvida automaticamente, sem necessidade de realizar nenhum tipo de configuração.

XAMPP: é um Software Livre Promovido pela Iniciativa Apache Friends. É composto por Servidos Apache, PHP, Perl e atualmente a distribuição Baseada em MySQL MariaDB. Esse software foi elencado para ser utilizado, pelo baixo consumo de recursos de Hardware, e fácil instalação e utilização de suas ferramentas. Para o o projeto a ferramenta da plataforma a ser utilizada é o MariaDB.



### **2.3.1.9. Recursos Externos**

Para satisfazer as necessidades dos usuários, faz-se necessária a utilização de dois recursos externos:

- I. Distance Matrix API: Fornecida pela Google a API disponibiliza o calculo de distancias e tempo de percurso, entre ponto de origem e ponto de destino. É utilizada por meio de requisições HTTP, tendo que enviar a URL utilizando o padrão apresentado na documentação fornecida pela Google. Retorna um JSON com as informações de distancia e tempo.
- II. ViaCep: É um Webservice gratuito para consulta de ceps. As requisições são feitas pelo Protocolo HTTP. Ao efetuar uma requisição o Webservice retorna um JSON com as informações do CEP, caso ele seja válido.

### **2.3.2. FrontEnd**

As tecnologias apresentadas a seguir fazem parte da camada de FrontEnd do projeto.

#### **2.3.2.1. NPM**

Npm é um Software para gerenciamento de Pacotes/Dependências para linguagem de programação JavaScript/TypeScript. Foi lançado inicialmente em 2010.

O NPM consiste em uma base de dados online com diversas dependências que podem ser baixadas pelo cliente NPM, utilizando linha de comando.

#### **2.3.2.2. Ionic**

Ionic é um SDK Software Development Kit foi lançado em 2013, sendo desenvolvido por Max Lynch, Ben Sperry e Adam Bradley. O Ionic possibilita o desenvolvimento de aplicativos híbridos ou multiplataforma, podendo ser executados em IOS, Android, Windows e Browser. O Ionic é executado uma camada acima do Cordova, ele prove a Interface ao usuário enquanto o Cordova age, transcrevendo as ações do FrontEnd em comandos para a plataforma no qual aplicação é executada.

Para prover o desenvolvimento multiplataforma o Ionic utiliza os recursos primordiais do desenvolvimento de FrontEnd no formato para Web. Para criação de aplicativos ele utiliza HTML5, CSS e JavaScript. Esses itens já são o suficiente para o desenvolvimento de uma Página Web, mas o Ionic vai mais além ele permite implementar também TypeScript e Angular.

Além das tecnologias já citadas, ele ainda permite a utilização de recursos nativos do dispositivo que o está executando como: câmera, GPS, acesso a arquivos locais dentre outros, isso é possível novamente por conta do Cordova. Outro recurso disponível são componentes já prontos desenvolvidos do Ionic que podem ser utilizados para enriquecer o FrontEnd e deixar o visual da aplicação mais elegante.

O Ionic foi selecionado para implementação do FrontEnd por conta da capacidade de ser executado em qualquer plataforma, podendo acessar os recursos nativos do dispositivo no qual é utilizado, sem a necessidade de fazer nenhum tipo de alteração do código fonte. Além de possuir componentes já preparados para serem utilizados.

#### **2.3.2.3. HTML5**

Hypertext Markup Language conhecido como HTML. É uma linguagem para estruturação e apresentação de conteúdo. É basicamente a Tecnologia chave da Internet na maneira que conhecemos, todos os sites utilizam o HTML.

Foi originalmente desenvolvido para descrever semanticamente, documentos científicos, mas devido ao seu design, ele pode ser adaptado para tornar-se a estrutura básica de um Site.

Sua nova Versão, a 5, foi lançada na íntegra no ano de 2014. Trazendo diversos novos recursos como: Funcionar Offline, armazenamento Local, viabilização de multimídia, novos efeitos, melhoria de desempenho e capacidade de acessar dispositivos.

#### **2.3.2.4. CSS**

CSS é a abreviação de Cascade Style Sheets que traduzindo significa, Folha e Estilos em Cascata. Ele é utilizado junto a com linguagens de marcação como HTML.

Sua principal função é determinar o visual de uma Página web, torna-la apresentável aos olhos de quem a utiliza. Ele é responsável por coordenar todos os estilos aplicados a Página Web e ‘dar vida’ a Página estática baseada em HTML.

### **2.3.2.5. TypeScript**

Lançada em 2012 pela Microsoft. O TypeScript é um superconjunto do JavaScript. Sua principal diferença com relação ao JavaScript é suportar o uso de Programação Orientada a Objeto. Com o TypeScript é possível escrever classes, interfaces, indicar o formato de retorno de métodos e indicar o tipo das variáveis.

O TypeScript consegue executar JavaScript, assim possibilitando a utilização de ferramentas já desenvolvidas. Na questão de execução, como apresentado no site da ferramenta, todo o código escrito em TypeScript é compilado para JavaScript podendo assim ser interpretado e executado no Browser.

### **2.3.2.6. AngularJS**

Angular foi lançado em 2016 e é mantido pela Google. Ele é uma Framework de código aberto, baseado em JavaScript. Tem a função de construir interfaces para uma aplicação web a partir da utilização de HTML, CSS e JavaScript.

O Angular funciona a partir da leitura de Páginas HTML que contenham atributos adicionais em suas Tags. Ele interpreta esses atributos como diretivas, para unir partes de entrada e saída da página, com uma variável. Facilitando a comunicação entre a Página e o controlador da página.

### **2.3.2.7. Cordova**

Cordova foi lançado em 2017. É um framework para desenvolvimento de aplicações mobile baseado em HTML, CSS e JavaScript. Ele também é capaz de acessar recursos nativos do ambiente no qual é executado.

### **2.3.2.8. Softwares utilizados**

VSCode: Lançado em 2015 e desenvolvido pela Microsoft, o VSCode é um editor de código fonte com distribuições para Windows, Linux e MacOS. Suas principais características são: Complemento inteligente de código (incluindo TypeScript, HTML, CSS e os componentes Ionic), Refatoração de código, suporte a Depuração, *Snippets* e controle de Git Incorporado.

Ionic Cli: *Ionic Command Line Interface* conhecido como Ionic CLI é uma ferramenta criada para o desenvolvimento de Aplicações Ionic. Com ele é possível criar uma aplicação Ionic, gerar automaticamente alguns componentes. O CLI também facilita os processos de compilação e execução além de prover processos de *Build* e *Live-Reload*.

### 2.3.3. Versionamento

Com relação ao versionamento do projeto, foram criado dois repositórios no GitHub, um para o *FrontEnd* e um para o *BackEnd*.

*FrontEnd*: <https://github.com/JoaoVFG/sysrlogapp>

*BackEnd*: <https://github.com/JoaoVFG/sysrlog>

### 3. DESENVOLVIMENTO

O presente Capítulo tem como objetivo apresentar a fase de Desenvolvimento do Projeto e será composto por: Arquitetura da Solução, Modelagem e Gestão dos Dados, Arquitetura do Software, Segurança e para concluir a apresentação de uma Visão Geral do Sistema.

#### 3.1. Padrão do Projeto

O padrão de escolhido para o desenvolvimento do projeto é o MVC: *Model, View e Controller*, traduzido ao português para Modelo, Visão e Controlador. O padrão foi escolhido por conta da organização que ele trás ao projeto além de separar o código de maneira lógica facilitando o desenvolvimento.

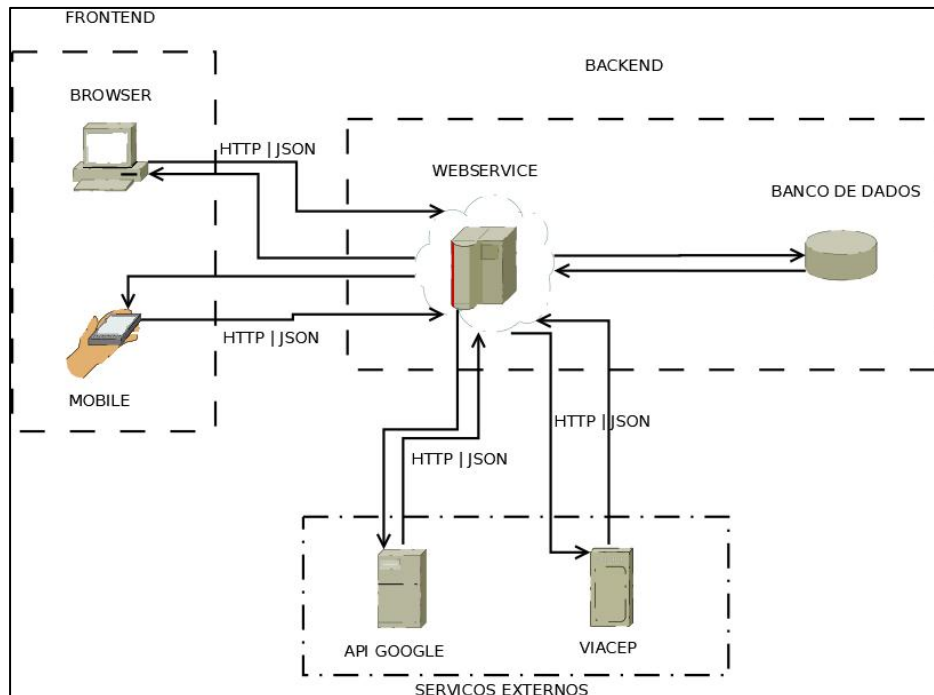
O usuário através do Navegador ou do Celular, fará uma requisição pela tela (View) que será enviada ao Controller. O Controller irá tratar as requisições vindas da View e, se necessário, ira acionar a camada Model, caso não seja, retornara a resposta para View. Se a camada Model tiver sido acionada, ela irá se conectar via um Driver ao Banco de dados e efetuará as devidas operações.

#### 3.2. Arquitetura da Solução

A Arquitetura da Solução foi planejada para que o aplicativo(FrontEnd) seja utilizado tanto em dispositivos móveis quanto em Navegadores. O Aplicativo irá se comunicar com o Software(BackEnd) sendo executada em um Servidor, por meio de requisições HTTP e arquivos no formato JSON para transferência dos dados.

O Software será responsável por receber as requisições, e executar os processos necessários para atender essa requisição, seja acessar a Base de dados ou executar consultas a serviços externos. Quando os processos originados pela requisição forem resolvidos o software devolverá a resposta para o aplicativo, seja ela o resultado de uma busca, um mensagem de confirmação ou uma advertência indicando algum problema com a requisição.

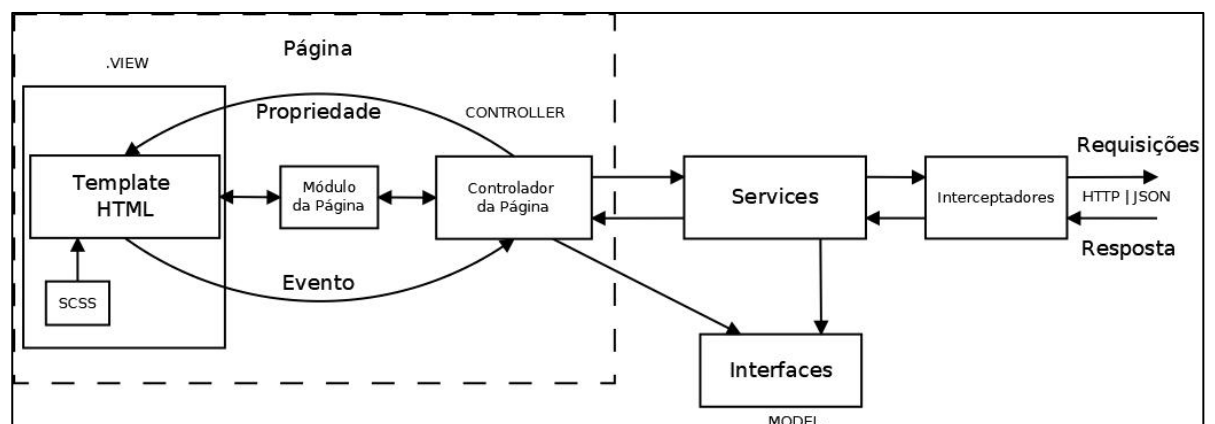
A Figura 8 apresenta esses processos a partir de uma visão macro de toda a aplicação:

**Figura 8. Arquitetura da Solução - BackEnd**

Fonte: O autor

### 3.2.1. Arquitetura da Solução - FrontEnd

O FrontEnd do projeto corresponde a toda uma estrutura necessária para que a View envie requisições ao BackEnd, além de receber e tratar as respostas recebidas. Ao implementar o desenvolvimento utilizando Ionic, Cordova e TypeScript, foi necessário adotar o Padrão MVC também no FrontEnd. A Figura 9, ilustra o Funcionamento do FrontEnd bem como onde estão localizados os componentes do Padrão MVC

**Figura 9. Arquitetura da Solução: FrontEnd**

Fonte: O autor

A estrutura do FrontEnd é composta por:

- I. Páginas: Formada pelos *Templates* HTML e Arquivos SCSS( a combinação destes dois itens constituem a *View* do projeto), o Controlador da Página e pelo Módulo da Página(Responsável por indicar que determinada página é controlada por determinado Controlador). Essa composição cada uma das Páginas é o Padrão utilizado pelo Ionic.
- II. Serviços: Responsáveis por realizar as Requisições Solicitadas pelas Páginas ao *BackEnd*, receber as respostas e devolve-las às Páginas.
- III. Interfaces: Define os Objetos que serão enviados e Recebidos. Só é possível utilizar esse recurso na camada de *FrontEnd* devida a utilização da Linguagem TypeScript.
- IV. Interceptadores: São componentes criados para realizarem duas funções no FrontEnd: adicionar o *Token* de autenticação a todas as requisições que serão realizadas e ao receber um erro do BackEnd, mostra-lo no formato de alerta na tela.

O *FrontEnd* funciona a partir de um evento que ocorre nos Templates HTML,o *click* em um botão por exemplo. O *click* nesse botão aciona um método dentro do Controlador da Página.

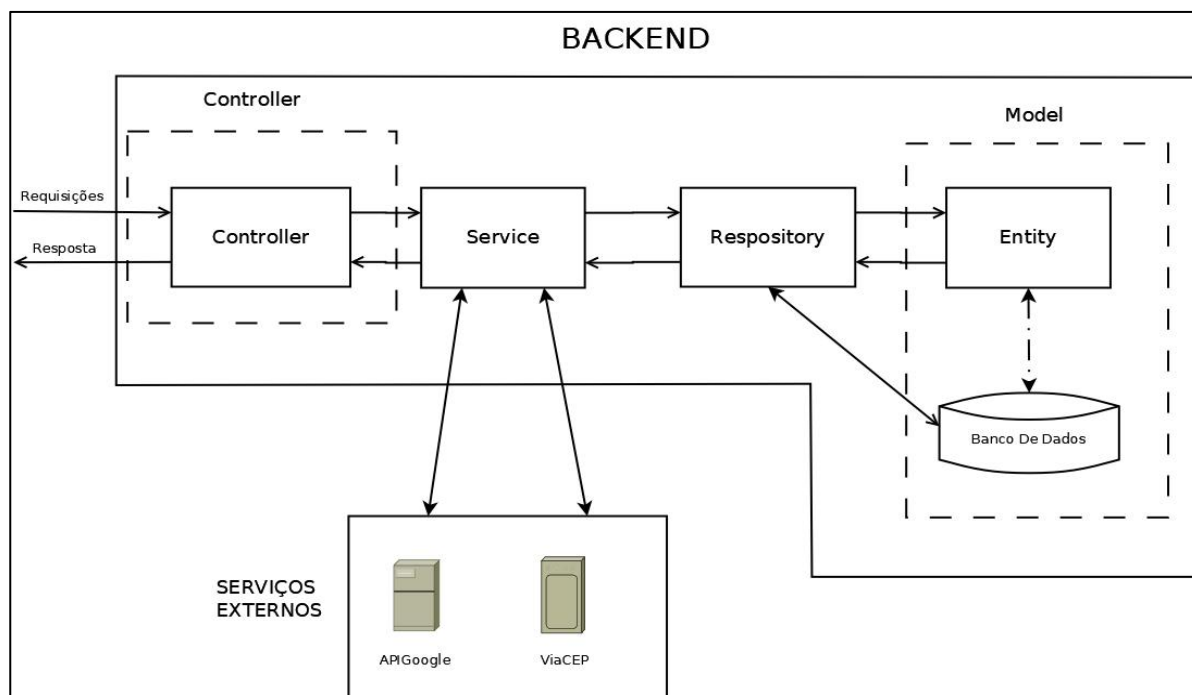
Esse Controlador irá realizar uma requisição irá acionar um método de um dos Serviços, que realizará um Requisição HTTP ao *BackEnd*, passando se necessário, alguma informação pela URL, ou algum objeto definido pelas *Models*, no corpo da requisição. Quando a requisição for efetuada, o Interceptador irá inserir o *Token* no Cabeçalho da requisição.

Ao receber a resposta da requisição o *Interceptador* verificará se a resposta foi um erro, caso não seja, irá deixar ela prosseguir a origem. O Controlador, quando receber a resposta, irá transferir as informações aos objetos que estão ligados ao *FrontEnd*. Quando os dados desses Objetos são alterados, automaticamente eles são apresentados no *Template* ao Usuário isso por meio das Diretivas do Angular.

### 3.2.2. Arquitetura da Solução - BackEnd

O BackEnd do Projeto corresponde a Camada de Model e Controller. A Figura 10 apresenta um maior detalhamento dessas camadas.

**Figura 10. Arquitetura da Solução: BackEnd**



Fonte: O autor

Todas as requisições são recebidas por *Controllers*, eles deverão validar a requisição e repassar a camada de serviço. Por sua vez, a camada de Serviço irá fazer as validações necessárias com os parâmetros passados pelo controlador, e realizar as chamadas necessárias, sejam repositórios ou outro serviço. Caso a ao validar a requisição seja identificado algum erro, a camada de serviço irá retornar a exception para o controlador disparar e apresentar a View.

A camada de serviço é a responsável por realizar acesso ao banco de dados para realizar as operações básicas, Utilizando os as Classes de Entidades como referência.

Após os repositórios executarem seus métodos com sucesso, os serviços irão receber o resultado e retorna-los aos controladores. O Controlador irá devolver a requisição HTTP feita pela view, com o Status(seja de sucesso ou de falha), o respectivo corpo apresentando o resultado da requisição.

### 3.2.3. Arquitetura da Solução - Implantação

Para a implantação do Software é necessária a alocação do BackEnd em um Servidor de Aplicações Java como o TomCat ou ser implantado em um serviço de hospedagem de aplicações como o Heroku. Seja onde o BackEnd for implementado é



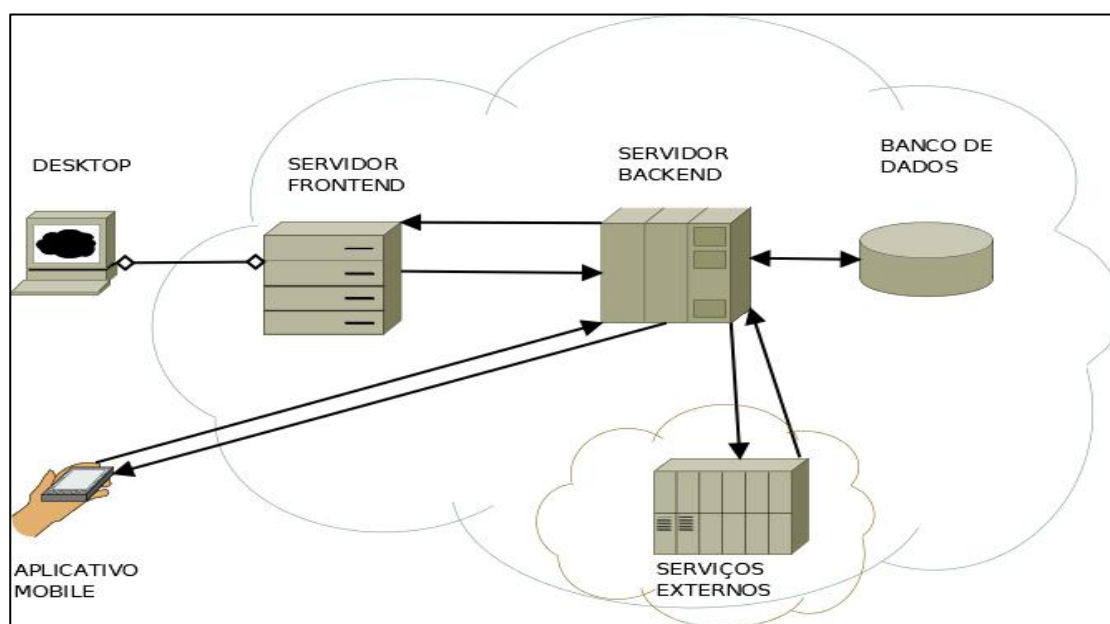
obrigatório ele possua conexão constante com a Internet, para que possa receber requisições e acessar os Serviços Externos.

Além de um servidor de aplicações, é necessário ter um serviço de Banco De Dados MySQL disponível para a armazenagem dos dados do Software. Seja qual for o serviço definido para implantação, basta configurar o acesso dentro dos arquivos de configuração do Spring.

O FrontEnd pode ser implantado em qualquer servidor com Suporte a Linguagem TypeScript e componentes do Ionic e Cordova. Heroku e Firebase são exemplos. Já para dispositivos móveis, basta que o Projeto Ionic seja compilado para o Sistema Operacional respectivo, tendo o BackEnd e o Banco de Dados implantados que ele poderá ser executado.

A Figura 11 apresenta a arquitetura de Implantação do Projeto:

**Figura 11. Arquitetura da Solução: Implantação de Projeto**



Fonte: O autor

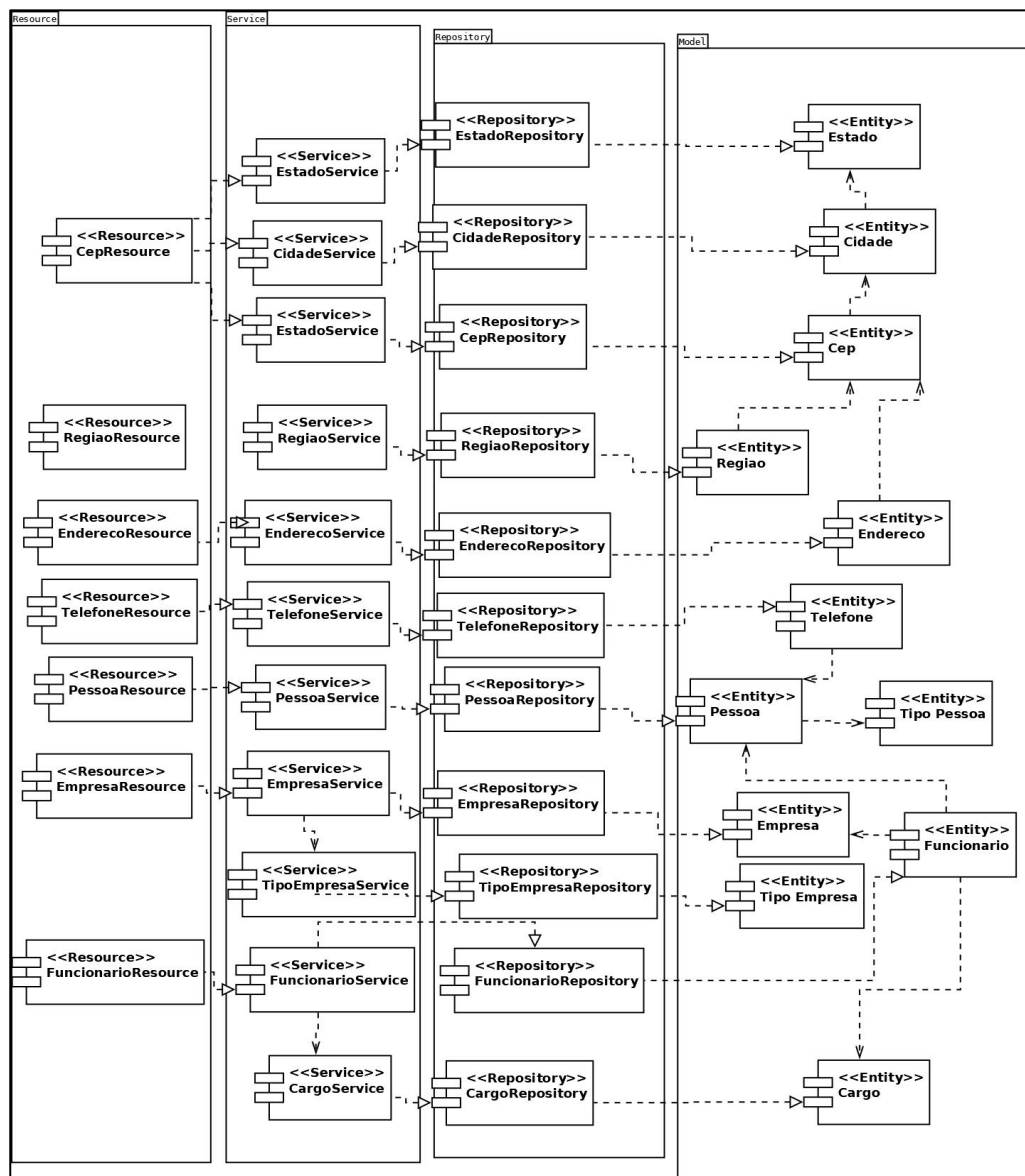
### 3.3. Arquitetura do Software

O presente sub capítulo apresentara a estruturação do código fonte do Projeto, tanto para o BackEnd quanto para o FrontEnd, além de explicar seu funcionamento com alguns fragmentos de Código

### 3.3.1. Diagrama de Componentes

O Diagrama de Componentes apresentado na Figura 12 ilustra os componentes do BackEnd relacionados as funcionalidades de cadastros, e sua relação entre as camadas e empacotamento

**Figura 12. Diagrama de Componentes**



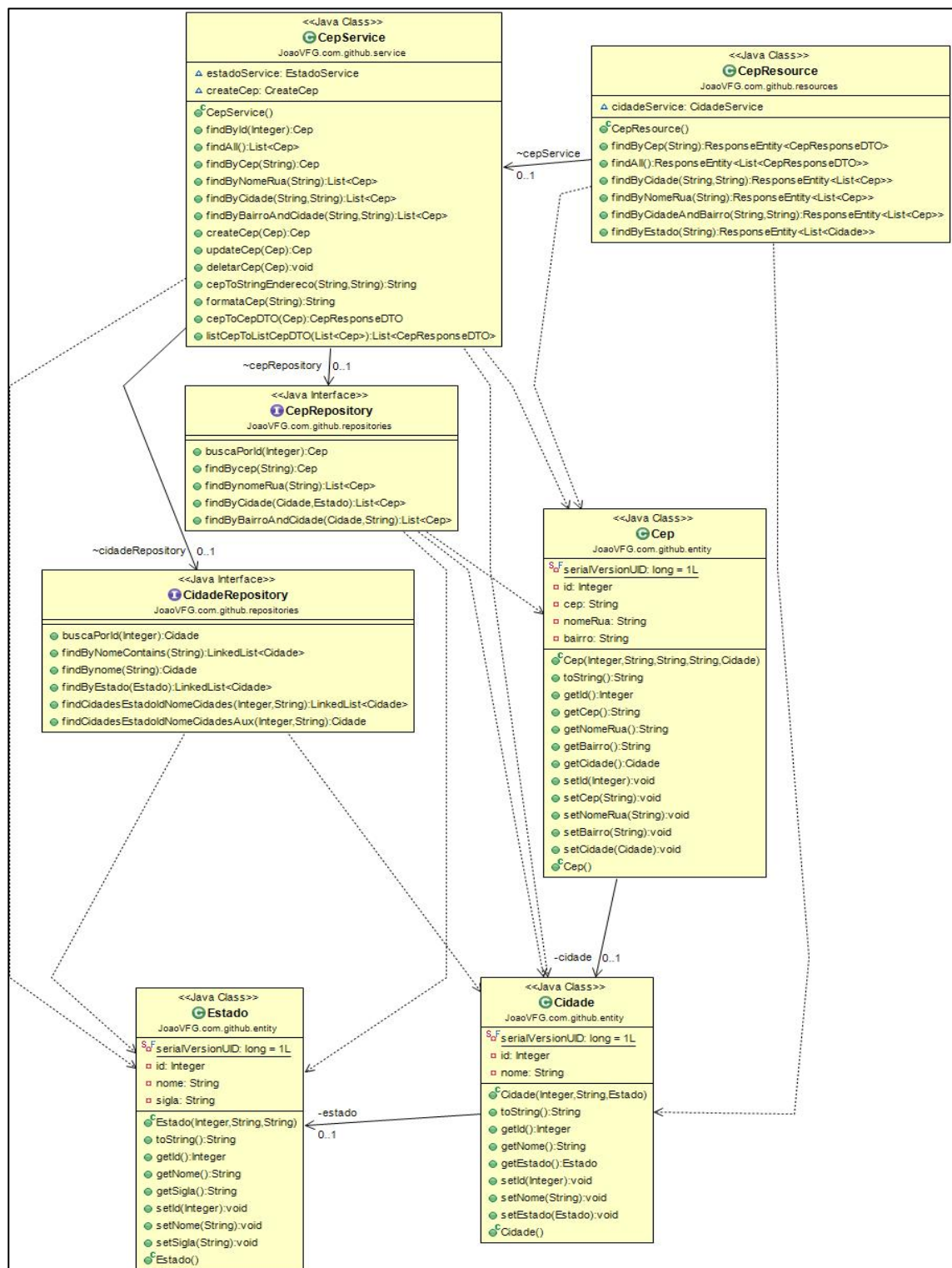
Fonte: O autor

### 3.3.2. Diagramas de Classes

A seguir serão mostrados alguns diagramas de algumas funcionalidades do Projeto.

A Figura 13 apresenta o diagrama de Classes referente as componentes à serem programados necessários para consulta e persistência de Ceps

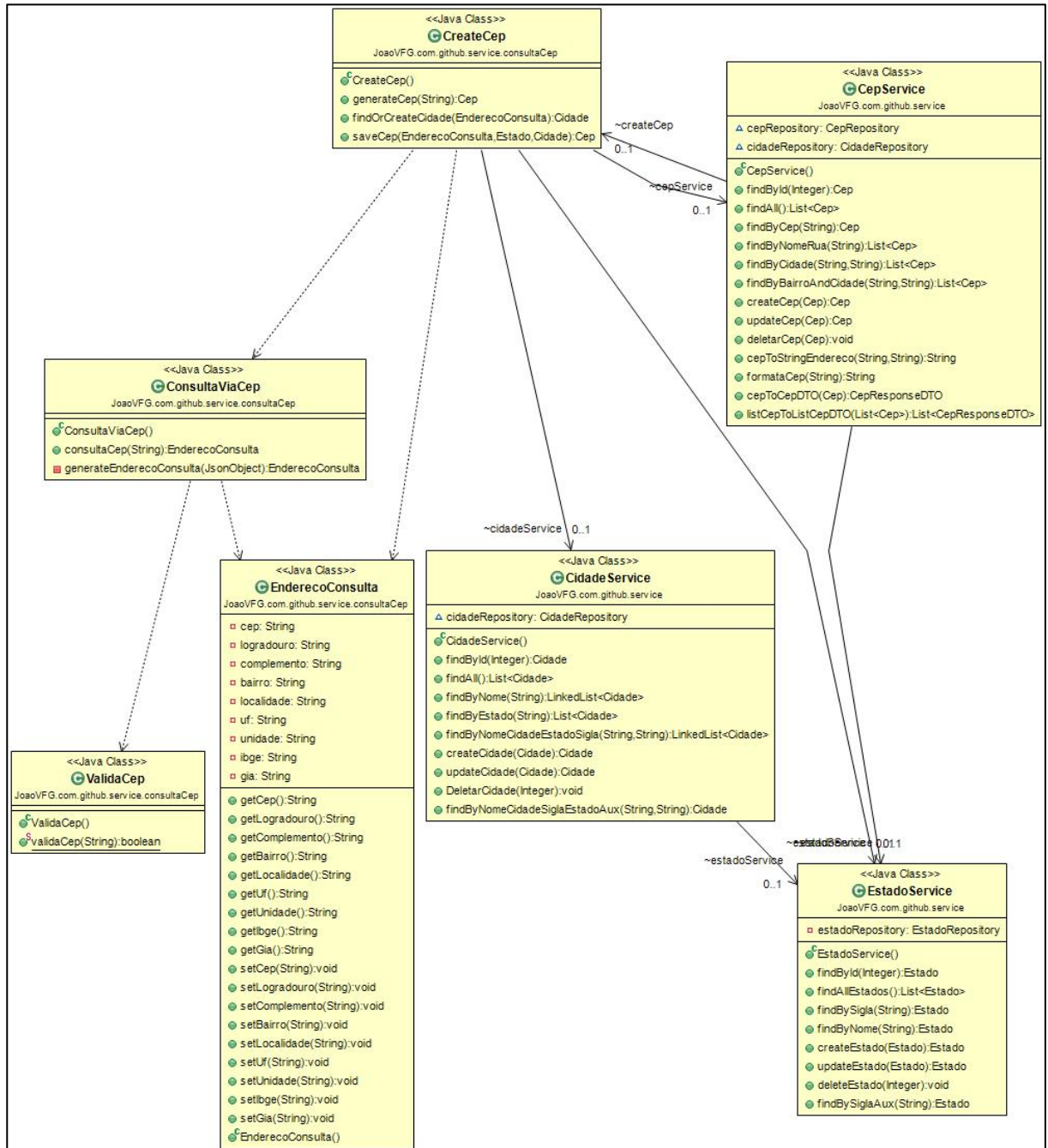
**Figura 13. Diagrama de Classes: Ceps**



Fonte: O autor

A Figura 14 apresenta o diagrama de Classes referente as componentes à serem programados necessários Consultar Ceps no Servidor do ViaCep

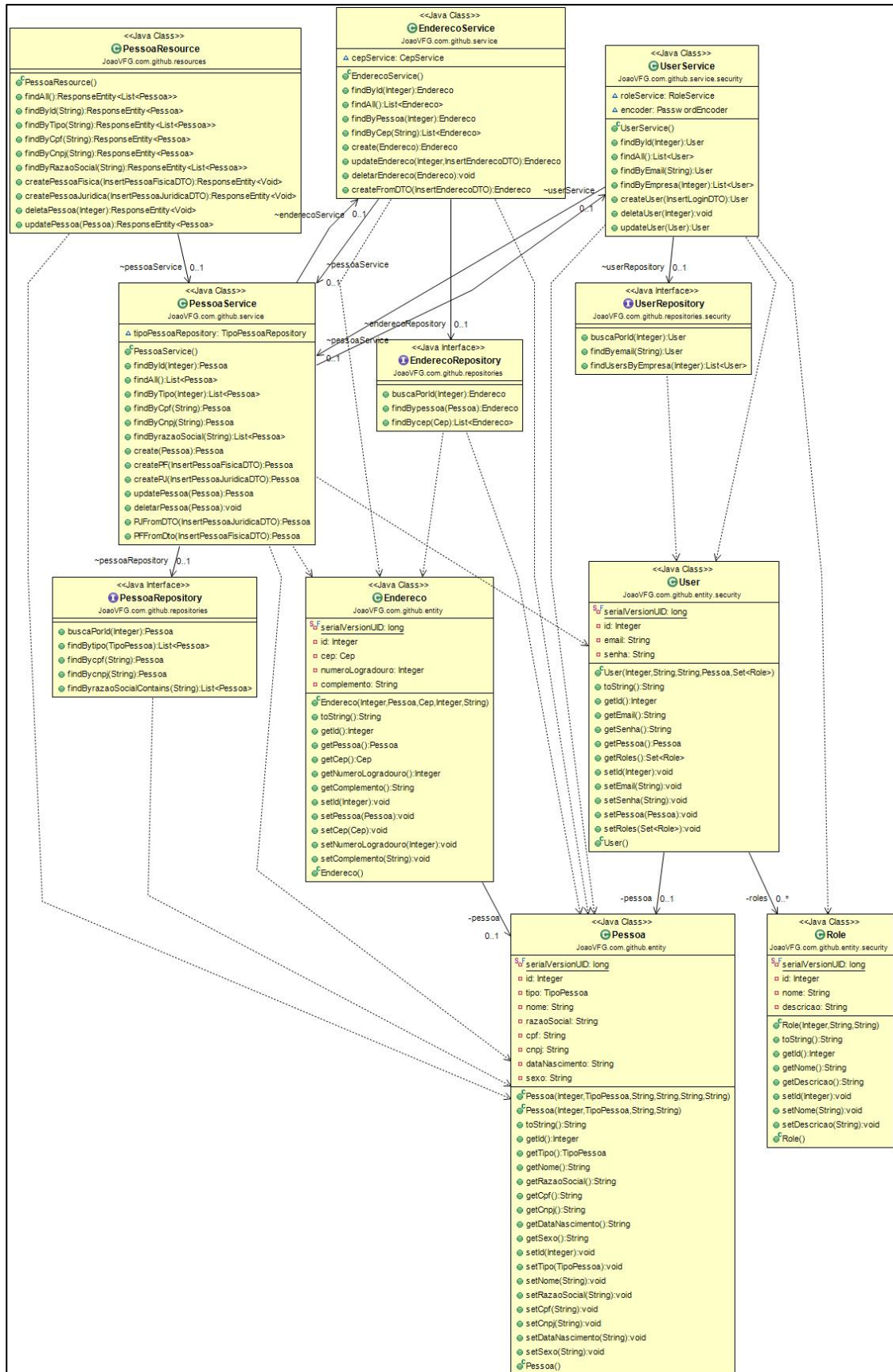
**Figura 14. Diagrama de Classes: Consulta Ceps**



Fonte: O autor

A Figura 15 apresenta o diagrama de Classes referente as componentes à serem programados necessários Persistência e consulta de pessoa, endereço e usuário:

Figura 15. Diagrama de Classes: Pessoa

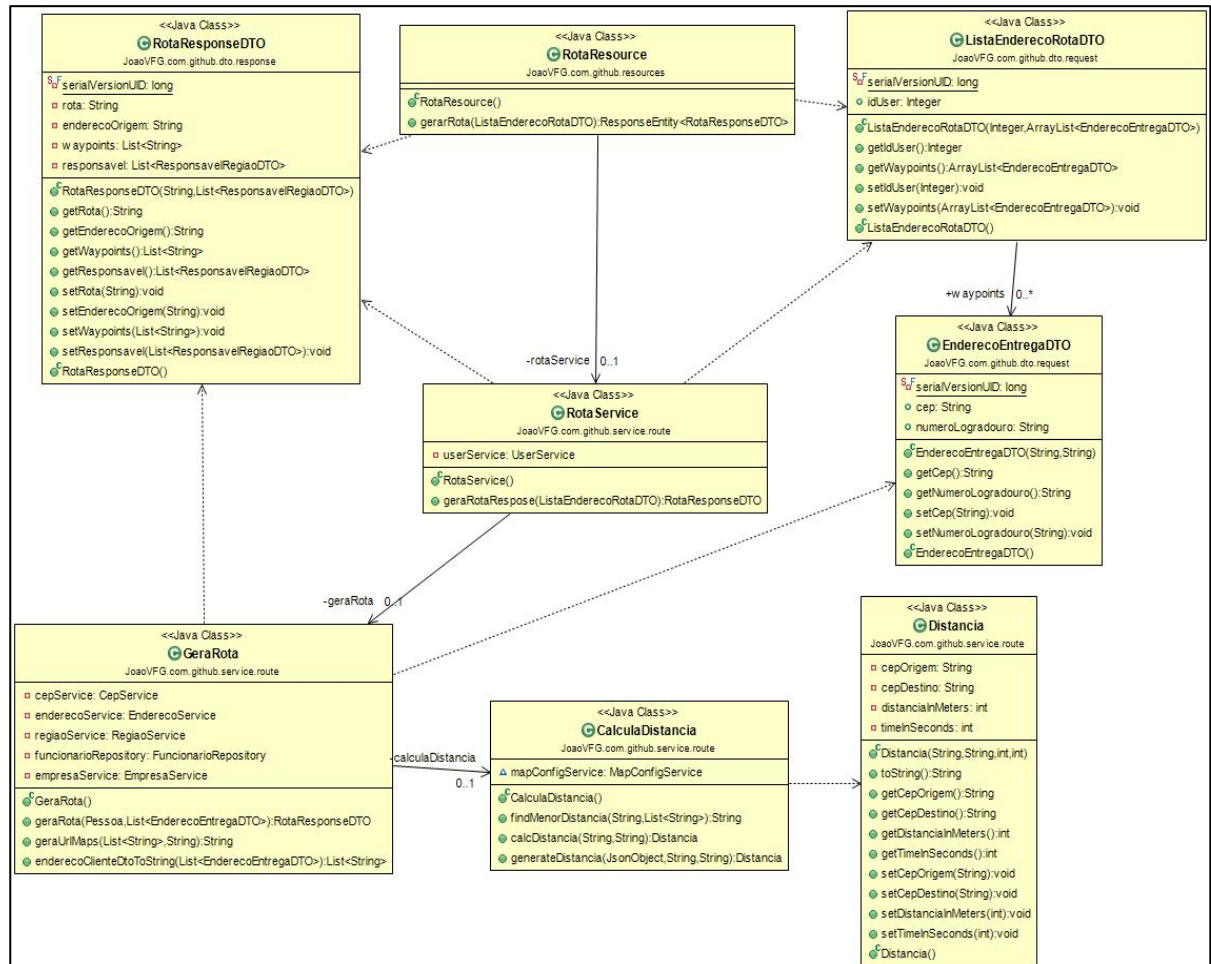


Fonte: O autor



A Figura 16 apresenta o diagrama de Classes referente as componentes à serem programados necessários para gerar rotas:

**Figura 16. Diagrama de Classes: Rota**



Fonte: O autor

### 3.3.3. Exemplificação de Funcionamento do Backend

Toda requisição realizada ao *Backend* é direcionada aos Controladores, eles são os responsáveis por receber as requisições e repassa-las a camada de serviço, ou caso a requisição seja inválida, o próprio *Controller* envia uma mensagem de erro para a *View*. Pelo projeto ser baseado em Spring, deve ser informado as classes que são controladores e suas respectivas URLs de acesso, a Figura 17 exemplifica como é realizada essa parametrização.

**Figura 17. Definição de Controlador e URL**

```

@RestController
@RequestMapping(value="/pessoa")
public class PessoaResource {

```

Fonte: O autor

A anotação `@RestController` tem a função de identificar que a Classe é um Controlador e a anotação `@RequestMapping()` informa o Spring que é necessária uma URL para acessar a Classe. A URL é informada dentro do atributo `value`. Devida a utilização do Spring Boot no projeto, não é necessário nenhuma outra configuração ou parametrização, o Spring Boot escaneia todos os arquivos do projeto e com base nessas anotações já atribui as funções destas Classes.

Anotar as Classes apenas define o caminho para a acessá-las, mas para utilizar suas funcionalidades, os métodos também devem ser anotados com `@RequestMapping` como mostrado na Figura 18.

**Figura 18. Definição de Acesso ao Método usando `@RequestMapping`**

```
@RequestMapping(value="/buscapessoa/id/{id}", method = RequestMethod.GET)
public ResponseEntity<Pessoa> findById(@PathVariable String id){
    Pessoa pessoa = pessoaService.findById(Integer.parseInt(id));
    return ResponseEntity.ok().body(pessoa);
}
```

Fonte: O autor

A anotação de um método necessita de outro atributos além do valor da URL. É necessário informar o tipo da requisição, identificado pelo atributo `method`, no exemplo é informado ao Spring que para acessar o método `findById` é necessária usar a URL ‘pessoa/’ definida para classe, combinada com ‘/buscapessoa/id/{id}’ utilizando o método de requisição do Tipo *get* traduzido como pegar ou obter. Existem ainda outros tipos de métodos de Requisições como *post*(enviar ou publicar), *delete*(deletar) e *put*(colocar).

Na assinatura do método `findById` existe a anotação `@PathVariable` utilizada para indicar que deve ser passado ao método, uma variável de nome `id`, a partir da URL da requisição. Na URL essa variável é identificada dentro de colchetes e deve ter o mesmo nome da variável de assinatura do método.

Dentro do método `findById` é criado uma objeto do tipo `Pessoa`(definido pela camada Model). Esse objeto receberá o retorno da chamada do Método `findById` da Classe de serviço ‘`PessoaService`’. O objeto `pessoaService` que faz a chamada ao método `findById` da Classe de serviço foi instanciado usando Injeção de Dependências pelo Spring, a figura 19 demonstra como utilizar essa funcionalidade.

**Figura 19. Definição Injeção de Dependências**

```
public class PessoaResource {
    @Autowired
    PessoaService pessoaService;
}
```

Fonte: O autor

A anotação *@Autowired* que realizada a injeção de dependência. Nesse caso o objeto de pessoa Service não precisa ser Criado e inicializado, ele apenas é chamado, liberando acesso aos seus métodos quando necessário.

As classes de serviço são as responsáveis por realizar a chamadas as Classes da camada de Repositórios(*Repositories*), realizar validações e o principal, é a responsável pelas Regras de Negócio do Software. Para identificar uma Classe de serviço é necessária anotá-la com *@Service* como mostrado na Figura 20.

**Figura 20. Definição Injeção de Serviço**

```
@Service
public class PessoaService {
```

Fonte: O autor

O Controlador de Pessoa, 'PessoaResource' em seu método *findById*, realizou chamada ao método *findById*, da Classe *PessoaService*, como foi mostrado na Figura 18.

Dentro da Classe *PessoaService* o método *findById* acessa o método *findById* da Interface *PessoaRepository* da camada de Repositório, conforme apresentado na Figura 21. Novamente é realizada Injeção de Dependência, neste caso gerando um objeto *pessoaRepository* do tipo *PessoaRepository*.

**Figura 21. Fragmento da Classe PessoaService**

```
@Autowired
PessoaRepository pessoaRepository;

public Pessoa findById(Integer id) {
    Optional<Pessoa> pessoa = Optional.ofNullable(pessoaRepository.buscaPorId(id));

    return pessoa.orElseThrow(() -> new ObjectNotFoundException(
        "Pessoa não encontrado! Id: " + id + ". Tipo: " + Pessoa.class.getName()));
}
```

Fonte: O autor



As Interfaces de Repositório são as responsáveis por realizar acesso ao Banco de Dados, também conhecida como camada de Persistência. Essa camada tem a função de efetuar as operações de CRUD: *create* (criar), *read* (ler ou busca), *update* (atualizar) e *delete* (deletar). A Figura 22 apresenta a Interface PessoaRepository:

**Figura 22. Interface Pessoa Repository**

```
@Repository
public interface PessoaRepository extends JpaRepository<Pessoa, Integer> {

    @Transactional(readOnly = true)
    @Query("SELECT pessoa FROM Pessoa pessoa WHERE pessoa.id = :id")
    public Pessoa buscaPorId(@Param("id") Integer id);

    @Transactional(readOnly = true)
    public List<Pessoa> findBytipo(TipoPessoa tipoPessoa);

    @Transactional(readOnly = true)
    public Pessoa findBycpf(String cpf);

    @Transactional(readOnly = true)
    public Pessoa findBycnpj(String cnpj);

    @Transactional(readOnly = true)
    public List<Pessoa> findByrazaoSocialContains(String razaoSocial);
}
```

Fonte: O autor

Os repositórios são definidos pela anotação *@repository*. *Eles* são Interfaces que estendem *JpaRepository*, para estender essa classe é necessário que indicar a qual entidade é referenciada pela Interface, no exemplo a classe referenciada é a *Pessoa*, e o seu Identificador é do Tipo *Integer*, conforme na Figura 22, o fragmento ‘*extends JpaRepository<Pessoa, Integer>*’.

Mesmo sendo Interface, quando utilizada a anotação de *@Repository* junto com estender *JpaRepository* a Classe *PessoaRepository* tem suas funções implementadas em tempo de execução pelo próprio Spring, por isso que é possível utilizar seus métodos, sem nenhum tipo de implementação desenvolvida. *JpaRepository* já oferece por padrão, alguns métodos implementados para operações básicas.

Caso exista necessidade de implementar outros métodos, eles devem ser inseridos nos Repositórios e podem ser criados com base em consultas JPQL, como o método ‘*findById*’ na Figura 22 ou por assinatura de método como no método ‘*findBytipo*’ também apresentado na Figura 22.

As entidades são definidas pela anotação `@Entity`, são as classes responsáveis por representarem as tabelas do Banco de Dados em objetos. Elas são denominadas camada *Model* em português Modelo por conta dessa representação. A Figura 23 mostra um fragmento da Classe Pessoa.

**Figura 23. Entidade Pessoa**

```
@Getter
@Setter
@NoArgsConstructor
@Entity
public class Pessoa implements Serializable {
    private static final long serialVersionUID = 1L;

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    private Integer id;

    @ManyToOne
    @JoinColumn(name = "TIPO_ID")
    private TipoPessoa tipo;

    private String nome;

    private String razaoSocial;
```

Fonte: O autor

A anotação `@Entity` indicam ao Spring que essa é uma Entidade. As anotações `@Getter`(gera os métodos get para os atributos) `@Setter`(gera os métodos set para os atributos) e `@NoArgsConstructor`(gera um construtor sem argumentos) são anotações do *Framework Lombok*.

Outra anotação que pode ser observada é a `@ManyToOne`(um para muitos) que indica o relacionamento da tabela ‘Pessoa’ para a tabela ‘TipoPessoa’. Ainda há as anotações `@OneToMany`(muitos para um) e `@ManyToMany`(muitos para muitos).

Retornando ao tratamento da requisição, após o repositório receber o resultado da consulta, ele enviará o retorno ao método da Classe de Serviço. Na classe de Serviço o retorno é inserido dentro de um Objeto *Optional* traduzido para Opcional. Objeto que resumidamente, pode se tornar um tipo de Objeto definido, ou receber *Null*, traduzido para nulo. E quando esse objeto é retornado por uma função, caso ele possua um valor definido caso contrario ele pode automaticamente lançar uma exceção, conforme visto na Figura 21.

Na Classe PessoaResource, caso o esse resultado retornado pela Camada de Serviço seja um objeto, a Camada Controller gerará um Objeto do tipo *ResponseEntity*, traduzido para entidade de resposta, cujo corpo será composto pelo objeto retornado pela Camada de Serviço. O *ObjetoResponseEntity*, retorna um JSON em seu corpo. O resultado será

enviado para o aplicativo, que faz a representa a camada View. Na camada View, o JSON retornado será interpretado e apresentado ao usuário.

### 3.3.4. Exemplificação de Funcionamento do FrontEnd

O frontEnd do Projeto funciona baseado na interação com o usuário. Quando ele clica em um botão, ou clica em um item para detalhamento.

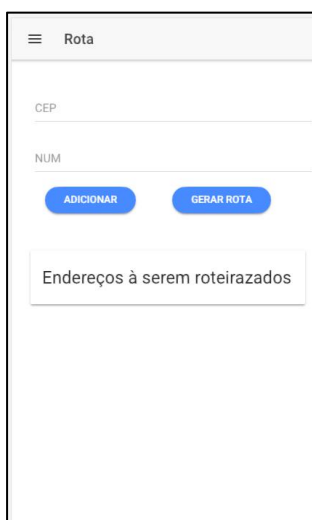
Para uma breve explicação do funcionamento do Front End, utilizaremos como exemplo a função de criação de rotas. Para poder utilizar essa função deve-se acessar a Página de criação de Rotas. No Ionic Página é composta por 4 arquivos:

1. HTML: É a tela, a qual o usuário vê e interage.
2. SCSS: Arquivo que contém as propriedades gráficas da tela, como definição de posicionamento dos objetos, cor de Fonts...
3. Modulo : é o arquivo onde são Injetadas as dependências da Página
4. Controlador da Página: Responsável por Captar os eventos, executar ações e retornar dados ao usuário.

Essas Páginas podem ser gerada automaticamente utilizando o Ionic Cli, tendo ele instalado basta executar a seguinte instrução utilizando um prompt de comando: `ionic g page 'nome da página'` .

Dada a explicação do conceito de Página dentro do Ionic , pode ser analisada a estrutura HTML da Página de geração de rotas. Antes de ser apresentado o código HTML, a Figura 24 mostra a Página, sem ter sofrido interações com o usuário.

**Figura 24. Página de Geração de Rotas**



Fonte: O autor

Podemos Observar alguns itens que são carregados quando a Página é acessada. A Figura 25 apresenta o código cabeçalho desta Página. Nesse cabeçalho é apresentado o nome da Página e inserido o botão para acessar o menu lateral.

**Figura 25. Cabeçalho da Página de Geração de Rotas**

```
<ion-header>
  <ion-navbar>
    <button ion-button menuToggle>
      <ion-icon name="menu"></ion-icon>
    </button>
    <ion-title>Rota</ion-title>
  </ion-navbar>
</ion-header>
```

Fonte: O autor

Podemos Observar também na Figura 24, alguns outros itens. Dois campos para inserir informações, no caso o cep para entrega e o número do cep, e também dois botões, um para adicionar o endereço digitado a lista de endereços a serem roteirizados e outro para solicitar a criação da rota. A Figura 26 apresenta o código HTML referente a essas funcionalidades.

**Figura 26. Fragmento da Página de Geração de Rotas**

```
<ion-item *ngIf="!rotaResponse?.rota">
  <ion-label floating>CEP</ion-label>
  <ion-input [(ngModel)]="enderecoEntrega.cep" name="enderecoCep" type="text"></ion-input>
</ion-item>
<ion-item *ngIf="!rotaResponse?.rota">
  <ion-label floating>NUM</ion-label>
  <ion-input [(ngModel)]="enderecoEntrega.numeroLogradouro" name="enderecoNum" type="text"></ion-input>
</ion-item>

<button *ngIf="!rotaResponse?.rota" name="btnAdicionar">
  ion-button round medium (click)="validaCep()">Adicionar</button>
<button *ngIf="!rotaResponse?.rota" name="btnGerar">
  ion-button round medium (click)="gerarRota()">Gerar Rota</button>
```

Fonte: O autor

As duas primeiras Tags nomeadas 'ion-item' é a união do nome do campo(nomeado como 'ion-lebel') com o campo de inserção de texto(nomeado ion-input). Dentro de ion-input, é utilizada a diretiva ngModel que tem a função de vincular o campo de input com uma variável declarada no Controlador da Pagina, os campos ion-input devem receber um nome e também o tipo de dado. Outro ponto a ser notado é a utilização da diretiva \*ngIf que na posição que esta no código tem a função de mostrar ou não o objeto ion-item. Nesse caso, quando uma rota é gerada, esses dois campos de inserção são ocultados, já que não tem mais utilidade no momento.

Abaixo dos campos de inserção há dois botões, um para inserir o endereço digitado na lista de endereços a ser roteirizada, e o outro serve para enviar a lista de endereços para o BackEnd e esperar a resposta com a rota gerada. Para que essas ações sejam executadas, é utilizada a propriedade (click) indicando um método do controlador da Página. Outro aspecto a ser notado é a utilização da diretiva \*ngIf, para esconder os botões após a rota ser gerada.

O Botão que adiciona o endereço efetua chamada ao método ‘validaCep’ que efetua uma requisição ao BackEnd solicitando o CEP que foi digitado, caso essa resposta seja positiva, esse cep é adicionado a Lista de Endereços para roteirização. Esse processo é mostrado com o fragmento do código do controlador da Página de Geração de Rotas apresentada na Figura 27.

**Figura 27. Fragmento de Código, Controlador Pagina de Geração de Rotas**

```
validaCep() {
  this.cepService.findByCep(this.enderecoEntrega.cep)
    .subscribe((response) => {
      this.addToList();
    }, error => {
      console.log(error);
    })
}

addToList() {
  this.listaEndereco.waypoints.push({
    cep: this.enderecoEntrega.cep,
    numeroLogradouro: this.enderecoEntrega.numeroLogradouro
  });
  this.enderecoEntrega.cep = '';
  this.enderecoEntrega.numeroLogradouro = '';
}
```

Fonte: O autor

O método Valida Cep, executa o método ‘findByCep’ de cepService Figura28, esse método irá realizar um requisição ao BackEnd e receber a resposta. Caso seja um erro, um alerta será mostrado na Tela. Se a requisição obtiver uma resposta, o método addToList será acionado, onde irá inserir esse endereço a Lista a ser roteirizada, e também irá limpar o valor dos campos de cep e número do endereço.

**Figura 28. Fragmento da Classe CepService**

```
@Injectable()
export class CepService{

  constructor(public http : HttpClient){

  }

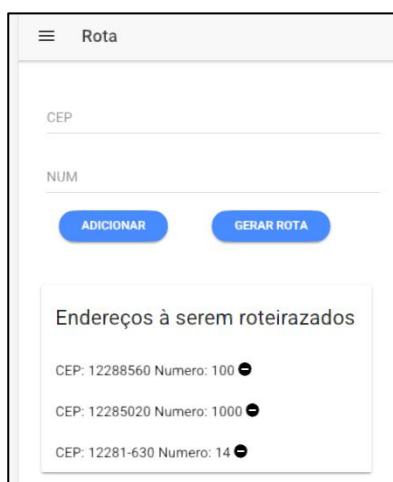
  findByCep(cep : String) : Observable<cep>{
    return this.http.get<cep>(`${API_CONFIG.baseUrl}/ceps/buscape/` + cep);
  }
}
```

Fonte: O autor

Devem ser destacados alguns Pontos da Classe de serviço CepService. Toda Classe que será utilizada em algum outro ponto da aplicação deve usar a Anotação `@Injectable`, para que ela possa ser injetada e usada em outras Classes. Todos os recursos que serão utilizados por uma classe, devem ser injetados no construtor da classe, em CepService por exemplo existe a injeção de um objeto HttpClient, o cliente padrão para requisições Http. Os métodos das classes de serviço devem retornar a execução de um requisição Http de qualquer tipo, passando como parâmetro obrigatório a URL e se necessário objetos no corpo da requisição.

Realizada a adição de alguns endereços para entrega o Card com os pontos de entrega a serem roteirizados será preenchido como apresentado na Figura 29.

**Figura 29. Tela para Geração e Rotas, com Endereços Inseridos**



Fonte: O autor

Para essa listagem de endereços ser apresentada é utilizada a diretiva `*ngFor` do Angular, responsável por gerar dinamicamente uma lista de itens que podem ter as suas informações acessadas no HTML da Página mostrado na Figura 30, com base em um vetor declarado no Controlador da Pagina.

**Figura 30. Tela para Geração e Rotas, com Endereços Inseridos**

```
<ion-card *ngIf="!rotaResponse?.rota">
  <ion-item>
    <ion-card-title>Endereços à serem roteirizados</ion-card-title>
  </ion-item>
  <ion-list>
    <ion-item *ngFor="let end of listaEndereco.waypoints">
      CEP: {{end.cep}} Numero: {{end.numeroLogradouro}}
      <ion-icon name="remove-circle"
        [click]="removeOfList(end.cep, end.numeroLogradouro)"></ion-icon>
    </ion-item>
  </ion-list>
</ion-card>
```

Fonte: O autor



Assim os campos do objeto que está sendo percorrido é acessado através do denominado ‘binding de atributo’ que pode ser entendido como vínculo do elemento dentro da página com uma variável, nesse caso é a variável gerada dinamicamente pelo \*ngFor.

Outro elemento a ser destacado é o ícone de remoção que tem função de remover o endereço selecionado da Lista de Endereços, esse ícone quando clicado aciona o método removeOfList. Que irá buscar a posição no vetor que o endereço encontra-se, e irá removê-lo do vetor. O fragmento de código que realiza essa função está apresentado na figura 31.

**Figura 31. Método para Remoção de Endereço da Lista.**

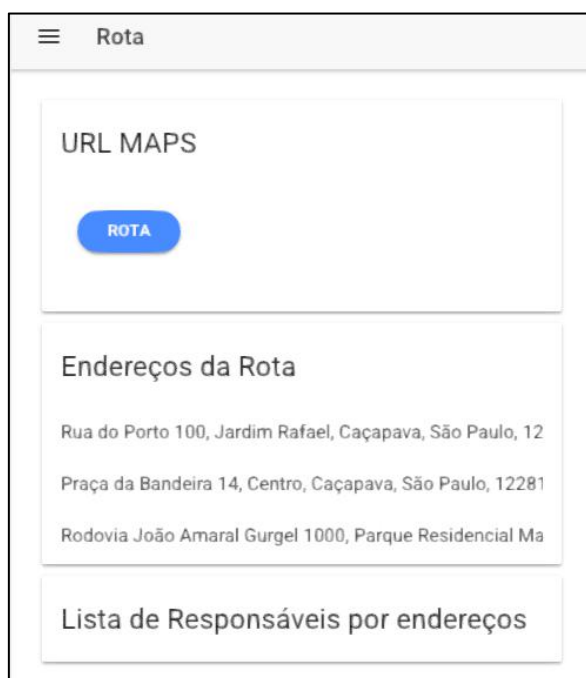
```
removeOfList(cepBusca: string, numLogradouro: string) {
  let waypoints: enderecoEntregaDTO = {
    cep: cepBusca,
    numeroLogradouro: numLogradouro
  }

  let index: number = this.listaEndereco.waypoints.findIndex
    (e => e.cep === cepBusca && e.numeroLogradouro === numLogradouro);
  this.listaEndereco.waypoints.splice(index, 1);
}
```

Fonte: O autor

Ao final da inserção dos endereços de entrega, deverá ser clicado o botão de geração de rotas. A tela será alterada para apresentar os dados da Rota gerada conforme Figura 32.

**Figura 32. Tela de Geração de Rotas após executada a Geração.**



Fonte: O autor

Os campos para inserção de endereços são ocultados, assim como a Lista de Endereços para ser roteirizada. É apresentado um Card, com o botão para abrir a rota no Maps, a Lista de endereços onde deverão ser realizadas as entregas, e por fim, caso a empresa use a parametrização por regiões, será mostrado a relação de entregas que deverão ser realizadas por outras filiais. O método responsável por realizar a solicitação para a criação de rotas é mostrado na Figura 33.

**Figura 33. Tela de Geração de Rotas após executada a Geração.**

```
gerarRota() {  
  let loading = this.loadingService.presentLoading();  
  this.rotaService.geraRotaJson(this.listaEndereco)  
    .subscribe(response => {  
    this.rotaResponse = JSON.parse(response.body);  
    loading.dismiss();  
  },  
  error => {  
    loading.dismiss();  
    console.log(error);  
  })  
}
```

Fonte: O autor

Nesse fragmento de código, é criado um objeto de *Loading* que apresentará uma animação de carregamento, até que seja retornada alguma resposta. Se a resposta for de sucesso, o objeto *rotaResponse*, irá receber o corpo da resposta, que contém os dados da rota, a URL, os endereços onde a entrega será efetuada, e os endereços atendidos por outras filiais. E esse objeto *rotaResponse* que é apresentado na tela ao usuário.



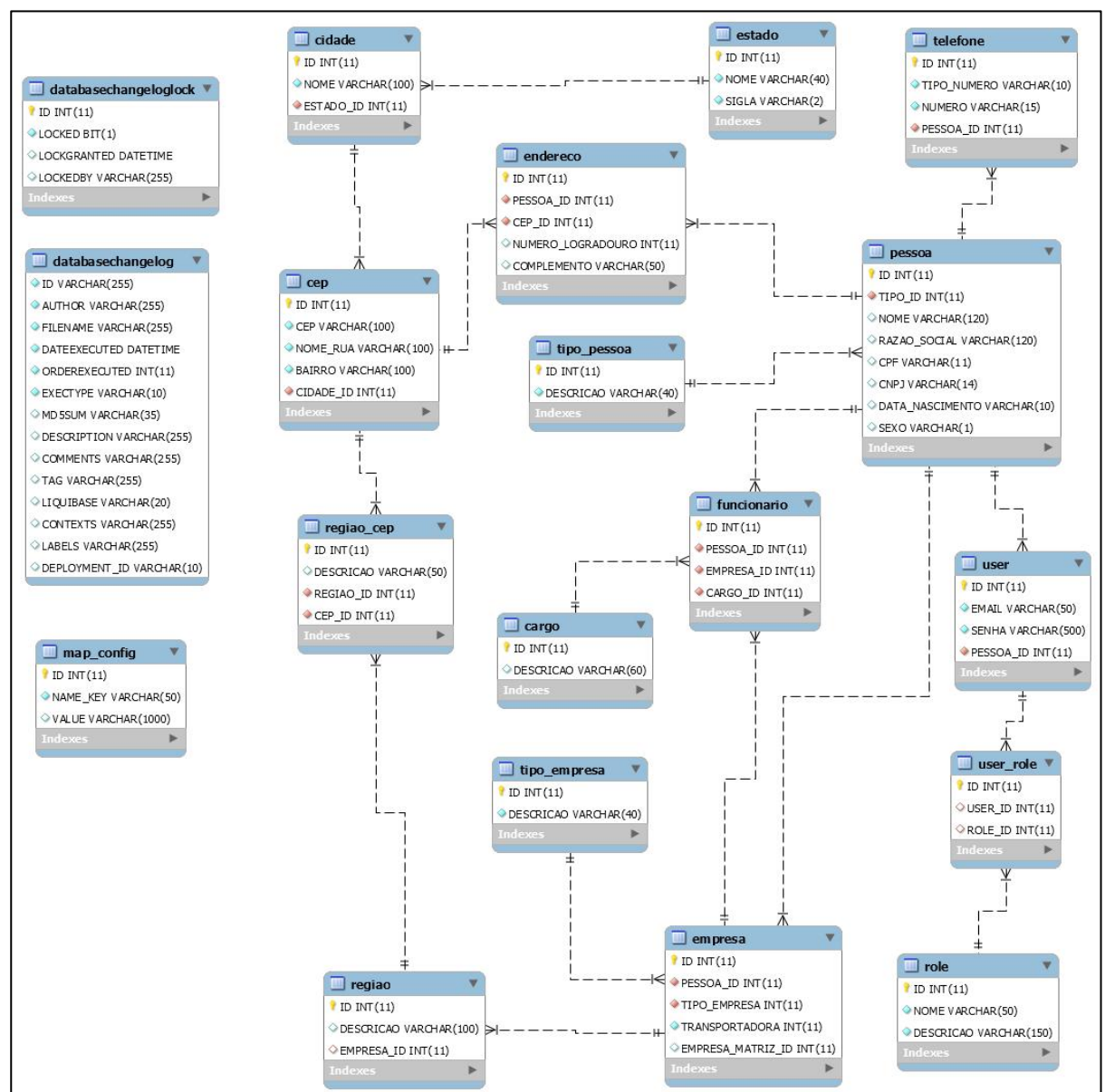
### 3.4. Modelagem e Gestão dos Dados

No presente SubCapítulo será apresentado a modelagem do Banco de Dados, Contemplando o Modelo Entidade Relacionamento, Dicionário de Dados detalhando as tabelas e campos do Banco de dados e utilização do Liquibase no Projeto.

#### 3.4.1. Modelo de Entidade Relacionamento

O Modelo Entidade Relacionamento representado na Figura 34 mostra as Tabelas do Banco De Dados, seus atributos e seus Relacionamentos.

**Figura 34. Modelo Entidade Relacionamento**



Fonte: O autor

### 3.4.2. Dicionário de Dados

O Dicionário de Dados tem a função de descrever de forma Objetiva as Tabelas, os seus Atributos, seus Tipos e a função de cada campo. As siglas FK vêm de Foreign Key que significa Chave Estrangeira e PK vem de Primary Key que significa Chave Primária.

A Tabela 33 descreve a Tabela que armazena as Informações de Cargo:

**Tabela 33. Dicionário de Dados: Tabela Cargo**

Cargo				
Descrição Da Tabela	Tabela para cadastro dos cargos do funcionários			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Cargo
DESCRICA	VARCHAR(60)	Não	Não	Campo para descrever o Cargo

Fonte: O autor

A Tabela 34 descreve a Tabela que armazena as Informações de Cep:

**Tabela 34. Dicionário de Dados: Tabela Cep**

Tabela:cep				
Descrição Da Tabela	Tabela que contem os ceps e a cidade respectiva			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para CEP
CEP	VARCHAR(100)	Não	Não	Campo do CEP
NOME_RUA	VARCHAR(100)	Não	Não	Campo do Nome da Rua
BAIRRO	VARCHAR(100)	Não	Não	Campo para o nome do Bairro
CIDADE_ID	INT(11)	Não	Sim	Campo para FK com Cidade(Atrbutu Id)

Fonte: O autor

A Tabela 35 descreve a Tabela que armazena as Informações de Cidade:

**Tabela 35. Dicionário de Dados: Tabela Cidade**

Tabela:cidade				
Descrição Da Tabela	Tabela que contem as cidades e o estado respectivo			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Estado
NOME	VARCHAR(100)	Não	Não	Campo para o Nome do Estado
ESTADO_ID	INT(11)	Não	Sim	Campo para Sigla do Estado

Fonte: O autor

A Tabela 36 descreve a Tabela que armazena as Informações de Empresa:

**Tabela 36. Dicionário de Dados: Tabela Empresa**

Tabela:empresa				
Descrição Da Tabela	Tabela de cadastro das empresas			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Empresa
PESSOA_ID	INT(11)	Não	Sim	Campo para FK com Pessoa(Atributo Id)
TIPO_EMPRESA	INT(11)	Não	Sim	Campo para FK com Tipo_empresa(Atributo Id)
TRANSPORTADORA	INT(11)	Não	Não	Campo para definir Se é transportadora (1 - Sim ; 0 Não)
EMPRESA_MATRIZ ID	INT(11)	Não	Não	Campo para referenciar a Empresa Matriz(Ligado a Pessoa)

Fonte: O autor

A Tabela 37 descreve a Tabela que armazena as Informações de Endereço:

**Tabela 37. Dicionário de Dados: Tabela Endereço**

Tabela:endereco				
Descrição Da Tabela	Tabela que contem os celulares das pessoas			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Endereço
PESSOA_ID	INT(11)	Não	Sim	Campo para FK com Pessoa(Atributo Id)
CEP_ID	INT(11)	Não	Sim	Campo para FK com Cep(Atributo Id)
NUMERO_LOGRADOURO	INT(11)	Não	Não	Campo para armazenar o Número do Endereço
COMPLEMENTO	VARCHAR(50)	Não	Não	Campo para armazenar o Complemento do Endereço

Fonte: O autor

A Tabela 38 descreve a Tabela que armazena as Informações de Estado:

**Tabela 38. Dicionário de Dados: Tabela Estado**

Tabela:estado				
Descrição Da Tabela	Tabela que contem estados e Siglas			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Estado
NOME	VARCHAR(40)	Não	Não	Campo para armazenar Nome do Estado
SIGLA	VARCHAR(2)	Não	Não	Campo para armazenar a Sigla do Estado

Fonte: O autor

A Tabela 39 descreve a Tabela que armazena as Informações de Funcionário:

**Tabela 39. Dicionário de Dados: Tabela Funcionário**

Tabela:Funcionário				
Descrição Da Tabela	Tabela para cadastro dos funcionários			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Funcionário
PESSOA_ID	INT(11)	Não	Sim	Campo para FK com Pessoa(Atributo Id)
EMPRESA_ID	INT(11)	Não	Sim	Campo para FK com Empresa(Atributo Id)
CARGO_ID	INT(11)	Não	Sim	Campo para FK com Cargo(Atributo Id)

Fonte: O autor

A Tabela 40 descreve a Tabela que armazena as Informações do Mapeamento de Configuração:

**Tabela 40. Dicionário de Dados: Tabela Map\_config**

Tabela:map_config				
Descrição Da Tabela	Tabela para salvar Chaves de configuração			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para MAP_CONFIG
NAME_KEY	VARCHAR(50)	Não	Não	Nome da chave de configuração
VALUE	VARCHAR(1000)	Não	Não	Nome do valor da Chave de configuração

Fonte: O autor

A Tabela 41 descreve a Tabela que armazena as Informações de Pessoa:

**Tabela 41. Dicionário de Dados: Tabela Pessoa**

Tabela:pessoa				
Descrição Da Tabela	Tabela que cadastro das pessoas que usam o sistema			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Pessoa
TIPO_ID	INT(11)	Não	Sim	Campo para FK com Tipo_Pessoa(Atributo Id)
NOME	VARCHAR(120)	Não	Não	Campo para armazenar Nome da Pessoa
RAZAO_SOCIAL	VARCHAR(120)	Não	Não	Campo para armazenar Razão Social da Pessoa
CPF	VARCHAR(11)	Não	Não	Campo para armazenar CPF da Pessoa
CNPJ	VARCHAR(14)	Não	Não	Campo para armazenar CNPJ da Pessoa
DATA_NASCIMENTO	VARCHAR(10)	Não	Não	Campo Data de nascimento da pessoa
SEXO	VARCHAR(1)	Não	Não	Campo para armazenar sexo da Pessoa

Fonte: O autor

A Tabela 42 descreve a Tabela que armazena as Informações de Região:

**Tabela 42. Dicionário de Dados: Tabela Região**

Tabela:Região				
Descrição Da Tabela	Tabela para cadastro das regiões de entrega das empresas			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Região
DESCRICAO	VARCHAR(100)	Não	Não	Descrição da Região(facilitar Identificação)
EMPRESA_ID	INT(11)	Não	Sim	Campo para FK com Empresa(Atributo Id)

Fonte: O autor

A Tabela 43 descreve a Tabela que armazena as Informações de Role, ou Autorizações:

**Tabela 43. Dicionário de Dados: Tabela Roles**

Tabela:role				
Descrição Da Tabela	TABELA PARA CONTROLE DAS AUTORIZACOES			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para Role
NOME	VARCHAR(50)	Não	Não	Campo para armazenar Nome da Role
DESCRICAO	VARCHAR(150)	Não	Não	Campo para armazenar a descrição da Role

Fonte: O autor

A Tabela 44 descreve a Tabela que armazena as Informações de Telefone:

**Tabela 44. Dicionário de Dados: Tabela Telefone**

Tabela:telefone				
Descrição Da Tabela	Tabela que contem os celulares das pessoas			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para
TIPO_NUMERO	VARCHAR(10)	Não	Não	Campo para identificar se é celular ou fixo
NUMERO	VARCHAR(15)	Não	Não	Campo para armazenar o número do Celular
PESSOA_ID	INT(11)	Não	Sim	Campo para FK com Pessoa(Atributo Id)

Fonte: O autor

A Tabela 45 descreve a Tabela que armazena as Informações de Tipo da Empresa:

**Tabela 45. Dicionário de Dados: Tabela Tipo\_Empresa**

Tabela:tipo_empresa				
Descrição Da Tabela	Tabela que para diferenciar matriz de filiais			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para
DESCRICAÇÃO	VARCHAR(40)	Não	Não	Descrição do Tipo da Empresa

Fonte: O autor

A Tabela 46 descreve a Tabela que armazena as Informações do Tipo da Pessoa:

**Tabela 46. Dicionário de Dados: Tabela Tipo\_Pessoa**

Tabela:tipo_pessoa				
Descrição Da Tabela	Tabela que contem os tipos das pessoas			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para
DESCRICA	VARCHAR(40)	Não	Não	Descrição do Tipo da Pessoa

Fonte: O autor

A Tabela 47 descreve a Tabela que armazena as Informações de Usuário:

**Tabela 47. Dicionário de Dados: Tabela User**

Tabela:user				
Descrição Da Tabela	Tabela para cadastro dos usuários			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para
EMAIL	VARCHAR(50)	Não	Não	Campo para armazenar email do usuário
SENHA	VARCHAR(500)	Não	Não	Campo para armazenar senha do usuário
PESSOA_ID	INT(11)	Não	Sim	Campo para FK com Pessoa(Atributo Id)

Fonte: O autor

A Tabela 48 descreve a Tabela que armazena as Informações de Usuário e suas Autorizações:

**Tabela 48. Dicionário de Dados: Tabela User\_Role**

Tabela:user_role				
Descrição Da Tabela	Tabela N pra N de User para Role			
Campos				
Nome Do Campo	Tipo do Campo	PK	FK	Comentário
ID	INT(11)	Sim	Não	Campo identificador para
USER_ID	INT(11)	Não	Sim	Campo para FK com User(Atributo Id)
ROLE_ID	INT(11)	Não	Sim	Campo para FK com Role(Atributo Id)

Fonte: O autor

### 3.4.3. Liquibase

A Biblioteca Liquibase é capaz de executar tanto as instruções DDL como DML. A criação do Banco de Dados do Projeto foi toda realizada através do Liquibase utilizando arquivos XML e SQL.

Para poder utilizar a Biblioteca, é necessário primeiramente adicionar sua dependência ao arquivo POM.xml. Realizada a adição da dependência é necessário especificar ao SpringBoot a localização do arquivo principal do Liquibase, dentro do arquivo application.properties utilizando a propriedade 'spring.liquibase.change-log' conforme apresentado na Figura 35.

**Figura 35. Propriedade do Liquibase em application.properties**

```
server.port=${port:8000}

spring.profiles.active=dev

spring.liquibase.change-log=classpath:db/liquibase-changelog.xml

app.jwtExpirationInMs = 604800000
```

Fonte: O autor

Esse arquivo é denominado dentro da documentação do liquibase como ChangeLogMaster que pode ser entendido como o arquivo responsável por controlar a execução de outros arquivos. Dentro do arquivo liquibase-changelog.xml foi inserido a utilização de diversos arquivos conforme fragmento mostrado na Figura 36.



**Figura 36. Fragmento do Arquivo liquibase-changelog.xml**

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <databaseChangeLog
3   xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
6     http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.1.xsd">
7   <include file="changelog/01-create-estado.xml" relativeToChangelogFile="true"/>
8   <include file="changelog/02-insert-estados.xml" relativeToChangelogFile="true"/>
9   <include file="changelog/03-create-cidade.xml" relativeToChangelogFile="true"/>
10  <include file="changelog/04-create-cep.xml" relativeToChangelogFile="true"/>
11  <include file="changelog/05-add-autoincrement.xml" relativeToChangelogFile="true"/>
12  <include file="changelog/06-add-foreignkey-cidade.xml" relativeToChangelogFile="true"/>
13  <include file="changelog/07-add-foreignkey-cep.xml" relativeToChangelogFile="true"/>
14  <include file="changelog/08-create-tipopessoa.xml" relativeToChangelogFile="true"/>
15  <include file="changelog/09-create-pessoa.xml" relativeToChangelogFile="true"/>
16  <include file="changelog/10-add-foreignkey-pessoa.xml" relativeToChangelogFile="true"/>
17  <include file="changelog/11-create-telefone.xml" relativeToChangelogFile="true"/>

```

Fonte: O autor

A Figura 37 apresenta o arquivo '01-create-estado.xml' onde é realizada a criação da Tabela Estado utilizando o formato XML.

**Figura 37. Fragmento do Arquivo liquibase-changelog.xml**

```

1 <?xml version="1.0" encoding="UTF-8"?>
2 <databaseChangeLog
3   xmlns="http://www.liquibase.org/xml/ns/dbchangelog"
4   xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
5   xsi:schemaLocation="http://www.liquibase.org/xml/ns/dbchangelog
6     http://www.liquibase.org/xml/ns/dbchangelog/dbchangelog-3.1.xsd">
7
8   <changeSet id="01" author="JoãoVFG">
9
10    <createTable tableName="ESTADO"
11      remarks="Tabela que contém estados e Siglas">
12
13      <column name="ID" type="int">
14        <constraints primaryKey="true" unique="true"/>
15      </column>
16
17      <column name="ACRQ" type="varchar(40)">
18        <constraints unique="true" nullable="false" />
19      </column>
20
21      <column name="SIGLA" type="varchar(2)">
22        <constraints unique="true" nullable="false" />
23      </column>
24
25    </createTable>
26
27  </changeSet>
28
29 </databaseChangeLog>

```

Fonte: O autor

Para todo arquivo a ser executado pelo Liquibase deve ser informado o Id do ChangeSet o autor. Todas as tags a serem utilizadas podem ser encontradas no Site Oficial do Liquibase.



A utilização do Liquibase é de grande importância pois centraliza o controle do Banco de Dados junto ao projeto, facilitando o a gestão de mudanças. No desenvolvimento caso seja realizada alguma alteração nas classes de Entidade, é necessário apenas inserir a instrução que reflita essa alteração no Banco de Dados, dentro de um arquivo XML ou SQL e inseri-lo dentro do Liquibase-changelog.xml essa facilidade faz com que o desenvolvedor não necessite abrir um SGBD para realizar as alterações, economizando tempo.

### 3.5. Segurança

A Segurança do Projeto ficará sob responsabilidade do módulo de segurança do Spring, unido à utilização de Token JWT.

Para a utilização do SpringSecurity devem ser desenvolvidas algumas Classes. A Primeira é a Classe UserPrincipal que implementa UserDetails e seus métodos, sendo obrigatória para o funcionamento o JWT. Suas funcionalidades são todas focadas em prover as informações de Usuários e suas respectivas Permissões com base no formato de Usuário utilizados na implementação do Projeto. Na Figura 38 é mostrada a definição da Classe e implementação de alguns métodos de UserDetails.

**Figura 38. Classe UserPrincipal e Alguns Métodos**

```
public class UserPrincipal implements UserDetails {
    private static final long serialVersionUID = 1L;

    public static UserPrincipal create(User user) {
        List<GrantedAuthority> authorities = user.getRoles().stream().map(role ->
            new SimpleGrantedAuthority(role.getName())).collect(Collectors.toList());

        String name = null;
        if(user.getPessoa().getTipo().getId() == 1) {
            name = user.getPessoa().getRazaoSocial();
        }else {
            name = user.getPessoa().getNome();
        }

        return new UserPrincipal(user.getId(), name, user.getEmail(), user.getSenha(), authorities);
    }

    @Override
    public Collection<? extends GrantedAuthority> getAuthorities() {
        return authorities;
    }

    @Override
    public String getPassword() {
        return password;
    }

    @Override
    public String getUsername() {
        return email;
    }
}
```

Fonte: O autor

A Classe é a `JWTTokenProvider`, que tem a função de gerar um Token com base na autenticação informada, além de também validar Tokens recebidos. A seguir será apresentado na Figura 39 um trecho da Classe `JWTTokenProvider`, com o método responsável por gerar o Token de autenticação.

**Figura 39. Método Generate Token de `JWTTokenProvider`**

```
public String generateToken(Authentication authentication) {
    UserPrincipal userPrincipal = (UserPrincipal) authentication.getPrincipal();

    Date now = new Date();
    Date expiryDate = new Date(now.getTime() + jwtExpirationInMs);

    return Jwts.builder()
        .setSubject(Integer.toString(userPrincipal.getId()))
        .setIssuedAt(new Date()).setExpiration(expiryDate)
        .signWith(SignatureAlgorithm.HS512, getMapConfigJWT().getValue())
        .claim("email", userPrincipal.getEmail())
        .claim("idUser", userService.findByEmail(userPrincipal.getEmail()).getId())
        .compact();
}
```

Fonte: O autor

A partir de um objeto do tipo `authentication` será extraído o usuário para manipulação. O método irá retornar um Token com as seguintes informações, Id(Número Identificador) do Usuário, data de expiração do Token, o segredo de criptografia(no projeto o segredo é gerado automaticamente toda vez que o software é inicializado, e salvo no banco de dados, o objeto `getValue()` de `getMapConfigJWT` faz acesso ao banco de dados para retornar esse segredo), por fim são inseridos dois outros parâmetros, email do usuário e Id do usuário

Prosseguindo com as Classes essenciais para o Spring Security, é necessário o desenvolvimento de uma Classe que implemente a Interface `AuthenticationEntryPoint`. No Projeto essa Classe é a `JWTAuthenticationEntryPoint`, cuja função é retornar erros caso os clientes tentem acessar um recurso sem a devida autorização. A Figura 40 apresenta a implementação dessa Classe.

**Figura 40. Classe JWTAuthenticationEntryPoint**

```

@Component
public class JwtAuthenticationEntryPoint implements AuthenticationEntryPoint {

    private static final Logger logger = LoggerFactory.getLogger(JwtAuthenticationEntryPoint.class);

    @Override
    public void commence(HttpServletRequest request, HttpServletResponse response,
        AuthenticationException authException) throws IOException, ServletException {

        logger.error("Respondendo com erro de Usuário em autorização. Mensagem - {}" + authException.getMessage());
        response.sendError(HttpServletResponse.SC_UNAUTHORIZED,
            "Desculpe mas você não tem autorização para acessar esse recurso");

    }

}

```

Fonte: O autor

A última Classe que serve como Base para o Spring Security é JWTAuthenticationFilter que tem as seguintes funções:

1. Ler o Token do Cabeçalho Authorization das Requisições
2. Validar o Token
3. Carregar os detalhes do Usuário associados ao Token
4. Inserir os Detalhes do Usuário dentro do Contexto de Segurança do Spring, para que o Spring possa fazer as checagens de segurança.

A seguir, na Figura 41 é apresentado o método principal da Classe de JWTAuthenticationFilter. Método esse responsável por executar as funções mencionadas anteriormente.

**Figura 41. Método FilterInternal de JWTAuthenticationFilter**

```

@Override
protected void doFilterInternal(HttpServletRequest request, HttpServletResponse response, FilterChain filterChain)
    throws ServletException, IOException {

    try {
        String jwt = getJwtFromRequest(request);

        if (StringUtils.hasText(jwt) && tokenProvider.validateToken(jwt)) {
            Integer userId = tokenProvider.getUserIdFromJWT(jwt);
            UserDetails userDetails = customUserDetailsService.loadByUserId(userId);
            UsernamePasswordAuthenticationToken authentication = new UsernamePasswordAuthenticationToken(
                userDetails, null, userDetails.getAuthorities());
            authentication.setDetails(new WebAuthenticationDetailsSource().buildDetails(request));

            SecurityContextHolder.getContext().setAuthentication(authentication);
        }
    } catch (Exception e) {
        logger.error("Could not set user authentication in security context", e);
    }

    filterChain.doFilter(request, response);
}

```

Fonte: O autor

Além das Classes mencionada anteriormente. Existe a necessidade do desenvolvimento de uma Classe com as configurações de Segurança. Essa Classe deverá estender a Classe ‘WebSecurityConfigurerAdapter’ como o mostrado na Figura 42 .

**Figura 42. Diagrama exemplificando Implementação de Segurança**

```
@Configuration
@EnableWebSecurity
@EnableGlobalMethodSecurity(securedEnabled = true, jsr250Enabled = true, prePostEnabled = true)
public class SecurityConfig extends WebSecurityConfigurerAdapter {
```

Fonte: O autor

Também foi necessária a utilização de algumas anotações nessa Classe: `@Configuration` indicando que a Classe é uma Classe de configuração, `@EnableWebSecurity` anotação utilizada para desativar a utilização de autenticação e das configurações de segurança Padrão do Spring e utilizar uma própria definição de segurança e a Última anotação `@EnableGlobalMethodSecurity` com parâmetros necessários para Habilitar modulo de segurança nos métodos do Projeto, e sinalizando que podemos utilizar anotações para verificar as autorizações na definição dos métodos.

É nesta mesma classe onde são efetuadas as parametrizações para liberação de CORS(Cross Origin Requests) em português, requisições vindas de origens adversas. É um recurso que barra as requisições de origens que não sejam o próprio ambiente de execução, deixando o CORS habilitado, não é possível fazer com que os Serviços do FrontEnd se comuniquem com o BackEnd. A Figura 43 mostra a configuração utilizada no Projeto.

**Figura 43. Bean para Configuração de Cors**

```
@Bean
CorsConfigurationSource corsConfigurationSource() {

    CorsConfiguration corsConfiguration = new CorsConfiguration().applyPermitDefaultValues();

    corsConfiguration.setAllowedMethods(Arrays.asList("POST", "GET", "PUT", "DELETE", "OPTIONS"));

    final UrlBasedCorsConfigurationSource source = new UrlBasedCorsConfigurationSource();

    source.registerCorsConfiguration("/**", corsConfiguration);

    return source;
}
```

Fonte: O autor

O método Principal da Classe SecurityConfig é o método configure mostrado na Figura 44. Além desse método indicar que haverá parametrização de CORS no Software, ele também é responsável por realizar a liberação de rotas Http que não precisam de proteção por autenticação, por exemplo a rota para cadastro de novo usuário.

**Figura 44. Método Configure da Classe SecurityConfig**

```
@Override
protected void configure(HttpSecurity http) throws Exception {
    http.cors().and().csrf().disable().exceptionHandling().authenticationEntryPoint(unauthorizedHandler).and()
        .sessionManagement().sessionCreationPolicy(SessionCreationPolicy.STATELESS).and().headers()
        .frameOptions().sameOrigin().and().authorizeRequests()
        .antMatchers("/", "/favicon.ico", "/*/*.png", "/*/*.gif", "/*/*.svg", "/*/*.jpg", "/*/*.html",
            "/*/*.css", "/*/*.js")
        .permitAll().antMatchers("/login/**", "/h2-console/**").permitAll()
        .antMatchers(HttpMethod.GET, "/ceps/buscapep/**", "/configs/buscacrypto", "user/buscauser/**",
            "pessoa/buscapessoa/id/**")
        .permitAll().antMatchers(HttpMethod.POST, "/pessoa/inserepf", "/pessoa/inserepj", "/api/**").permitAll()
        .anyRequest().authenticated();

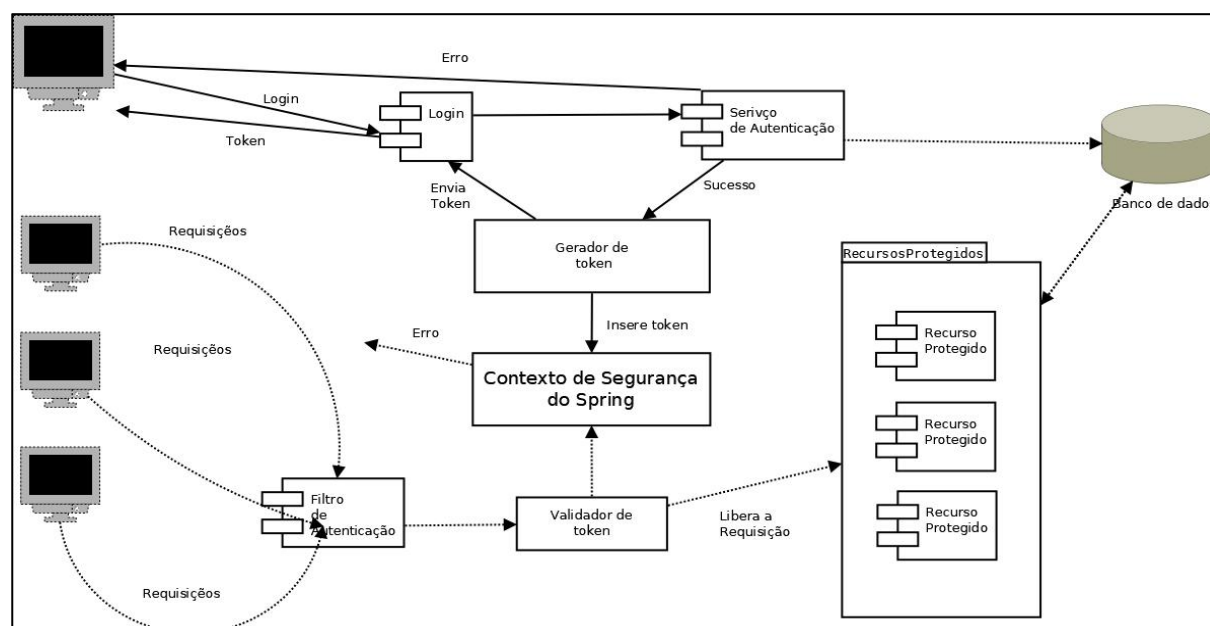
    // Add our custom JWT security filter
    http.addFilterBefore(jwtAuthenticationFilter(), UsernamePasswordAuthenticationFilter.class);
}
```

Fonte: O autor

### 3.5.1. Visão Geral - Segurança

A Figura 45 ilustra de maneira básica, uma visão geral sobre o funcionamento do modulo de segurança dentro da aplicação.

**Figura 45. Diagrama exemplificando Implementação de Segurança**



Fonte: O autor

Todas as requisições realizadas, que estejam parametrizadas com liberação na classe SecurityConfig, são recebidas diretamente pelos Controladores e repassadas as Classes de serviço. Um exemplo é o método de Login, ele precisa ser liberado, pois um usuário que ainda não efetuou Login, ainda não possui o Token de Autenticação e ainda também não se encontra no contexto de Segurança.

Efetuada o método de Login, caso as credenciais estejam corretas, será retornado o token de autenticação do usuário, e o Usuário será inserido no Contexto de Segurança. Caso ocorra algum problema no Login, o JWTAuthenticationEntryPoint irá lançar um erro ao usuário.

Nos demais casos de métodos que possuem rotas protegidas todas as requisições irão sofrer o mesmo processo.

Elas antes de acessar o método das Classes Controladoras serão filtradas pelo JWTAuthenticationFilter, essa classe lerá o cabeçalho Authorization dos Token. Extraíra suas informações e solicitará validação do Token para a Classe JWTTokenProvider. Caso exista algum problema com o Token, automaticamente irá ser retornado um erro ao usuário. Se o Token for válido, a classe irá inserir a autenticação dentro do Contexto de Segurança, caso ela ainda não esteja armazenada.

Feito esse processo, a requisição chegará dentro das Classes Controladoras, onde antes do método ser executado, é realizada a última validação. É validada a se o usuário possui a permissão para acessar o recurso, utilizando a anotação @PreAuthorize conforme mostrado na Figura 46. Na Figura, é apresentado o método para alteração do cadastro de uma pessoa, mas para acessar essa rota precisa ter a autorização 'ROLE\_UPDATE\_PESSOA'.

**Figura 46. Utilização da Anotação @PreAuthorize**

```
@PreAuthorize("hasRole('ROLE_UPDATE_PESSOA' ) or hasRole('ROLE_ADMIN')")
@RequestMapping(value="/update", method = RequestMethod.PUT)
public ResponseEntity<Pessoa> updatePessoa(@RequestBody Pessoa updatePessoa){
    Pessoa pessoa = pessoaService.updatePessoa(updatePessoa);
    return ResponseEntity.ok(pessoa);
}
```

Fonte: O autor



### 3.6. Visão geral do Sistema

No presente subcapítulo serão apresentadas algumas funcionalidades do Software desenvolvido.

A primeira funcionalidade a ser apresentada é a tela de Login ,Figura 47, na tela é mostrado o Logo do Software, e os campos de login e senha para acesso. Um dos botões tem a funcionalidade de transmitir os dados para a tentativa de login, e o outro direciona o usuário à página de página de cadastro.

**Figura 47. Tela de Login**

Fonte: O autor

Ao clicar no botão de cadastro, será apresentada a Página para inserção das informações básicas de cadastro, Figura 48, dependendo do tipo de pessoa escolhido (Pessoa Física ou Pessoa Jurídica). Caso seja escolhido Pessoa Física, terá que ser informado CPF, nome, Data de Nascimento e Sexo, caso a Pessoa informada seja Jurídica terá que ser informado CNPJ e razão Social. Os *Cards* de endereço e Usuário tem que ser cadastrados obrigatoriamente.

**Figura 48. Tela de Cadastro**

← Cadastro

Cadastro Pessoa

Tipo de Pessoa

Pessoa Física ☐

Pessoa Jurídica ☒

Razão Social

CNPJ

Cadastro de Endereço

CEP

Número

Complemento

Cadastro do Usuário

Email

Senha

CRIAR CONTA

Fonte: O autor

No menu Principal, localizado à esquerda, são apresentadas todas as Páginas principais, conforme mostrado na Figura 49.

**Figura 49. Menu Principal**

Menu

Profile

Rota ☒

Gerar nova Rota

Rotas Geradas

Pessoa

Endereco ☒

Empresa ☒

Funcionario ☒

Cargos

Ceps

Usuários

Regiao

Logout

Fonte: O autor



A opção Rotas Geradas, apresenta a lista de Todas as rotas já geradas pela empresa, caso o usuário esteja vinculado a uma empresa. Ou todas as rotas que o usuário gerou, caso ele seja uma Pessoa Física sem vínculo com nenhuma empresa, Figura 50. Caso o usuário clique em cima de uma das rotas geradas, ele será direcionado à página com o detalhamento do Rota, Figura 51, que mostra o botão para abrir a rota no Google Maps, os pontos de entrega, e os pontos não atendidos pela empresa, esse campo é alimentado quando no ato de geração de rotas, caso o usuário possua vínculo com alguma empresa, e a empresa possua a parametrização por regiões não atendendo algum cep em específico.

**Figura 50. Listagem de Rotas Criadas**

Rotas Criadas
Rotas Geradas
Id da Rota: 4 Data de Criação da Rota: 15/09/2018 Usuário Criador: adm@adm.com.br
Id da Rota: 5 Data de Criação da Rota: 15/09/2018 Usuário Criador: adm@adm.com.br
Id da Rota: 8 Data de Criação da Rota: 17/09/2018 Usuário Criador: adm@adm.com.br
Id da Rota: 9 Data de Criação da Rota: 17/09/2018 Usuário Criador: adm@adm.com.br
Id da Rota: 10 Data de Criação da Rota: 17/09/2018 Usuário Criador: adm@adm.com.br

Fonte: O autor

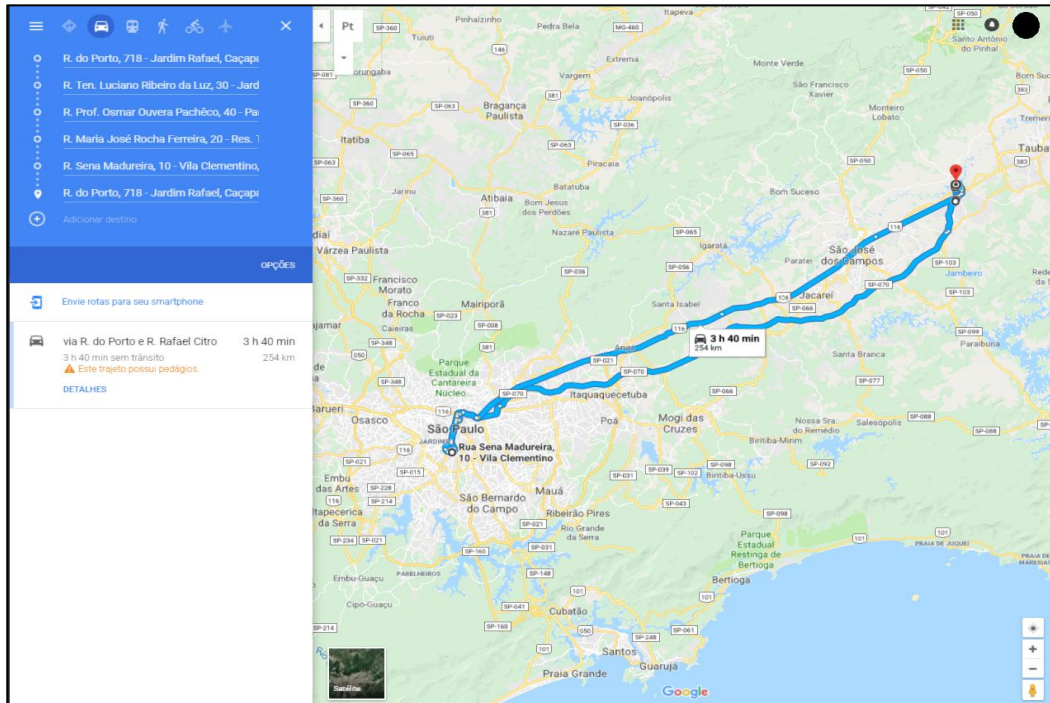
**Figura 51. Página de detalhamento da Rota**

←	Detalhes da Rota
<b>Informações</b> Identificação da Rota : 4 Usuário Criador : adm@adm.com.br Data de Criação : 15/09/2018	
Abrir no Google Maps	
<b>Endereços da Rota</b> CEP: 04021001. Nome Rua: Rua Sena Madureira. Bairro: Vila Clementino. Cidade:São Paulo. Estado: SP. Numero Logradouro: 10. Complemento: null CEP: 12285240. Nome Rua: Rua Maria José Rocha Ferreira. Bairro: Parque Residencial Maria Elmira. Cidade:Caçapava. Estado: SP. Numero Logradouro: 20. Complemento: null CEP: 12288430. Nome Rua: Rua Tenente Luciano Ribeiro da Luz. Bairro: Jardim Rafael. Cidade:Caçapava. Estado: SP. Numero Logradouro: 30. Complemento: null CEP: 12285130. Nome Rua: Rua Professor Osmar Ouverá Pacheco. Bairro: Parque Residencial Maria Elmira. Cidade:Caçapava. Estado: SP. Numero Logradouro: 40. Complemento: null	

Fonte: O autor

Quando clicar em abrir rota no Google Maps ele será direcionado ao Google maps com a Rota já sendo carregada automaticamente conforme Figura 52.

**Figura 52. Rota Criada Aberta no Google Maps**



Fonte: O autor

vogradouro para entrega, Figura 53. A medida que os ceps vão sendo inseridos, eles são listados, podendo ser removidos caso tenham sido inseridos erroneamente. Todos os ceps inseridos são validados antes de aparecerem na listagem. Ao final da Inserção dos ceps o usuário pode clicar no botão Gerar Rota para que sua rota seja calculada e o botão para abrir no Google Maps apareça, Figura 54.

**Figura 53. Página para Gerar a Rota**

Rota

CEP

NUM

ADICIONAR

GERAR ROTA

Endereços à serem roteirizados

CEP: 12288560 Numero: 321

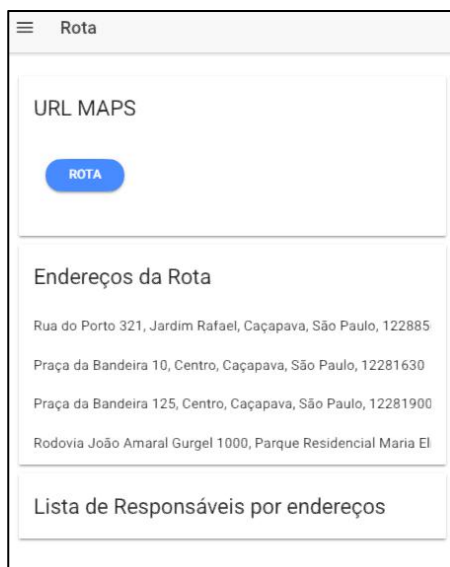
CEP: 12285020 Numero: 1000

CEP: 12281-630 Numero: 10

CEP: 12281-900 Numero: 125

Fonte: O autor

**Figura 54. Página Após Rota Ser Gerada**



Fonte: O autor

Neste exemplo o Campo Lista de Responsáveis por endereços ficou em branco pois, o usuário é Pessoa Física e não possui nenhum tipo de parametrização de Regiões.

Dentro do menu principal existe a opção de acessar o endereço cadastrado do usuário Figura 55, tendo a opção de realizar alteração, caso necessário.

**Figura 55. Página de Endereço**



Fonte: O autor

A Página de Empresas Figura 56, exibe as informações da Empresa e permite que seja acessado a Funcionalidade de Alteração da Empresa e Criar uma nova Empresa, nesta Opção, automaticamente a empresa Criada se torna uma Filial, da empresa do Usuário que está efetuando o cadastro. Outras opções apresentadas nesta Página é a exibição das Filiais da Empresa Figura 57, e também exibição dos seus Funcionários Figura 58.

**Figura 56. Página Empresa**

EMPRESA

Empresa - Info

Id Empresa: 13

Tipo da Pessoa : JURIDICA

Razão Social: Transp 1

CNPJ : 45678932112366

Tipo da Empresa: MATRIZ

Categoria : TRANSPORTADORA

Id Matriz:

Endereço

Cep: 12285020

Logradouro : Rodovia João Amaral Gurgel

Número: 1002

Complemento: Galpão A

Bairro: Parque Residencial Maria Elmira

Cidade: Caçapava

ALTERAR EMPRESA

CRIAR NOVA EMPRESA

EXIBIR FILIAIS

EXIBIR FUNCIONARIOS

Fonte: O autor

**Figura 57. Página de Filiais da Empresa**

FILIAIS

Filiais

Id da Empresa: 14

Nome da Empresa : Transp 1 A

Id da Empresa: 15

Nome da Empresa : Transp 1 B

Fonte: O autor

**Figura 58. Página de Listagem Funcionários da Empresa**

Fonte: O autor

Tanto na Listagem de Funcionários, quanto na Listagem das Filiais, quando é clicado em cima de um item da lista o usuário é redirecionado a pagina de detalhamento do mesmo, onde há todas as informações do item selecionado, seja empresa ou funcionário

A Página de Regiões, Figura 59 ,Apresenta as informações da região cadastrada para a empresa. Se o usuário necessitar, também poderá exibir todos os Ceps dessa região.

**Figura 59. Página de Região**

**REGIÃO**

**INFORMAÇÕES DA REGIÃO CADASTRADA**

Identificação: 2  
 Cadastrado para : Transp 1  
 Descrição da Rota : TESTE INSERÇÃO

Cep: 12209060  
 Logradouro: Rua Tenente Manuel Pedro de Carvalho  
 Bairro: Vila Santa Helena  
 Cidade : São José dos Campos. Estado : São Paulo

Cep: 12209310  
 Logradouro: Rua Felisbino Pinto da Cunha  
 Bairro: Vila Nova São José  
 Cidade : São José dos Campos. Estado : São Paulo

Cep: 04021001  
 Logradouro: Rua Sena Madureira  
 Bairro: Vila Clementino  
 Cidade : São Paulo. Estado : São Paulo

Cep: 12216300  
 Logradouro: Rua Opala  
 Bairro: Jardim São José Centro  
 Cidade : São José dos Campos. Estado : São Paulo

Cep: 12209010  
 Logradouro: Avenida São José  
 Bairro: Jardim Bela Vista  
 Cidade : São José dos Campos. Estado : São Paulo

**ESCONDER CEPS**

**ALTERAR REGIÃO**

Fonte: O autor

Quando o usuário Clicar em Alterar Região ele será levado para a Página de alteração de Região, Figura 60. Essa Página contém diversas opções, podendo remover todos os Ceps da região, ou remover alguns Ceps desejados. Outra opção é adicionar mais ceps a Região, os ceps para essa adição podem buscados a partir de uma cidade, ou de um bairro de uma cidade. O usuário pode adicionar todos os Ceps dessa busca, ou apenas os que ele desejar. Realizando todas as alterações desejadas ,o usuário pode clicar em salvar para efetuar a alteração na Região.

**Figura 60. Página para Alterar a Região.**

← ALTERAÇÃO DE REGIÃO

TESTE INSERÇÃO

CEPS DA REGIÃO

CEP : 12209060  
Nome rua : Rua Tenente Manuel Pedro de Carvalho  
-

CEP : 12209310  
Nome rua : Rua Felisbino Pinto da Cunha  
-

CEP : 04021001  
Nome rua : Rua Sena Madureira  
-

CEP : 12216300  
Nome rua : Rua Opala  
-

CEP : 12209010  
Nome rua : Avenida São José  
-

REMOVER TODOS

Adição de Ceps

Selecionar modo de Busca Busca por Estado e Cid... ▾

Estado São Paulo ▾

Cidade Caragatatuba ▾

CEP

Cep : 11660070  
Nome do logradouro : Rua Guarulhos  
+

ADICIONAR TODOS

NOVA BUSCA SALVAR ALTERAÇÕES

A última funcionalidade a ser apresentada é a de Gestão dos Usuários. Ao clicar em usuários no Menu Principal, será listado todos os usuários da empresa, Figura 61.

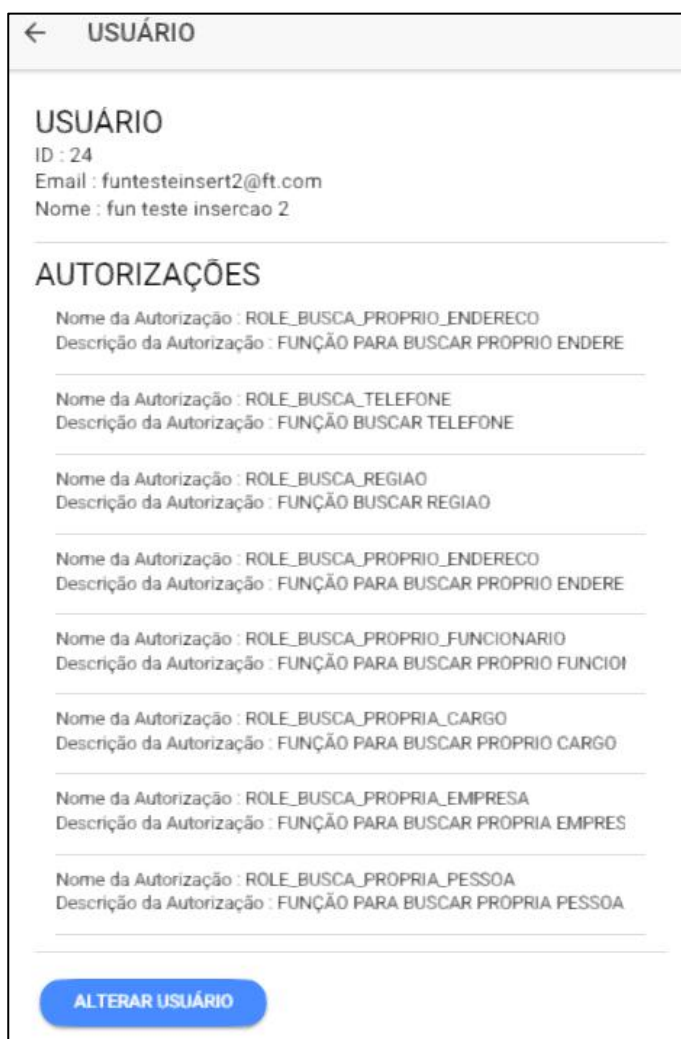
**Figura 61. Página para Alterar a Região.**



Fonte: O autor

Ao clicar no Funcionário desejado o usuário é direcionado a Pagina de detalhamento do usuário, onde é exibido todas as permissões que o usuário possui, Figura 62, e também é apresentada a opção de alteração do usuário.

**Figura 62. Página de Detalhamento do Usuário.**



Fonte: O autor

Caso o usuário clique em alterar será aberta uma pagina que listará todas as permissões do usuário, que podem ser tiradas clicando no ícone de remover, e todas as permissões disponíveis para serem aplicadas ao usuário, que podem ser aplicadas clicando no ícone de adição, Figura 63. Ao realizar as alterações desejadas o usuário poderá clicar em Salvar alteração, Figura 64.

**Figura 63. Página de Alteração do Usuário.**



Nome da Autorização : ROLE\_BUSCA\_PROPRIA\_CARGO  
Descrição da Autorização : FUNÇÃO PARA BUSCAR PRÓPRIO CARGO  
-

Nome da Autorização : ROLE\_BUSCA\_PROPRIA\_EMPRESA  
Descrição da Autorização : FUNÇÃO PARA BUSCAR PRÓPRIA EMPRES  
-

Nome da Autorização : ROLE\_BUSCA\_PROPRIA\_PESSOA  
Descrição da Autorização : FUNÇÃO PARA BUSCAR PRÓPRIA PESSOA  
-

**AUTORIZAÇÕES DO USUARIO**

Nome da Autorização : ROLE\_ADMIN  
Descrição da Autorização : FUNÇÃO DE ADMINSTRADOR DO SISTEM/  
+

Nome da Autorização : ROLE\_CREATE\_CARGO  
Descrição da Autorização : FUNÇÃO DE CRIAR CARGOS  
+

Nome da Autorização : ROLE\_DELETE\_CARGO  
Descrição da Autorização : FUNÇÃO DE DELETAR CARGOS  
+

Nome da Autorização : ROLE\_BUSCA\_CARGO  
Descrição da Autorização : FUNÇÃO DE BUSCAR CARGOS  
+

Fonte: O autor

**Figura 64. Botões para Salvar e Cancelar alteração do Usuário.**



Nome da Autorização : ROLE\_BUSCA\_USERS  
Descrição da Autorização : FUNÇÃO PARA BUSCAR USUÁRIOS  
+

Nome da Autorização : ROLE\_BUSCA\_ROLES  
Descrição da Autorização : FUNÇÃO PARA BUSCAR AUTORIZAÇÕES  
+

**SALVAR ALTERAÇÃO** **CANCELAR ALTERAÇÃO**

Fonte: O autor

Essas foram as principais funcionalidades desenvolvidas no Software.



## **4. VALIDAÇÃO E ANÁLISE DOS DOS RESULTADOS OBTIDOS**

### **4.1. Métricas do sistema**

### **4.2. Técnicas de Verificação e Validação aplicadas**

### **4.3. Resultados de Verificação e Validação**

## **5. CONCLUSÃO**

### **5.1. Principais contribuições**

### **5.2. Considerações Gerais, Limitações e Dificuldades**

### **5.3. Sugestões de trabalhos futuros**

## REFERÊNCIAS BIBLIOGRÁFICAS

**A MELHOR de cada Segmento.** Revista As Melhores do Transporte. Editora OTM, ano 14, no 14, novembro 2001.

BALLOU, R. H. **Gerenciamento da cadeia de suprimentos: Logística empresarial.** 5 ed. Porto Alegre, Bookman, 2006.

BRANSKI, R. M. **O papel da tecnologia da informação no processo logístico:** estudo de caso com operadores logísticos. 2008. 252 f. Tese (Doutorado em Engenharia) – Escola Politécnica, Universidade de São Paulo, São Paulo

CENTRO DE ESTUDOS EM LOGÍSTICA –CEL/COPPEAD. **Panorama Logístico – Gestão do Transporte Rodoviário de Cargas nas Empresas - Práticas e Tendências,** 2007.

CHOPRA, Sunil; MEINDL Peter. **Gestão da Cadeia de suprimentos: Estratégias, Planejamentos e Operações.** 4ª Ed. São Paulo: Pearson, 2011.

CHOPRA, Sunil. MEINDL, Peter. **Gerenciamento da cadeia de suprimento: Estratégia, planejamento e operação.** São Paulo: Prentice Hall, 2003.

Demaria, Marjory. **"O operador de transporte multimodal com fator de otimização da logística."** (2004).

DORNIER, Philippe-Pierre. ERNST, Ricardo. FENDER, Michel. KOUVELIS, Panos. **Logística e operações globais. Textos e casos.** São Paulo: Atlas, 2000.

FLEURY, Paulo F. **Vantagens competitivas e estratégicas no uso de operadores logísticos.** Revista TecnoLogística, São Paulo, ano V, n. 46, set. 1999.

FLEURY, Paulo F. **Vantagens Competitivas e Estratégicas no Uso de Operadores Logísticos. Logística Empresarial: a perspectiva brasileira.** Ed. Atlas S.A., São Paulo, 2000.

FRANCISCHINI, P.G.; AMARAL GURGEL, F. **Administração de materiais e do patrimônio.** São Paulo: Pioneira Thomson, 2002.

MACHLINE, C. **Cinco décadas de logística empresarial e administração da cadeia de suprimentos no Brasil.** Rev. adm. empres. vol.51 no.3 São Paulo May/June 2011. Disponível em: [http://www.scielo.br/scielo.php?pid=S0034-75902011000300003&script=sci\\_arttext](http://www.scielo.br/scielo.php?pid=S0034-75902011000300003&script=sci_arttext). Acesso em: 29 de mar 2017

MATOS JUNIOR, Carlos Alberto de et al. **O papel da roteirização na redução de custos logísticos e melhoria do nível de serviço em uma empresa do segmento alimentício no Ceará.** In: Anais do Congresso Brasileiro de Custos-ABC. 2013.

MARQUES, Keise de Leone. **Back-end vs Front-end vs Full-Stack: qual é a melhor escolha?**. Disponível em: <https://becode.com.br/back-end-front-end-full-stack/> . Acesso em: 06/09/2018

NAZÁRIO, P. **A importância de sistemas de informação para a competitividade logística.** Rio de Janeiro: Centro de Estudos em Logística, Coppead, 1999.

PERÇIN ,S.; MIN, H. A hybrid quality function deployment and fuzzy decision-making methodology for the optimal selection of third-party logistics service providers. **International Journal of Logistics: Research and Applications**, [S1], v. 16, n. 5, p.380-397 - 2013.

POZO, H. **Administração de recursos materiais e patrimoniais: uma abordagem logística.** 6ª Ed. São Paulo: Atlas, 2010

RIBEIRO, Priscilla Cristina Cabral;Ferreira, Karine Araújo . **Logística e transportes: uma discussão sobre os modais de transporte e o panorama brasileiro.** XXII Encontro Nacional de Engenharia de Produção (2002).

ROMERO Monica, SOUZA Dario. **Gerenciamento da cadeia de suprimentos.** Revista Científica Emersão v.1, nº 1 – maio/2015 – p. 146-155 Porto Belo/ SC

ROSA, Adriano Carlos. **Gestão do transporte na logística de distribuição física: uma análise da minimização do custo operacional.** 2007. Tese de Doutorado. Dissertação (Mestrado). Departamento de Economia, Contabilidade e Administração, Universidade de Taubaté, SP, Brasil.

THAYER, Richard; DORFMAN, Merlin. **System and Software Requirements Engineering - Second Edition.** Los Alamitos: IEEE Computer Society Press Tutorial, 2000. 528p

SOMMERVILLE, I. **Engenharia de software.** Tradução: Ivan Bosnic e Kalinka G. O. Gonçalves; Revisão técnica: Kechi Hirama. 9 ed. São Paulo: Pearson Prentice Hall, 2011.

UDACITY. Conheça as linguagens de programação mais utilizadas no Brasil e no Mundo - Disponível em: <https://br.udacity.com/blog/post/linguagens-de-programacao-mais-usadas-no-brasil-e-no-mundo> Acesso em: 05/09/2018

FOWLER, Martin. **Inversion of Control Containers and the Dependency Injection pattern** - Disponível em: <https://martinfowler.com/articles/injection.html> . Acesso em: 05/09/2018