

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

**ANIRO CORREIA MONTENEGRO**

**PROJETO PARA CRIAÇÃO DE UMA APLICAÇÃO PARA  
DESPACHO DE OCORRÊNCIAS A DEFESA CIVIL DE SÃO  
JOSE DOS CAMPOS**

São José dos Campos  
2018

**ANIRO CORREIA MONTENEGRO**

**PROJETO PARA CRIAÇÃO DE UMA BASE DE DADOS PARA  
A DEFESA CIVIL DE SÃO JOSE DOS CAMPOS**

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

**Orientador: Me. Eduardo Sakaue**

São José dos Campos  
2018

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

MONTENEGRO, Aniro Correia  
Projeto para Criação de uma Base de Dados para a Defesa Civil de São Jose Dos Campos.  
São José dos Campos, 2018.  
50f.

Trabalho de Graduação – Curso de Tecnologia em Banco de dados  
FATEC de São José dos Campos: Professor Jessen Vidal, 2018.  
Orientador: Me Eduardo Sakaue.

1. Áreas de conhecimento. I. Faculdade de Tecnologia. FATEC de São José dos Campos:  
Professor Jessen Vidal. Divisão de Informação e Documentação. II. Título

**REFERÊNCIA BIBLIOGRÁFICA –**

MONTENEGRO, Aniro Correia. **Projeto para criação de uma Base de Dados para a Defesa Civil de São Jose dos Campos.** 2018. 50f. Trabalho de Graduação - FATEC de São José dos Campos: Professor Jessen Vidal.

**CESSÃO DE DIREITOS –**

NOME DO AUTOR: Aniro Correia Montenegro

TÍTULO DO TRABALHO: Projeto para criação de uma Base de Dados para a Defesa Civil de São Jose dos Campos

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.

---

Aniro Correia Montenegro  
Rua Jamil Cury nº101 Bloco 03 Ap32, Vila Industrial  
CEP:12.221-520-São José dos Campos-SP

**ANIRO CORREIA MONTENEGRO**

**PROJETO PARA CRIAÇÃO DE UMA BASE DE DADOS PARA  
A DEFESA CIVIL DE SÃO JOSE DOS CAMPOS**

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

**Composição da Banca**

---

**Valter João de Souza, Dr., FATEC Professor Jessen Vidal**

---

**Carlos Augusto Lombardi Garcia, Me., FATEC Professor Jessen Vidal**

---

**Eduardo Sakaue, Me., FATEC Professor Jessen Vidal**

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

**11/12/2018**

## RESUMO

Com o avanço da tecnologia todos os setores públicos de prestação de serviços essenciais à população tendem a fazer uso de recursos tecnológicos e a Defesa Civil necessita realizar esta atualização no modo de execução dos trabalhos. Apesar de prestar serviços de prevenção, mitigação, preparação de resposta a desastres, socorro a população afetada e reabilitação de cenários de desastre entre outros, muito da comunicação necessária para realizar estes trabalhos são feitas usando relatórios de preenchimento manual. O objetivo deste trabalho é criar uma aplicação para gerenciamento automatizado das solicitações feitas à Defesa Civil. Consequentemente os funcionários ganharão tempo nos atendimentos, os administradores poderão ter maior controle do serviço que está sendo prestado e os relatórios de atendimentos serão encaminhados com maior rapidez aos seus solicitantes.

**Palavras-Chave:** defesa civil; java; web service; ionic; despacho de ocorrências

## ABSTRACT

With the advance of technology all public sectors that provide essential services to the population tend to make use of technological resources and the Civil Defense needs to carry out this update in the execution mode of the works. Despite providing disaster prevention, mitigation, disaster preparedness, relief to the affected population and rehabilitation of disaster scenarios, among others, much of the communication required to perform these jobs is done using manual fill reports. The objective of this work is to create an application for automated management of requests made to Civil Defense. Consequently, employees will gain time from atendimento, administrators may have greater control over the service being provided and the reports will be forwarded more quickly to your applicants.

**Palavras-Chave:** civil defense; java; web service; ionic; dispatch of occurrences

## LISTA DE FIGURAS

Figura 1- Diagrama de acesso ao sistema.....	13
Figura 2-Diagrama de Caso de uso- Perfis Cadastrador e Agente .....	14
Figura 3-Diagrama de Caso de uso- Perfil Administrador .....	15
Figura 4-Protótipo de tela- Acesso ao Sistema.....	20
Figura 5-Protótipo de tela- Cadastro de Nova Ocorrência. ....	21
Figura 6-Protótipo de tela- Tela de criação de relatório. ....	22
Figura 7- Protótipo de tela- Tela de Registros Fotográficos. ....	23
Figura 8-Protótipo de tela- Tela de Criação de Relatório em formato PDF.....	23
Figura 9- Diagrama do modelo MVC- Backend. ....	24
Figura 10-Diagrama do modelo MVC- Frontend.....	25
Figura 11-Modelo Lógico.....	26
Figura 12-Fragmento pom.xml.corrigir .....	28
Figura 13-Diagrama de Classes .....	29
Figura 14-Estrutura arquivos aplicação mobile .....	30
Figura 15- EnderecoDTO .....	30
Figura 16- Insere Relatório frontend .....	31
Figura 17- Relatório Service .....	31
Figura 18- Relação objeto/relacional .....	33
Figura 19- DepartamentoRepository .....	34
Figura 20- Busca de relatórios por funcionário .....	35
Figura 21- OcorrenciaService – Insert.....	35
Figura 22- Tabela Url_Foto. ....	36
Figura 23-UploadPicture –Função do frontend. ....	36
Figura 24-UploadPicture – Função do backend. ....	37
Figura 25- Controle para Lista Ocorrências Abertas.....	38
Figura 26- JSON de Ocorrências Abertas .....	38
Figura 27- Fluxograma Controle de Acesso.....	40
Figura 28- Resposta Login.....	40
Figura 29- Senha Criptografada com BCrypt.....	41
Figura 30- Retorno de acesso negado. ....	42
Figura 31-Tela Insere Novo Funcionário .....	44
Figura 32-Alerta de permissão negada .....	45
Figura 33-Telas insere rua .....	46
Figura 34-Atualização de relatório .....	48
Figura 35-Tela de Despacho de ocorrência .....	49
Figura 36- Trabalhos Futuros.....	50

## Lista de Tabelas

Tabela 1-Caso de Uso 01- Efetuar acesso .....	16
Tabela 2-Caso de Uso 02 - Alterar senha .....	16
Tabela 3-Caso de Uso 03 - Recuperar Senha de Usuário.....	17
Tabela 4-Caso de Uso 04 - Cadastrar Novo Usuário.....	17
Tabela 5-Caso de Uso 05 - Inserir ocorrência .....	18
Tabela 6-Caso de Uso 06- Inserir Relatório .....	19
Tabela 7-Caso de Uso 07 - Registros Fotográficos do Relatório .....	19
Tabela 8-Caso de Uso 08 - Criação de Relatório em formato PDF .....	20
Tabela 9-Endpoints.....	41
Tabela 10- Testes de acesso ao sistema 1º Experimento .....	<b>Erro! Indicador não definido.</b>
Tabela 11- Testes de acesso ao sistema 1º Experimento .....	43
Tabela 12- Testes de acesso ao sistema 2º experimento.....	45
Tabela 13- Testes de acesso ao sistema 3º experimento.....	47

## LISTA DE ABREVIATURAS E SIGLAS

DC	Defesa Civil
MVC	<i>Model View Controller</i>
JSON	<i>JavaScript Object Notation</i>
REST	<i>Representational State Transfer</i>
URI	<i>Uniform Resource Identifier</i>
API	<i>Application Programming Interface</i>



## SUMÁRIO

<b>1.</b>	<b>INTRODUÇÃO .....</b>	<b>10</b>
1.1.	MOTIVAÇÃO.....	10
1.2.	OBJETIVO GERAL.....	11
1.3.	OBJETIVOS ESPECÍFICOS .....	11
1.4.	ESCOPO.....	11
<b>2-</b>	<b>LEVANTAMENTO DE REQUISITOS .....</b>	<b>12</b>
2.1	REQUISITOS RELACIONADOS AO ACESSO AO SISTEMA. ....	12
2.2	REQUISITOS RELACIONADOS À INSERÇÃO, CONTROLE E DESPACHO DAS SOLICITAÇÕES DE RELATÓRIO NO SISTEMA: .....	12
2.3	REQUISITOS RELACIONADOS AOS ATENDIMENTOS REALIZADOS PELOS AGENTES DE CAMPO NAS LOCALIDADES INDICADAS NAS SOLICITAÇÕES.....	12
2.4	DIAGRAMAS DE CASOS DE USO.....	13
2.5	DIAGRAMA DE CASO DE USO DE ACESSO AO SISTEMA PELOS USUÁRIOS COM PERFIS DE CADASTRADOR E AGENTE. ....	14
2.6	DIAGRAMA DE CASO DE USO DE ACESSO PELO USUÁRIO COM PERFIL ADMINISTRADOR. ....	15
2.7	DETALHAMENTO DOS CASOS DE USO. ....	15
2.7.1	ACESSO AO SISTEMA. ....	16
2.7.2	REQUISITOS RELACIONADOS À INSERÇÃO, CONTROLE E DESPACHO DAS SOLICITAÇÕES DE RELATÓRIO NO SISTEMA. ....	17
2.7.3	REQUISITOS RELACIONADOS AOS ATENDIMENTOS REALIZADOS PELOS AGENTES DE CAMPO NAS LOCALIDADES INDICADAS NAS SOLICITAÇÕES.....	18
2.10	PROTÓTIPOS DE TELA. ....	20
2.7.4	PROTÓTIPO DE TELA DE ACESSO AO SISTEMA. ....	20
2.7.5	PROTÓTIPO DE TELA DE CADASTRO DE NOVA OCORRÊNCIA. ....	21
2.7.6	PROTÓTIPO DE TELA DE CRIAÇÃO DE RELATÓRIO.....	22
2.7.7	PROTÓTIPO DE TELA DE REGISTRO FOTOGRÁFICO PARA O RELATÓRIO. ....	22
2.7.8	PROTÓTIPO DE TELA DE CRIAÇÃO DE RELATÓRIO EM FORMATO PDF.....	23
<b>3-</b>	<b>DESENVOLVIMENTO .....</b>	<b>24</b>
3.1	ARQUITETURA DO SISTEMA. ....	24
3.2	BANCO DE DADOS .....	26
3.3	ESTRUTURA DE CLASSES DO BACKEND DO PROJETO.....	27
3.4	ESTRUTURA DO FRONTEND DO PROJETO.....	29
3.5	PERSISTÊNCIA .....	32
3.6	MODEL .....	32
3.7	REPOSITORY.....	34
3.8	SERVICES .....	34
3.9	REGISTROS FOTOGRÁFICOS .....	35
3.10	SERVIÇOS.....	37
3.11	CONTROLE DE ACESSO.....	39
3.12	AUTENTICAÇÃO.....	40
3.13	AUTORIZAÇÃO .....	41
<b>4-</b>	<b>RESULTADOS .....</b>	<b>43</b>
4.1	- 1º EXPERIMENTO- ACESSO AO SISTEMA .....	43
4.2	- 2º EXPERIMENTO- SOLICITAÇÕES DE OCORRÊNCIAS.....	45
4.3	3º EXPERIMENTO- ATENDIMENTO E GERAÇÃO DE RELATÓRIOS PELOS AGENTES DE CAMPO.....	47
4.4	BATERIA DE TESTES:.....	47
<b>5</b>	<b>- TRABALHOS FUTUROS .....</b>	<b>50</b>

## 1. INTRODUÇÃO

A Defesa Civil de São José dos Campos<sup>1</sup> foi fundada em maio de 1983 e atua através de ações de prevenção, de mitigação, de resposta e recuperação com a finalidade de evitar ou minimizar desastres, tanto naturais como os de origem de ação humana. O grupo de agentes é formado por servidores e voluntários e são treinados para agirem em casos de inundações, soterramentos, quedas de barreira, desabamentos e incêndios.

A Defesa Civil é mantida pela Prefeitura Municipal, mas recebe apoio de instituições, organizações, empresas e da própria comunidade. Com grande incidência apoia o Corpo de Bombeiros nas ações de queimadas que ocorrem nas áreas vegetais do município.

Um dos trabalhos da Defesa Civil é enviar agentes a localidades de riscos que são apontadas através de solicitações feitas pessoalmente pelo munícipe, através dos telefones 156 e 190 ou através de solicitações feitas por outras autoridades como promotoria, bombeiros, etc.

### 1.1. Motivação

Na prestação de serviços à comunidade a Defesa Civil tem usado um sistema de gerenciamento de ocorrências onde varias etapas são feitas de forma manual. A solicitação para uma ocorrência pode ser solicitada pessoalmente, onde o munícipe comparece a Defesa Civil e solicita uma vistoria no local desejado, via telefone, ou através de solicitações de outros setores tanto da prefeitura como de outras esferas da administração pública.

Em todos os casos as solicitações são realizadas em formulários de papel, onde o funcionário anota todas as informações pertinentes ao atendimento. Feito o preenchimento do pedido de vistoria ou de algum apoio que a DC realiza, o formulário é encaminhando a um agente de campo que ficará responsável por realizar o serviço.

Ao realizar o serviço, o agente preenche um rascunho de forma manual e depois em um editor de texto realiza a elaboração de um relatório relacionado ao pedido da ocorrência. Se houver registros fotográficos, o agente descarrega as fotos de sua maquina fotográfica no computador onde está editando o relatório e insere de forma manual no texto que esta elaborando.

Pode-se notar que estas etapas consomem tempo e esforço dos funcionários, pois todo o serviço prestado não é informatizado.

Com base no contexto atual de atendimento podemos elencar como problemas principais os seguintes pontos:

1. Perda de tempo no preenchimento do formulário de solicitação.

---

<sup>1</sup> Disponível em: <https://www.sjc.sp.gov.br/servicos/protecao-ao-cidadao/defesa-civil/>

2. Perda de tempo na elaboração de relatórios.

## **1.2. Objetivo Geral**

O objetivo deste trabalho é o estudo do atual sistema de gerenciamento das solicitações feitas à Defesa Civil e a criação de uma ferramenta para substituir o atual processo que gerencie as solicitações recebidas pela Defesa Civil, distribuindo os serviços aos agentes de campo e permitindo a criação de relatórios de forma automatizada.

## **1.3. Objetivos específicos**

Como objetivos específicos deste trabalho, listamos os seguintes:

- O estudo do processo de solicitações atual da Defesa Civil.
- O estudo do processo de distribuição das requisições aos agentes de campo.
- O estudo do processo da elaboração de relatórios impressos e sua disponibilização para as partes interessadas.
- Levantamento de requisitos funcionais do sistema para a definição da estrutura do software.
- O estudo e aprofundamento das tecnologias usadas no desenvolvimento da aplicação.
- O desenvolvimento de um módulo servidor que ficará responsável em disponibilizar os serviços de comunicação entre a aplicação mobile e o banco de dados conforme as regras de negocio estabelecidas.
- O desenvolvimento de uma aplicação mobile que permita aos funcionários da DC realizarem abertura de ocorrência, elaboração de relatório com a possibilidade de registros fotográficos e a disponibilização dos relatórios impressos para as partes interessadas.
- Elaborar um banco de dados para persistência dos dados relacionados aos atendimentos que permita a recuperação e atualização dos dados no intuito de gerar relatórios de forma eficiente.
- Apresentar os resultados obtidos com as considerações devidas e possibilidades de trabalhos futuros.

## **1.4. Escopo**

Este trabalho restringe-se a análise e desenvolvimento da aplicação. Não faz parte deste estudo o levantamento de requisitos não funcionais.

## **2- LEVANTAMENTO DE REQUISITOS**

O levantamento de requisitos foi realizado através de reuniões com os diretores e agentes da Defesa Civil onde conseguimos informações sobre o funcionamento atual do trabalho realizado pelos funcionários do setor.

Foi necessário o uso de entrevistas e apresentação de prototipagem de telas para uma melhor compreensão dos requisitos.

Neste capítulo serão apresentados os requisitos de sistema, diagrama de casos de uso, diagramas UML e os protótipos de telas que auxiliarão na realização do desenvolvimento do sistema. Através do levantamento de requisitos realizado com os funcionários que trabalham no recebimento, classificação, despacho, atendimento a localidade e finalização da ocorrência foram levantados os seguintes requisitos:

### **2.1 Requisitos relacionados ao acesso ao sistema.**

R1- O usuário acessará o sistema através de credenciais que são *email* e uso de senha.

R2- A senha deverá ser armazenada usando algoritmo *hash*.

R3- O acesso às funcionalidades do aplicativo será disponibilizado conforme perfil do usuário.

R4- A aplicação permitirá a inserção de novos usuários através do perfil administrador.

### **2.2 Requisitos relacionados à inserção, controle e despacho das solicitações de relatório no sistema:**

R5- O sistema será capaz de receber solicitações de relatórios em formato único.

R6- Ao receber a solicitação o sistema disponibilizará as requisições para os agentes de campos, marcando o status do atendimento como atendimento aberto.

### **2.3 Requisitos relacionados aos atendimentos realizados pelos agentes de campo nas localidades indicadas nas solicitações.**

R7- O sistema permitirá que o agente realize o atendimento em dispositivo móvel tendo acesso às solicitações e enviando o relatório remotamente.

R8- O sistema permitirá que o agente preencha o relatório de ocorrência de forma automatizada no local.

R9- O sistema permitirá que o agente faça registro fotográfico do local para serem anexadas no relatório

R10- O sistema permitira a criação de relatório em formato PDF para assinatura manual e disponibilização de relatório para partes afins.

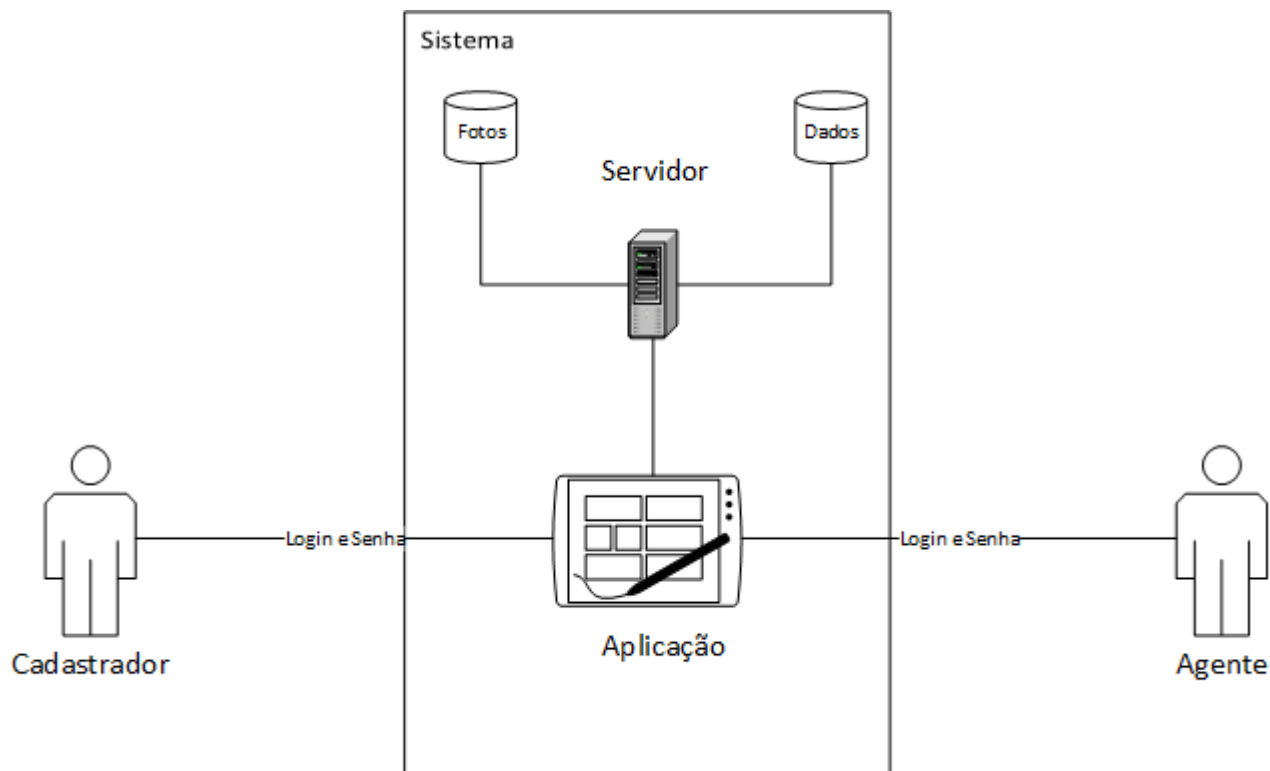
## 2.4 Diagramas de casos de uso.

Este tópico apresenta os diagramas de caso de uso sobre a interação dos usuários com o sistema. Estes diagramas apresentam cenários que mostram as funcionalidades do sistema do ponto de vista do usuário.

Mediante o levantamento de requisitos podemos definir como atores do sistema o usuário **agente** que realiza o atendimento no local indicado pela solicitação, o usuário **cadastrador** que será responsável pela inserção das requisições no sistema, monitoramento das ocorrências, impressão e guarda dos relatórios dos atendimentos e o usuário **administrador** que terá as funções do usuário agente mais a função de cadastrar novos usuários.

O cadastrador receberá o pedido de atendimento, que poderá ser feito de um departamento ou munícipe e realizará a inserção do mesmo no sistema.

**Figura 1- Diagrama de acesso ao sistema.**



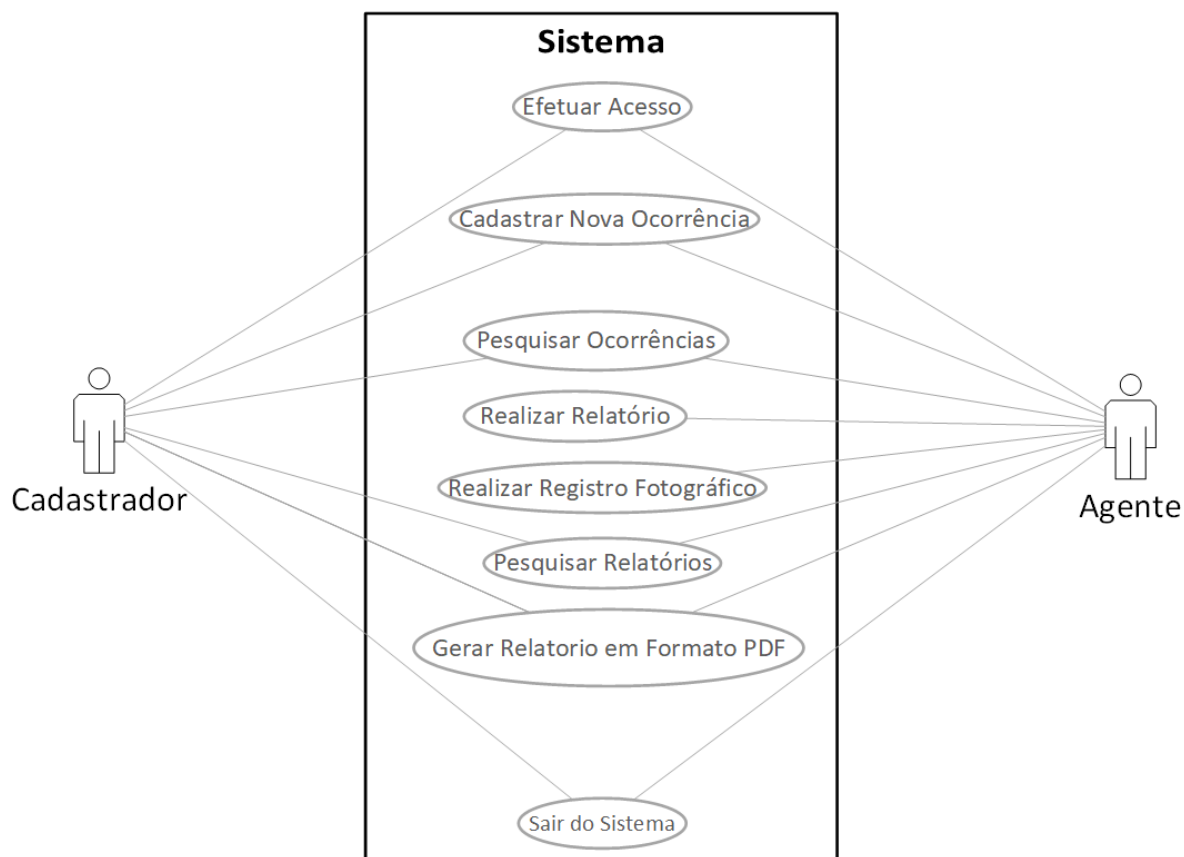
Fonte: Autor, 2018.

## 2.5 Diagrama de caso de uso de acesso ao sistema pelos usuários com perfis de cadastrador e agente.

O Cadastrador tem acesso ao sistema que lhe permite inserir no sistema uma nova ocorrência, realizar pesquisas sobre o andamento das ocorrências e gerar relatórios em PDF que serão assinados e disponibilizados para as partes interessadas.

O Agente tem o mesmo nível de acesso que o Cadastrador e também cria relatórios, faz registros fotográficos referentes aos relatórios.

**Figura 2-Diagrama de Caso de uso- Perfis Cadastrador e Agente**

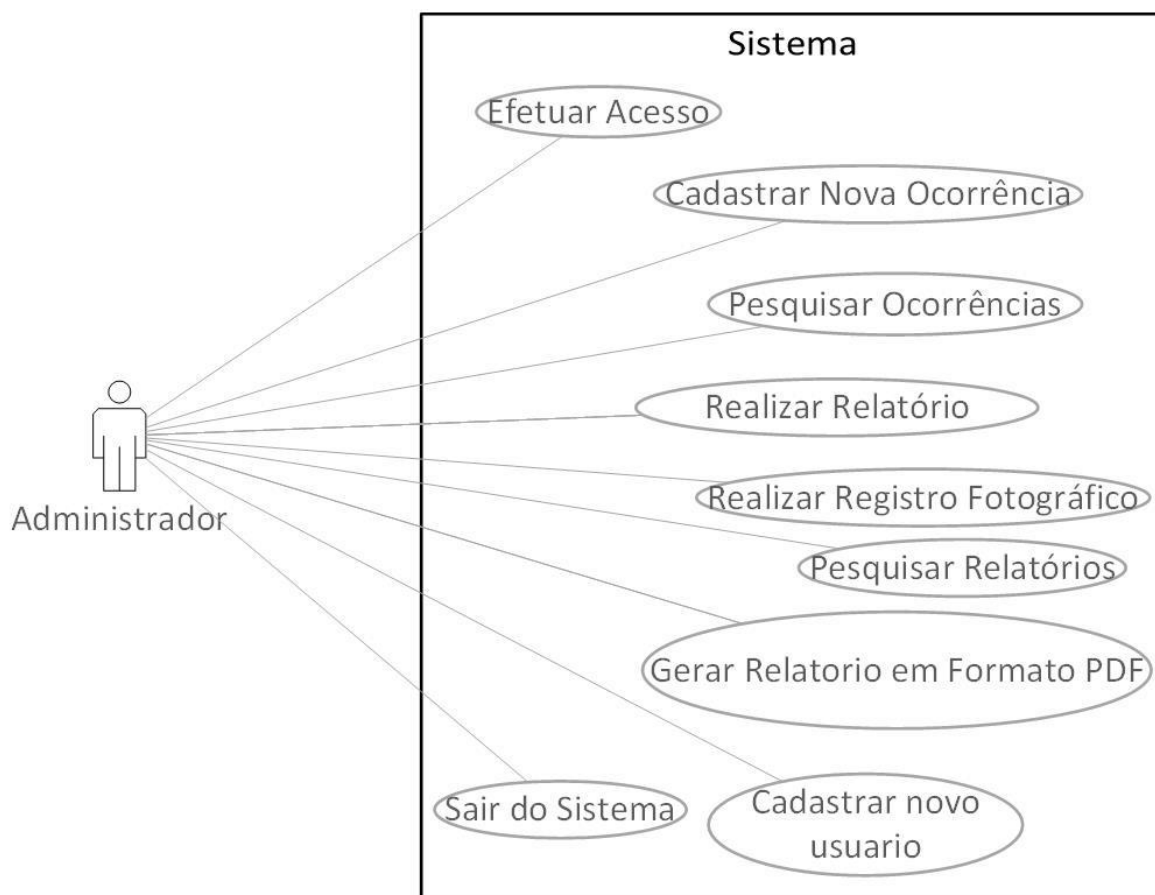


Fonte: Autor, 2018.

## 2.6 Diagrama de caso de uso de acesso pelo usuário com perfil administrador.

O Administrador tem o mesmo nível de acesso que o Agente e também insere novos usuários no sistema. Pode criar um novo usuário em qualquer perfil.

**Figura 3-Diagrama de Caso de uso- Perfil Administrador**



Fonte: Autor, 2018.

## 2.7 Detalhamento dos casos de uso.

Neste tópico serão apresentados os detalhamentos dos casos de usos. Conforme foram definidos os requisitos do sistema, o detalhamento dos casos de uso será dividido em casos de uso de acesso ao sistema, casos de uso de solicitações de relatórios, casos de uso dos agentes de campo e casos de uso dos relatórios de atendimentos.

### 2.7.1 Acesso ao sistema.

**Descrição:** Para o controle de acesso foi definido que o usuário deverá estar cadastrado no sistema possuindo um nome de *login* e senha.

**Tabela 1-Caso de Uso 01- Efetuar acesso**

Nome do caso de uso	Efetuar acesso
Atores	Agente/Cadastrador/Administrador
Resumo	Como agente ou funcionário realiza sua autenticação para acessar o sistema
Pré-condições	Estar cadastrado no sistema
Fluxo Principal	
Ações do ator	Ações do sistema
1- Acessar pagina inicial de autenticação	
2-Informar nome de usuário e senha	
3-Clicar em entrar	
	4- Realizar autenticação
	5- Redirecionar para tela principal
Restrições/Validações	1- Informar nome de usuário e senha corretos
Fluxo de Exceção- Informações de autenticação incorretas	
Ações do ator	Ações do sistema
	1-Usuario ou senha incorreta

**Tabela 2-Caso de Uso 02 - Alterar senha**

Nome do caso de uso	Alterar senha
Atores	Agente/Funcionário/Administrador
Resumo	Como usuário realizar troca de senha
Pré-condições	Estar cadastrado no sistema
Fluxo Principal	
Ações do ator	Ações do sistema
1- Acessar pagina inicial de autenticação	
2-Realizar login do sistema	
3-Na pagina Menu Principal clicar no botão Mudar Senha	
	4-Encaminhar usuário para pagina de troca de senha
5-Inserir nos campos o email cadastrado e nova senha e clicar em gravar	
	6- Gravar nova senha no banco e enviar email com nova senha para o usuário
	7-Retornar a pagina Menu Principal



**Tabela 3-Caso de Uso 03 - Recuperar Senha de Usuário**

Nome do caso de uso	Recuperar Senha de Usuário
Atores	Agente/Funcionário/Administrador
Resumo	Como usuário recuperar o acesso ao sistema
Pré-condições	Estar cadastrado no sistema
Fluxo Principal	
Ações do ator	Ações do sistema
1- Acessar pagina inicial de autenticação	
2- Clicar no botão Esqueci a Senha	
	1- Encaminhar usuário para pagina de solicitação de nova senha
2- Informar o email cadastrado no sistema e clicar no botão Enviar	
	3- Realizar mudança de senha e gravar no banco
	4- Enviar email para o usuário informando nova senha
	5- Retornar a pagina Menu Principal

**Tabela 4-Caso de Uso 04 - Cadastrar Novo Usuário**

Nome do caso de uso	Cadastrar novo funcionário
Ator	Administrador
Resumo	Como administrador cadastrar novo usuário
Pré-condições	Ser cadastrado com perfil de administrador
Fluxo Principal	
Ações do ator	Ações do sistema
1- Acessar pagina de cadastro de novo funcionário	
2- Inserir os dados do novo funcionário	
	3- Cadastrar novo funcionário no sistema

### **2.7.2 Requisitos relacionados à inserção e despacho das solicitações de relatório no sistema.**

Descrição: Para a inserção de solicitações é necessário que o usuário esteja logado no sistema. Todos os perfis poderão cadastrar novas ocorrências. Para realização de relatórios ficou definido que somente os agentes e administradores podem realizar relatório de ocorrência. Todos os funcionários poderão gerar relatórios em formato PDF para serem assinados pelos respectivos agentes que os elaboraram.

**Tabela 5-Caso de Uso 05 - Inserir ocorrência**

Nome do caso de uso	Inserir Ocorrência
Atores	Agente/Funcionário/Administrador
Resumo	Como usuário realizar inserção de nova ocorrência
Pré-condições	Estar logado no sistema
Fluxo Principal	
Ações do ator	Ações do sistema
1- Acessar pagina de inserção para nova ocorrência	
2- Preencher os campos conforme natureza da informação <ul style="list-style-type: none"> <li>a. Rua do local da ocorrência</li> <li>b. Origem da ocorrência</li> <li>c. Tipo da ocorrência</li> <li>d. Numero Local</li> <li>e. Histórico Inicial</li> <li>f. Tipo de solicitante</li> <li>g. Departamento</li> <li>h. Rua do solicitante</li> <li>i. Nome solicitante</li> <li>j. email solicitante</li> <li>k. Numero da residência do solicitante</li> <li>l. Numero telefone celular</li> <li>m. Numero de Telefone fixo</li> </ul>	
3- Clicar em gravar	
	4- Gravar a ocorrência com status “aberta”
Restrições/Validações	1- Ocorrência não pode ser gravada
Fluxo de Exceção- Informações de autenticação incorretas	
Ações do ator	Ações do sistema
	1- Informar que ocorrência não pode ser gravada
	2- Retornar ao menu principal

### 2.7.3 Requisitos relacionados aos atendimentos realizados pelos agentes de campo nas localidades indicadas nas solicitações.

**Descrição:** Os agentes de campo devem ter acesso a todas as ocorrências abertas para poderem realizar a inserção de relatório. Após a inserção de relatório o agente terá a opção de fazer registros fotográficos relacionados ao relatório selecionado.

**Tabela 6-Caso de Uso 06- Inserir Relatório**

Nome do caso de uso	Inserir Relatório
Atores	Agente
Resumo	Como agente inserir relatório em ocorrência aberta
Pré-condições	Ter acesso ao sistema como agente de campo.
<b>Fluxo Principal</b>	
<b>Ações do ator</b>	<b>Ações do sistema</b>
1- Acessar pagina de menu de ocorrências	
2- Acessar pagina de ocorrências abertas	
3- Selecionar uma ocorrência.	
	4- Abrir pagina de inserção de relatório
5- Preencher os campos com os dados correspondentes: a. Vistoria: Texto da avaliação do local pelo agente de campo. b. Observação: Texto com uma informação importante relacionada ao relatório descrito no campo de vistoria.	
6- Enviar o relatório	
	7- Enviar os dados para o sistema associando a ocorrência com o funcionário
	8- Informar inserção de relatório
	9- Retornar ao menu de ocorrências
Restrições/Validações	1- Relatório não pode ser gravado
<b>Fluxo de Exceção- Retornar ao menu de ocorrências</b>	
<b>Ações do ator</b>	<b>Ações do sistema</b>
	1- Informar que relatório não pode ser gravado
	2- Retornar ao menu de ocorrências

**Tabela 7-Caso de Uso 07 - Registros Fotográficos do Relatório**

Nome do caso de uso	Inserir Registros Fotográficos do Relatório
Atores	Agente
Resumo	Como agente inserir registros fotográficos para o relatório selecionado
Pré-condições	Ter acesso ao sistema como agente de campo.
<b>Fluxo Principal</b>	
<b>Ações do ator</b>	<b>Ações do sistema</b>
1- Acessar pagina de inserção de fotos	
2- Selecionar um relatório	
	3- Abrir pagina para acesso a câmera
4- Realizar registro fotográfico	
5- Enviar foto.	
	6- Enviar foto para o servidor
	7- Registrar o caminho da foto
	8- Retornar para pagina de acesso a câmera
Restrições/Validações	1- Informar que imagem não foi enviada ao servidor
<b>Fluxo de Exceção- Imagem não enviada ao servidor</b>	
<b>Ações do ator</b>	<b>Ações do sistema</b>
	1- Informar que imagem não foi enviada ao servidor
	2- Retornar ao menu de relatório

**Tabela 8-Caso de Uso 08 - Criação de Relatório em formato PDF**

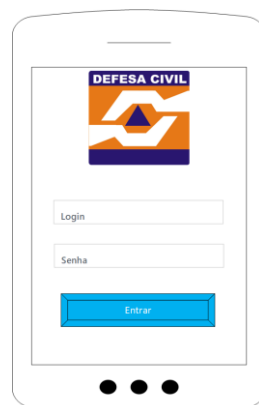
Nome do caso de uso	Criação de Relatório em formato PDF
Atores	Agente/Cadastrador
Resumo	Criação de relatório para impressão em formato PDF
Pré-condições	Ter acesso ao sistema como agente de campo ou cadastrador.
<b>Fluxo Principal</b>	
Ações do ator	Ações do sistema
1- Acessar pagina de Relatórios-Gerar PDF	
2- Selecionar um relatório	
	3- Abrir pagina com resumo do relatório selecionado
4- Conferir dados e dar comando para criação de documento em pdf	
	5- Gerar documento em pdf
6- Fazer download do documento	
7- Imprimir o documento	
8- Disponibilizar o documento ao solicitante da ocorrência relacionada ao relatório ou outra parte interessada	
Restrições/Validações	1- Informar que não foi possível criar o documento
<b>Fluxo de Exceção- Impossibilidade de criação de documento em PDF</b>	
Ações do ator	Ações do sistema
	9- Informar que não foi possível criar relatório
	10- Retornar ao menu de relatório

## 2.10 Protótipos de tela.

Os protótipos de tela foram realizados visando melhor compreensão das funcionalidades como também no auxilio para construção das telas da aplicação.

### 2.7.4 Protótipo de tela de acesso ao sistema.

A Figura 4 mostra o protótipo de tela de acesso ao sistema. A visão desta tela é igual para todos os perfis.

**Figura 4-Protótipo de tela- Acesso ao Sistema**

Fonte: Autor, 2018.

Conforme Caso de Uso 01 o usuário deverá informar o *login* e senha previamente cadastrados por administrador do sistema.

### 2.7.5 Protótipo de tela de cadastro de nova ocorrência.

A Figura 5 apresenta a visão de cadastro de nova ocorrência. Esta tela poderá ser acessada por todos os perfis de usuários.

**Figura 5-Protótipo de tela- Cadastro de Nova Ocorrência.**

Fonte: Autor, 2018.

Conforme Caso de uso 05 o usuário preencherá os campos definidos da seguinte forma:

- Rua Local: Nome da rua do local da ocorrência.
- Origem da Ocorrência: A forma que a ocorrência se originou.
- Tipo da Ocorrência: Classifica tipo de ocorrência.
- Numero do Local.
- Historio Inicial: Motivo da solicitação.
- Tipo de Solicitante: Classificação do tipo de solicitante.
- Departamento: Departamento que fez a solicitação.
- Rua do Solicitante: Nome da rua do solicitante.
- Email do Solicitante.

- Numero da Residência: Numero da residência do solicitante.
- Celular: Numero de celular do solicitante.
- Telefone: Numero do telefone fixo do solicitante.

### 2.7.6 Protótipo de tela de criação de relatório.

Podemos ver na Figura 6 a visão de tela de criação de relatório. Esta visão estará disponível para os perfis de usuário administrador e agente.

**Figura 6-Protótipo de tela- Tela de criação de relatório.**



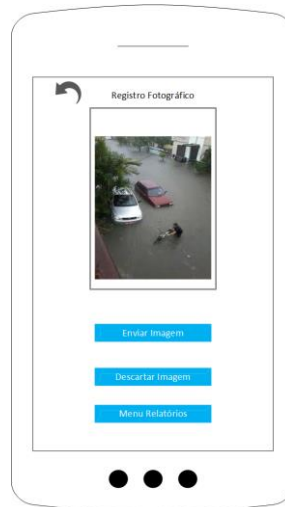
Fonte: Autor, 2018.

Consoante o Caso de Uso 07, esta tela ficou definida com apenas dois campos de inserção de informações para que, desta forma, abranja os vários tipos de relatórios que possam ser gerados mediante as variedades de ocorrências.

### 2.7.7 Protótipo de tela de Registro Fotográfico para o Relatório.

A Figura 07 demonstra a visão que o usuário tem da tela de registro de fotografia para o relatório. Esta visão é somente para os usuários com perfil de administrador ou agente.

**Figura 7- Protótipo de tela- Tela de Registros Fotográficos.**



Fonte: Autor, 2018.

Conforme Caso de Uso 08, usuário deverá escolher o relatório que deseja inserir fotos. Ao fazer registro da foto no aparelho mobile o agente terá a opção de enviar a foto ao repositório remoto ou descartá-la.

### 2.7.8 Protótipo de tela de Criação de Relatório em formato PDF.

A Figura 08 apresenta a visão do usuário na tela de geração de documento em PDF.

**Figura 8-Protótipo de tela- Tela de Criação de Relatório em formato PDF.**



Fonte: Autor, 2018.

Conforme Caso de Uso 08, o usuário realizará uma conferência nos dados do relatório. Para gerar o relatório o agente deverá solicitar que seja criado o relatório e depois dará o comando para baixar o relatório no aparelho móvel.

### 3- DESENVOLVIMENTO

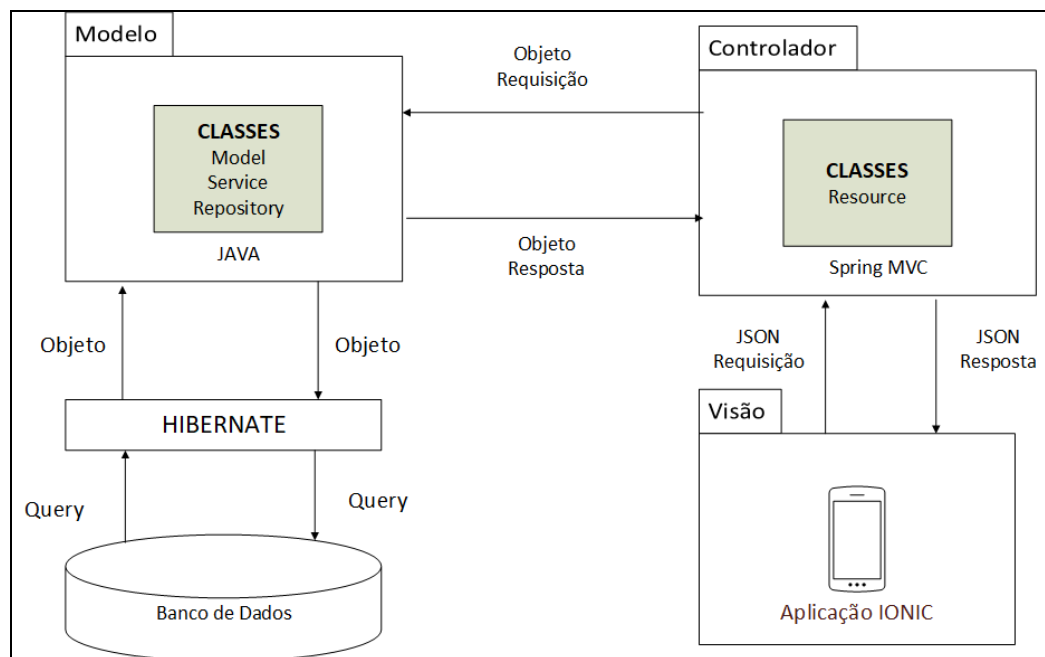
Neste capítulo será apresentado o desenvolvimento do sistema e a arquitetura usada para o desenvolvimento da aplicação atendendo os requisitos descritos no Capítulo 2(dois).

#### 3.1 Arquitetura do sistema.

Para a execução do projeto foi escolhido o padrão de arquitetura MVC. Este padrão divide o sistema em três camadas (Modelo, Visão e Controle).

A Figura 09 apresenta o modelo MVC para o servidor da aplicação.

**Figura 9- Diagrama do modelo MVC- Backend.**



Fonte: Autor, 2018.

Na camada Modelo temos os componentes Model, Service e Repository, que são os componentes responsáveis pela persistência dos dados e regras de negócio<sup>2</sup>.

<sup>2</sup> SOUSA, Eduardo. **Arquitetura de Aplicações Spring MVC: Uma Análise Baseada no Acoplamento Lógico**. Disponível em: [http://vem2017.ufu.br/artigos/Sousa\\_et\\_al\\_2017.pdf](http://vem2017.ufu.br/artigos/Sousa_et_al_2017.pdf). Acesso 05/09/2018

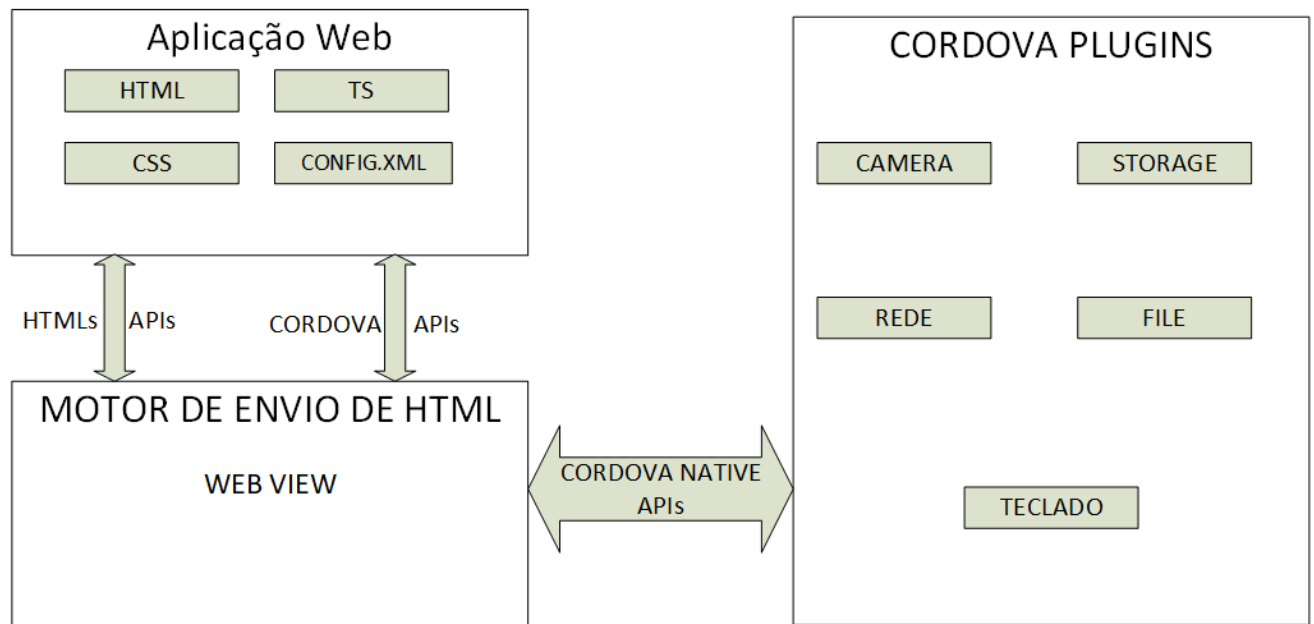


A camada Controlador é responsável pelo fluxo de dados entre as camadas Modelo e Visão definindo a maneira como a interface gráfica (aplicação mobile) deve agir a partir da interação do usuário e também atualiza as informações e objetos da camada Modelo.

A interação com a camada *backend* é feita por aplicação em dispositivo móvel, que realiza as requisições por meio de rotas REST, que é um estilo arquitetural de representação de dados baseado em recursos onde as trocas de informações são feitas em formatos de tipos de representações de conteúdo, que para esta aplicação será usado o formato JSON.

A Figura 10 apresenta arquitetura da aplicação do dispositivo móvel.

**Figura 10-Diagrama do modelo MVC- Frontend.**



A aplicação para o dispositivo móvel foi desenvolvida usando o *framework Ionic*. Este faz uso de tecnologias comumente utilizadas em soluções web de *frontend* como HTML, CSS e TypeScript.

O Ionic empacota o Cordova e o TypeScript para poder utilizar as funções nativas do aparelho móvel e a partir de um único código disponibilizar uma aplicação que pode ser usada em varias plataformas<sup>3</sup>.

<sup>3</sup> RIBEIRO, João. **Guimifiu: Uma aplicação para controle social sobre postos de combustíveis**. Trabalho de Conclusão de Curso – Universidade de Brasília – UnB Faculdade UnB Gama – FGA , 2017.

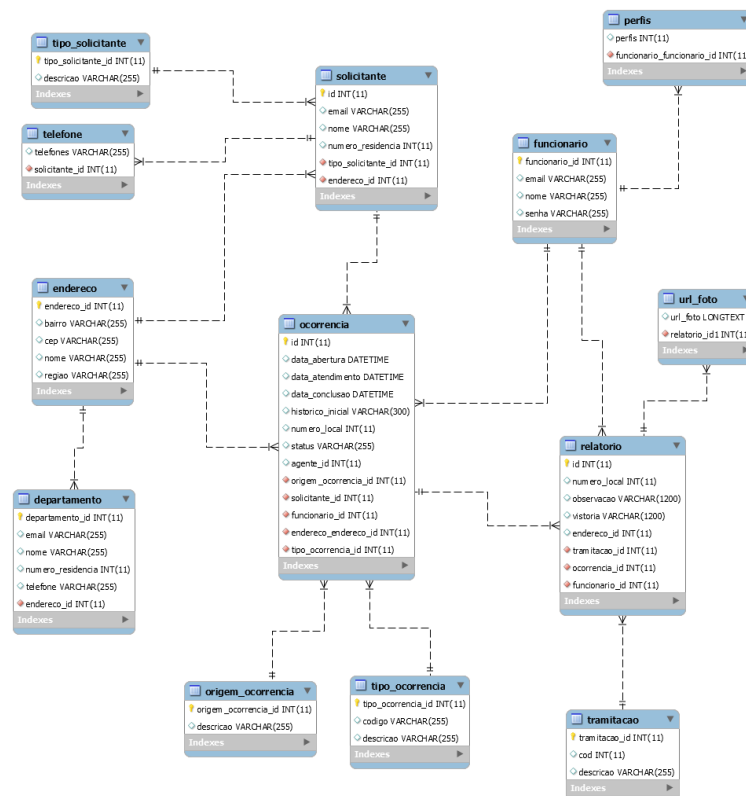
O framework Cordova é usado para desenvolvimento de aplicações nas plataformas Android, Blackberry, iOS, Windows Phone, Ubuntu, Windows (8.1/10) e OS X. Através de plug-ins, que são invocados pela linguagem *TypeScript*, o *Cordova* disponibiliza acesso aos recursos nativos do sistema operacional<sup>4</sup>.

O *TypeScript* é uma linguagem de script, criada pela Microsoft, que tem como característica ser puramente orientada a objetos, com herança baseada em classes e ser uma linguagem de script fortemente tipada, exigindo a definição de dados para o seu uso. As funcionalidades, quando compiladas através de processo chamado transpilação, converte o código para *JavaScript*<sup>5</sup>. O *TypeScript* proporciona sintaxe simplificada para criação do código.

### 3.2 Banco de Dados

A Figura 11 mostra o Modelo de Entidade de Relacionamento projetado para criação do banco de dados.

Figura 11-Modelo Lógico



Fonte: Autor, 2018.

<sup>4</sup> MENEGASSI, André Augusto. **Testes automatizados para aplicações móveis multiplataforma.** Dissertação (Mestrado) – Universidade Tecnológica Federal do Paraná. Programa de Pós-Graduação em Informática. Cornélio Procópio, 2018

<sup>5</sup> Vitor da Silva Cruz. **Tecnologias Web para o Desenvolvimento Mobile Nativo.** IV Edição do SIMTEC - Fatec Taquaritinga

Para contemplar o Caso de Uso 5 a tabela Ocorrência foi definida conforme mostra a Figura 11 e esta relacionada diretamente com outras 6(seis) tabelas.

Esta tabela está relacionada com a tabela Funcionário no tipo de relacionamento Um para Muitos, pois um funcionário pode cadastrar varias ocorrências e uma ocorrência especifica pode ser cadastrada por um funcionário somente. O objetivo desta tabela é armazenar as informações dos funcionários da DC e usar as informações para criar restrições de acesso na aplicação com os dados fornecidos pela Tabela Perfil que esta relacionada com a Tabela Funcionário.

A relação com a Tabela Tipo\_Ocorrência é necessária para que exista uma classificação da ocorrência. A Tabela Origem\_Ocorrência foi criada para indicar quem originou o pedido de ocorrência. A Tabela Endereço informa o logradouro da ocorrência indicando inclusive a região da cidade onde encontra-se o local a ser vistoriado, auxiliando o agente na locomoção. A Tabela Url\_Foto armazena o caminho da fotografia no servidor onde está armazenada e através do id do relatório relaciona a foto com o respectivo atendimento.

Conforme Caso de Uso 8, a Tabela Solicitante é necessária para vincular o solicitante da ocorrência com os relatórios gerados e constar no relatório pdf quem realizou o pedido.

Neste sistema, os Departamentos da Prefeitura podem ser incluídos como solicitantes de ocorrências e por este motivo, foi criada a Tabela Departamento. O intuito é armazenar os dados dos Departamentos para poder inseri-los automaticamente, através da aplicação, como dados do solicitante da ocorrência.

O SGBD escolhido para este projeto foi o MySQL<sup>6</sup> versão 8.0.12. É um banco de dados de código aberto muito usado em aplicações baseadas na Web com grande capacidade de armazenamento e usado por grandes empresas.

### **3.3 Estrutura de classes do backend do projeto.**

Para a implementação do *backend* foi utilizado o framework Spring<sup>7</sup>. O Spring é um framework Java de código aberto sendo dividido em módulos. Os aplicativos podem escolher quais módulos eles precisam. No coração estão os módulos do contêiner central, incluindo um modelo de configuração e um mecanismo de injeção de dependência.

---

<sup>6</sup> Disponível em <https://dev.mysql.com/doc/refman/8.0/en/what-is-mysql.html>. Acessado em 12/09/2018.

<sup>7</sup> Disponível em <https://docs.spring.io/spring/docs/current/spring-framework-reference/overview.html#overview>. Acessado em 12/09/2018

Como ferramenta de gerenciamento de automação e construção de projetos foi usado o Maven<sup>8</sup> que atua na gestão de dependências que são configuradas em um arquivo chamado pom.xml que contem a estrutura, dependências e características do projeto.

Para este trabalho foi usado o Spring Boot<sup>9</sup> que é um projeto que tem como finalidade acelerar o processo de configuração da aplicação estabelecendo configurações default.

A Figura 12 mostra parte do arquivo pom.xml deste projeto.

**Figura 12-Fragmento pom.xml.corrigir**

```
<?xml version="1.0" encoding="UTF-8"?>
<project xmlns="http://maven.apache.org/POM/4.0.0"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xsi:schemaLocation="http://maven.apache.org/POM/4.0.0 http://maven.apache.org/xsd/maven-
4.0.0.xsd">
  <modelVersion>4.0.0</modelVersion>

  <groupId>com.defesa</groupId>
  <artifactId>DefesaCivl2</artifactId>
  <version>0.0.1-SNAPSHOT</version>
  <packaging>jar</packaging>

  <name>DefesaCivl2</name>
  <description>Criacao Ferramenta de Despacho de Ocorrencias</description>
  <dependencies>
    <!-- https://mvnrepository.com/artifact/org.junit/junit5-engine -->
    <dependency>
      <groupId>org.junit</groupId>
      <artifactId>junit5-engine</artifactId>
      <version>5.0.0-ALPHA</version>
    </dependency>
```

Fonte Autor, 2018

Para que haja correspondência dos dados persistidos no banco e os objetos manipulados no *backend* as classes do projeto precisam estar em sincronia com as tabelas do banco de dados modelado para o sistema. Os campos das tabelas e os nomes dos atributos das classes precisam ser idênticos para que quando for realizada alguma consulta no banco de dados, o modelo receba os dados correspondentes aos atributos, preenchendo-os e retornando ao

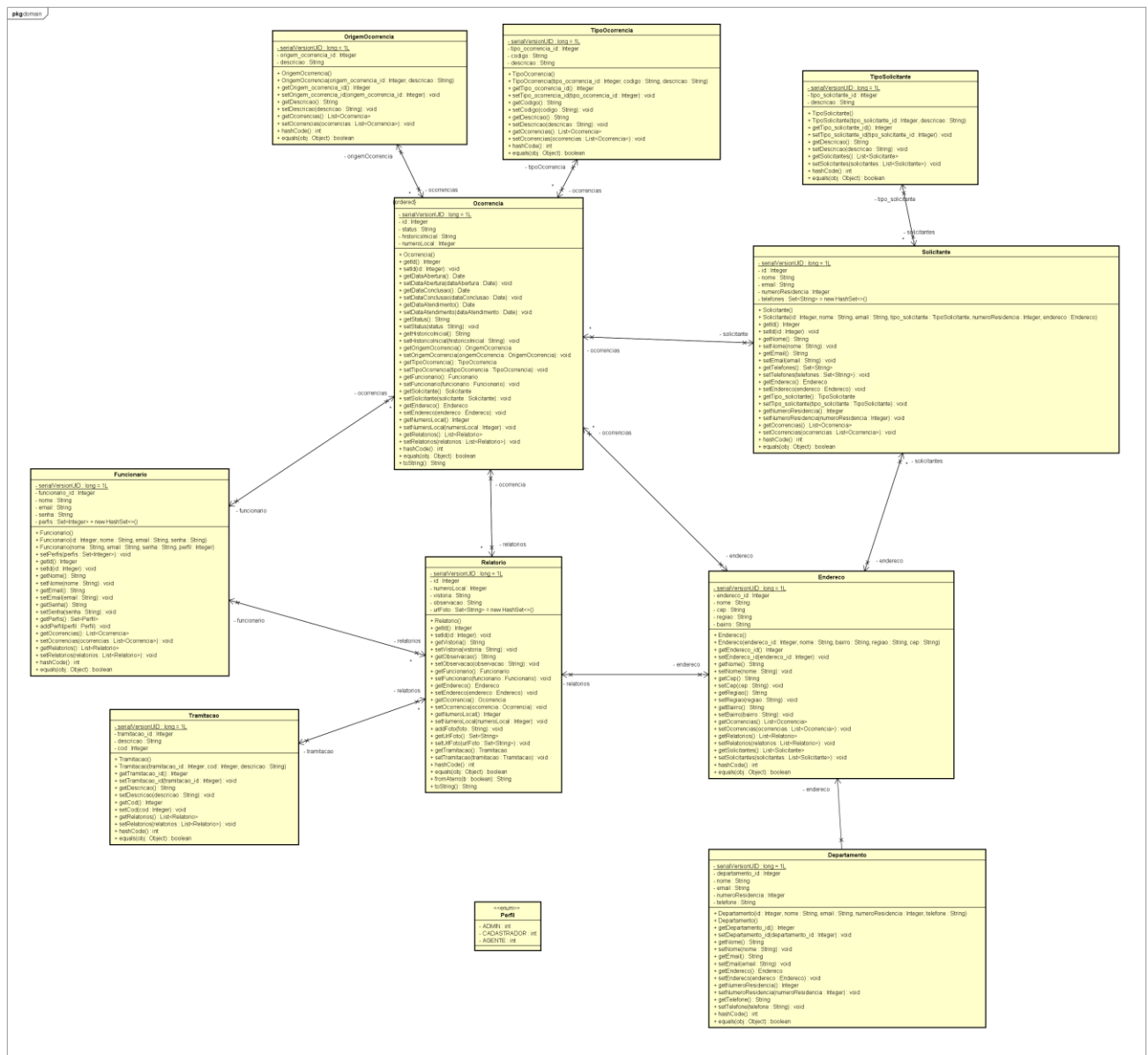
<sup>8</sup> Disponível em <https://maven.apache.org/what-is-maven.html>. Acessado em 12/09/2018

<sup>9</sup> Disponível em <https://spring.io/projects/spring-boot>. Acessado 12/09/2018

método que solicitou o objeto. Esta consulta dá-se por meio de ferramenta ORM (Object Relational Mapper).

A figura 13 apresenta a estrutura de classes de modelo do objeto.

**Figura 13-Diagrama de Classes**



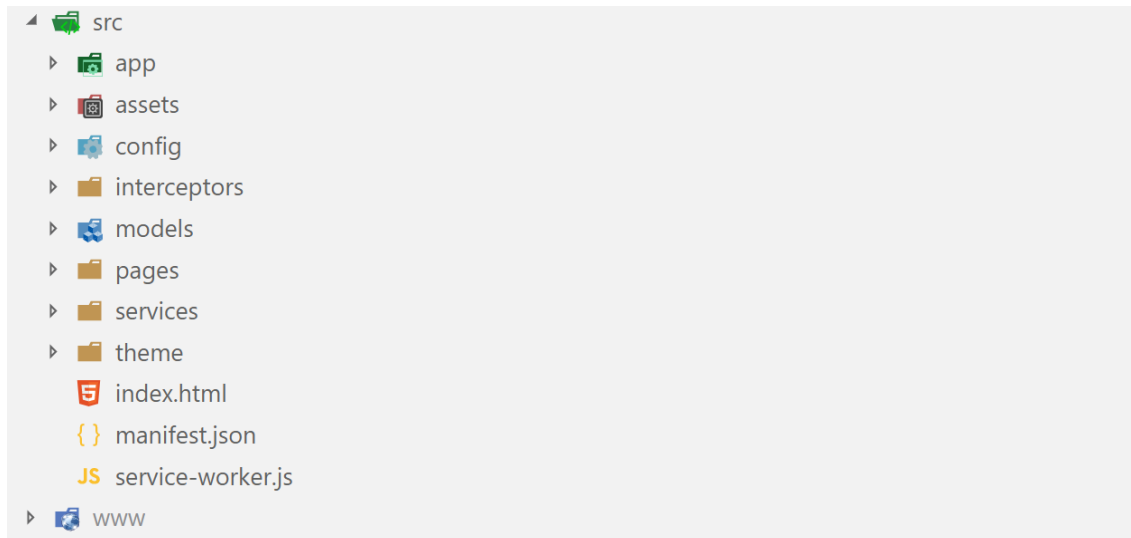
Fonte Autor, 2018

### 3.4 Estrutura do frontend do projeto.

O *frontend* do projeto, conforme descrito no capítulo 2(dois), consiste em uma aplicação que consome os serviços disponibilizados pelo *backend* do projeto.

A Figura 14 apresenta estrutura do código da aplicação *frontend*.

**Figura 14-Estrutura arquivos aplicação mobile**



Fonte Autor, 2018

Para que haja correspondência com os dados da aplicação *mobile* e os objetos das classes da aplicação *backend* foram criados arquivos de interfaces DTO.

A Figura 15 mostra a correspondência dos dados entre a interface EnderecoDTO e a classe EnderecoDTO .

**Figura 15- EnderecoDTO**

<pre> 1 2  export interface EnderecoDTO{ 3 4      id:string; 5      nome:string; 6      cep:string; 7      bairro:string; 8      regiao:string; 9  }</pre>	<pre> 1  package com.defesa.domain.dto; 2 3 4  import java.io.Serializable; 5 6  import com.defesa.domain.Endereco; 7  public class EnderecoDTO implements Serializable { 8 9      private static final long serialVersionUID = 1L; 10 11     private Integer id; 12     private String nome; 13     private String cep; 14     private String regiao; 15     private String bairro; 16 17</pre>
--	--

Fonte Autor, 2018

DTO(*Data Transfer Object*)<sup>10</sup> é um objeto simples usado para transferência de dados. São classes com atributos somente e com o intuito de criar correspondência entre as classes da

<sup>10</sup> Disponível em <https://martinfowler.com/eaCatalog/dataTransferObject.html>. Acessado em 12/09/2018

aplicação *backend* com a interface do usuário. Desta forma entre as duas camadas do sistema somente os dados necessários são trafegados.

O envio dos dados relacionados aos relatórios Conforme Caso de Uso 6 ao entrar na pagina de inserção de relatório, o agente informará os dados pertinentes ao atendimento e enviará os dados para o servidor. A página fica responsável por agrupar os dados e associa-los a um objeto, conforme mostra a Figura 16.

**Figura 16- Inseere Relatório frontend**



Fonte Autor, 2018

Através da função `this.formGroup` a pagina agrega os dados recebidos da camada de interação(arquivos.html) e os encaminha a função `inseereRelatorio()`. Esta função entrega os dados para o serviço responsável de envia-los ao servidor. O serviço encaminha um objeto do tipo `RelatorioNewDTO` que faz correspondência com classe com mesmo nome e atributos no *backend*. A Figura 17 mostra o serviço responsável em encaminhar os dados de um novo relatório.

**Figura 17– Relatório Service**



Fonte Autor, 2018

### 3.5 Persistência

Para que a Defesa Civil possa utilizar um banco de dados de sua escolha é necessário à criação de uma camada de persistência que abstraia o banco de dados permitindo que as informações geradas pelo atendimento das ocorrências possam ser persistidas em qualquer base via SQL.

Persistência de dados é um procedimento de armazenamento e manutenção das informações de um sistema com o objetivo de garantir que as informações utilizadas sejam duráveis, possibilitando sua recuperação posterior<sup>11</sup>.

A persistência de dados foi realizada usando os frameworks *Hibernate*<sup>12</sup>. Este framework implementa a especificação *JPA* do Java que padroniza a persistência de dados armazenados em bancos relacionais

O *Hibernate* realiza consulta e persistência Objeto-Relacional em Java, transformando os registros contidos no banco de dados relacional em objetos. As informações destes objetos em Java são persistidas em forma de linha e coluna no banco. Para o desenvolvedor, as particularidades do SGBD são todas abstraídas.

### 3.6 Model

Esta camada é responsável pela definição das classes de domínio que representam as propriedades que serão persistidas bem como a relação de uma classe com outra.

Conforme Caso de Uso 5 uma nova ocorrência deve informar um histórico inicial, que podemos entender como o motivo da solicitação de atendimento. Desta forma, foi criado na tabela Ocorrência um campo do tipo texto para armazenar esta informação e na classe Java correspondente um atributo do tipo String com o mesmo nome.

A anotação *@Entity* é utilizada para informar que uma classe é uma entidade e a partir disso o *Hibernate* estabelece a ligação desta classe com uma tabela do banco de dados.

Uma entidade representa uma tabela no banco de dados e uma instância dessa entidade representa uma linha desta tabela no banco.

---

<sup>11</sup> Disponível em <http://www.revistatis.dc.ufscar.br/index.php/revista/article/view/366/129>. Acessado em 21/09/2018

<sup>12</sup> Disponível em <http://hibernate.org/orm/>. Acessado em 12/09/2018



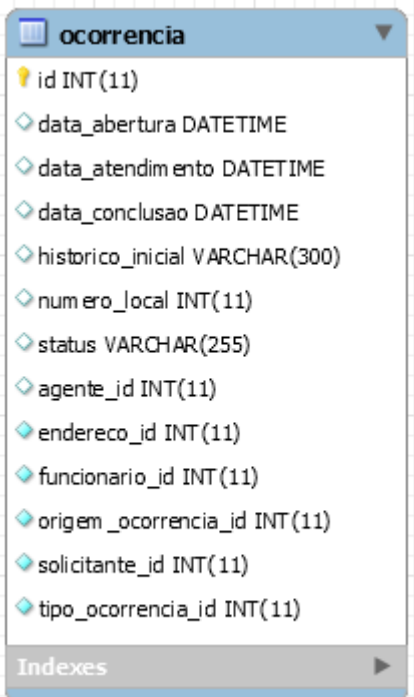
A anotação `@Id` indica na classe qual atributo vai ser mapeado com o campo da tabela que é chave primaria. A anotação `@GeneratedValue(strategy = GenerationType.IDENTITY)` determina que o id será gerado automaticamente.

A anotação `@Column` é utilizada para especificar os detalhes da coluna que um campo ou propriedade será mapeado. A anotação é opcional e na classe representada foi usada para definir o tamanho do campo na tabela, no caso um campo de 300 (trezentos) caracteres.

A relação da classe Ocorrência com a classe Relatório é realizada pela anotação `@OneToMany` que indica uma relação de Um-Para-Muitos, ou seja uma Ocorrência pode ter vários relatórios e um Relatório está associado a uma única Ocorrência.

Para que ocorra o mapeamento-objeto-relacional entre as classes Java e as tabelas do banco de dados relacional, os atributos da classe devem corresponder aos campos da tabela como podemos ver na Figura 18.

**Figura 18– Relação objeto/relacional**

 <p>The screenshot shows a table named 'ocorrencia' with the following columns and data types:</p> <ul style="list-style-type: none"> <li>id INT(11)</li> <li>data_abertura DATETIME</li> <li>data_atendimento DATETIME</li> <li>data_conclusao DATETIME</li> <li>historico_inicial VARCHAR(300)</li> <li>numero_local INT(11)</li> <li>status VARCHAR(255)</li> <li>agente_id INT(11)</li> <li>endereco_id INT(11)</li> <li>funcionario_id INT(11)</li> <li>origem_ocorrencia_id INT(11)</li> <li>solicitante_id INT(11)</li> <li>tipo_ocorrencia_id INT(11)</li> </ul> <p>Below the columns, there is a section for 'Indexes'.</p>	<pre> @Entity public class Ocorrencia implements Serializable {      private static final long serialVersionUID = 1L;      @Id @GeneratedValue(strategy = GenerationType.IDENTITY) private Integer id;  private Date dataAbertura; private Date dataConclusao; private Date dataAtendimento; private String status; @Column(length = 300) private String historicoInicial;  private Integer numeroLocal; @ManyToOne @JoinColumn(name = "endereco_id") private Endereco endereco; </pre>
---	---

### 3.7 Repository

Através da anotação *@Repository* indicamos que uma classe é responsável pelo acesso aos dados.

A Figura 19 mostra a interface *DepartamentoRepository* que é responsável pelo acesso aos dados relacionados com a classe de domínio *Departamento*. A implementação dos métodos não é necessária, pois é realizada em tempo de execução pelo framework *Spring Data*<sup>13</sup>. A *JpaRepository* fornece funcionalidades CRUD sofisticadas para a classe que está sendo gerenciada

**Figura 19- DepartamentoRepository**

```
package com.defesa.repositories;

import org.springframework.data.jpa.repository.JpaRepository;
import org.springframework.stereotype.Repository;

import com.defesa.domain.Departamento;

@Repository
public interface DepartamentoRepository extends JpaRepository<Departamento, Integer>
{
}
}
```

Fonte Autor, 2018

### 3.8 Services

Esta camada atua como mediador entre uma camada de repositório e uma camada de controle. Nesta camada ocorrem as validações dos dados e aplicação de regras de negócios.

Na Figura 20 vemos uma função que ao receber o numero de id realiza uma validação nos dados retornando somente os relatórios do funcionário informado e que não foram classificados com tramitação para arquivo.

---

<sup>13</sup> Disponível em <https://docs.spring.io/spring-data/data-commons/docs/1.6.1.RELEASE/reference/html/repositories.html>. Acessado 12/09/2018

**Figura 20- Busca de relatórios por funcionário**

```

public List<RelatorioDTO> findAllFuncionario(Integer id) { // Busca Relatorios por
funcionario
    Funcionario func = funcService.find(id);
    List<Relatorio> rel = repo.findByFuncionario(func);
    List<RelatorioDTO> listDto = new ArrayList<RelatorioDTO>();
    for (Relatorio r : rel) {
        if (!(r.getTramitacao().getDescricao().equals("ARQUIVO"))) {
            listDto.add(new RelatorioDTO(r));
        }
    }
    return listDto;
}

```

Fonte Autor, 2018

Na Figura 21 vemos uma função que aplica regra de negocio que define que para cada nova ocorrência a mesma deve ser persistida no banco de dados com o status “Aberta”. A camada de persistência recebe o objeto pronto para ser gravado.

**Figura 21- OcorrenciaService – Insert**

```

@Transactional
public Ocorrencia insert(Ocorrencia obj) {
    obj.setId(null);
    obj.setStatus(status);
    obj.setDataAbertura(new Date(System.currentTimeMillis())); data de abertura
    solService.insert(obj.getSolicitante());

    return repo.save(obj);
}

```

Fonte, Autor 2018

### 3.9 Registros Fotográficos

Conforme Caso de uso 7 ao realizar um registro fotográfico a imagem é enviada para um servidor e o caminho da fotografia é armazenado no banco de dados conforme Figura 22.

**Figura 22- Tabela Url\_Foto.**


Url_Foto	
url_foto	VARCHAR(60)
relatorio_relatorio_id	INT(11)
Indexes	

Fonte, Autor 2018

Como repositório das imagens para este projeto foi escolhido o serviço S3 da Amazon AWS<sup>14</sup> que é um serviço de armazenamento de objetos criado para armazenar e recuperar qualquer quantidade de dados de qualquer local: sites e aplicativos móveis, aplicativos corporativos e dados de sensores ou dispositivos da IoT.

A aplicação do dispositivo móvel captura a imagem e realiza as conversões necessárias para enviar o arquivo para o serviço S3 da Amazon. A Figura 23 mostra a função que envia a imagem juntamente com a indicação do relatório do qual está vinculado.

**Figura 23-UploadPicture –Função do frontend.**

```
uploadPicture(picture, id_relatorio) {
  let pictureBlob = this.imageUtilService.dataUriToBlob(picture);
  let formData: FormData = new FormData();
  formData.set('file', pictureBlob, 'file.png');
  return this.http.post(
    `${API_CONFIG.herokuBaseUrl}/relatorios/picture/${id_relatorio}`,
    formData,
    {
      observe: 'response',
      responseType: 'text'
    }
  );
}
```

Fonte, Autor 2018

O servidor recebe os dados, realiza a associação do relatório com a imagem através do número de id e armazena o caminho da imagem ao enviá-la para o repositório de imagens. A

<sup>14</sup> Disponível em <https://aws.amazon.com/pt/s3/?nc=sn&loc=1>. Acessado em 02/10/2018

Figura 24 mostra a função que do lado do servidor classifica e envia a imagem para o serviço de armazenamento.

**Figura 24-UploadPicture – Função do backend.**

```
public URI uploadProfilePicture(MultipartFile multipartFile, Integer id_relatorio) {
    Relatorio rel = find(id_relatorio);
    int a = rel.getUrlFoto().size();// recuperando quantidade de url
    a++;
    BufferedImage jpgImage = imageService.getJpgImageFromFile(multipartFile);
    jpgImage = imageService.cropSquare(jpgImage);
    jpgImage = imageService.resize(jpgImage, size);
    String fileName = prefix + rel.getId() + "-" + a + ".jpg"; // atribuindo numero a foto do relatório
    URI uri = s3Service.uploadFile(imageService.getInputStream(jpgImage, "jpg"),fileName, "image");
    String b= "https://s3-sa-east-1.amazonaws.com/projeto-defesacivil- sjc/";
    rel.addFoto(b+fileName);
    repo.save(rel);
    return uri;
}
```

Fonte, Autor 2018

### 3.10 Serviços

Esta camada é responsável em estabelecer a comunicação da camada de persistência com a camada de visão que, neste caso, é representada pela aplicação de dispositivo móvel.

Dentro do padrão MVC esta camada representa o controlador da aplicação que disponibiliza recursos através de rotas REST para serem consumidos pela camada de visão ou outras aplicações. Recurso é uma abstração de determinado tipo de informação gerenciada pela aplicação. Este recurso deve ter uma identificação única que será informada na requisição usando o conceito de URI. As URIs, que são cadeias de caracteres que identificam recursos na internet, devem ser definidas relacionadas com o domínio da aplicação.

Para contemplar requisito do Caso de uso 6(seis), o sistema deve fornecer a aplicação do dispositivo móvel as ocorrências que encontra-se com o status “Aberta”. A Figura 25 mostra função que disponibiliza este recurso. O método faz uma chamada à camada de persistência solicitando os dados necessários para responder a requisição.

**Figura 25- Controle para Lista Ocorrências Abertas**

```

@RequestMapping(value="/abertas",method = RequestMethod.GET)
public ResponseEntity<List<OcorrenciaDTO>> listarAbertas() {

    List<Ocorrencia> list= service.findByStatus();

    List<OcorrenciaDTO> listDto = list.stream().map(obj ->
new OcorrenciaDTO(obj)).collect(Collectors.toList());

    return ResponseEntity.ok().body(listDto);

}

```

Fonte, Autor 2018

A função então retorna ao solicitante os dados no formato JSON<sup>15</sup>, que é um formato de troca de dados independente da linguagem de programação usada na aplicação. A Figura 26 mostra parte do retorno da requisição acima descrita. Os objetos estão no padrão DTO.

**Figura 26- JSON de Ocorrências Abertas**

```

{
  "id": 1,
  "dataAbertura": "20-10-2017",
  "dataAtendimento": "20-10-2017",
  "dataConclusao": "",
  "status": "Aberta",
  "historicoInicial": "Deslizamento pos chuvas",
  "rua": "Rua Benedito Pedroso de Andrade",
  "bairro": "Águas da Prata",
  "regiao": "Leste",
  "cep": "12225261",
  "nomeFuncionario": "Andre Simão",
  "idFuncionario": 4,
  "numeroLocal": 101,
  "tipo": "ALAGAMENTO DE VIA",
  "origem": "PESSOALMENTE"
},

```

Fonte, Autor 2018

<sup>15</sup> Disponível em <http://www.json.org/>. Acessado em 04/10/2018

### 3.11 Controle de Acesso

Acesso é a possibilidade de realizar alguma operação em algum recurso computacional. Controle de acesso<sup>16</sup> é o meio pelo qual a capacidade de realizar a operação é explicitamente habilitada ou restringida de alguma forma. Divide-se em duas partes: autenticação e autorização. Autenticação refere-se ao processo de fornecer ao sistema informações que identifiquem com um grau de certeza suficiente o usuário que está requisitando o acesso aos recursos computacionais. Em termos gerais, é o processo de provar ao sistema que o usuário é realmente quem diz ser, e não alguém se passando por ele. O processo de autorização rege exatamente que operações, sob que recursos computacionais, o usuário poderá executar no sistema. Para que sejam efetivadas quaisquer avaliações de autorização é necessário ter passado pela etapa de autenticação.

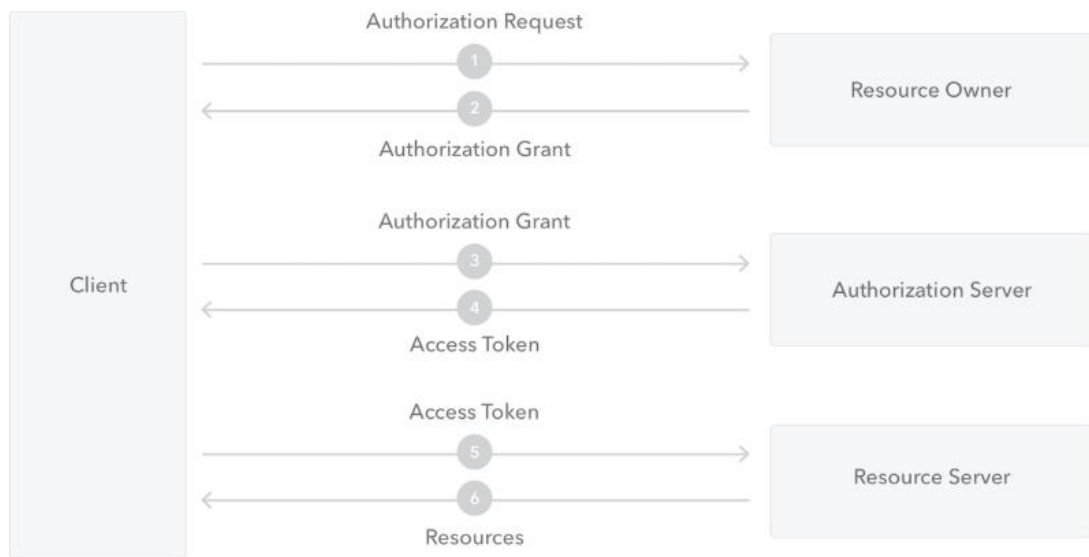
Para esta aplicação foi usado o padrão OAuth que é um protocolo de autorização para API's web voltado a permitir que aplicações cliente acessem um recurso protegido em nome de um usuário. Os usuários tem acesso, por uma aplicação externa, aos seus recursos privados sem ter que compartilhar suas senhas e usuários.

A Figura 27 mostra o fluxo de autorização/authenticação.

1. O aplicativo (cliente) solicita autorização do proprietário do recurso para acessar os recursos.
2. Desde que o Proprietário do Recurso autorize esse acesso, o Aplicativo recebe uma Concessão de Autorização. Esta é uma credencial que representa a autorização do Proprietário do Recurso.
3. O aplicativo solicita um *token* de acesso, autenticando-se com o servidor de autorização e concedendo a concessão de autorização.
4. Contanto que o Aplicativo seja autenticado com êxito e a Concessão de Autorização seja válida, o Servidor de Autorização emitirá um Token de Acesso e o enviará ao Aplicativo.
5. O aplicativo solicita acesso ao recurso protegido pelo servidor de recursos e autentica apresentando o *token* de acesso.
6. Desde que o *token* de acesso seja válido, o servidor de recursos atende a solicitação do aplicativo.

---

<sup>16</sup> Mattos, C. L. A. Sentinel: um engenho Java para controle de acesso RBAC. Trabalho de Graduação em Segurança da Informação. Recife, Agosto de 2003

**Figura 27- Fluxograma Controle de Acesso**

Fonte, <https://auth0.com/docs/protocols/oauth2>

### 3.12 Autenticação

Nesta aplicação foi usado o framework Spring Security<sup>17</sup> que é responsável por fornecer autenticação e autorização para aplicações em Java.

O usuário solicita acesso ao sistema através de suas credenciais e o *backend* realiza a verificação dos dados para confirmar ou não o acesso.

Ao realizar a autenticação com sucesso, o Spring retorna uma resposta à requisição contendo o *token*, conforme Figura 28. Foi usada a biblioteca JWT<sup>18</sup> que é responsável pela geração de tokens de acesso e tempo de sessão de uma forma segura e compacta de transmissão de informações entre duas partes no formato JSON.

**Figura 28– Resposta Login.**

Authorization → Bearer

eyJhbGciOiJIUzUxMiJ9.eyJzdWIiOiJYXBhYmxhbmNhMUBib2wuY29tLmJyIiwiaXNjaXoxNTQ0NjU1MDEzfQ.B61M4cFl1EVUf8U7Q9LSS3fYWitKa6TI\_igpVHG01GmXaHmPbRwqoaKU7H9eIbbAHGiE9kIA1dtJd5XdXyZOWQ

Fonte, Autor 2018

<sup>17</sup> Disponível em <https://spring.io/projects/spring-security>. Acessado em 08/10/2018

<sup>18</sup> Disponível em <https://jwt.io/introduction/>. Acessado em 08/10/2018.



Para contemplar Caso de Uso 01 que determina que o usuário deva informar senha e nome de usuário para acessar o sistema foi implementado gerenciamento de senhas com a criptografia BCrypt<sup>19</sup>. Após ser criptografada a senha é armazenada no banco de dados através de algoritmo *hash*<sup>20</sup> que mapeia uma *string* de comprimento arbitrário para comprimento fixo. Apenas o *hash* da senha é armazenado no banco de dados, não sendo possível resgatar a senha original mesmo com o acesso ao banco de dados conforme a Figura 29 a seguir.

**Figura 29- Senha Criptografada com BCrypt**

FUNCIONARIO_ID	EMAIL	NOME	SENHA	PERFIL_ID
1	capablanca1@bol.com.br	Antonio Alves	\$2a\$10\$F10Q9wxXrwU.7p4/Gh7Ooe9MLSKgHEmXyG8kRadIYDfRdzM8tWJ5O	1
2	programacaoacm@gmail.com	Joao carlos	\$2a\$10\$c./Y49agAK46tdzmAPYQpOe6T/GSIFLG6sNhthC1PQqoD/EIY5qW	2
3	pedroTiago@ex.com.br	Pedro Tiago	\$2a\$10\$D9KbcrRGqwwduHsgLEn5ZOQXF2GGdTzElsybFVTNhxExZfBFmq0C	3
4	andre@ex.com.br	Andre Simão	\$2a\$10\$/mYJok2OifJXREz9D/xage0q9zpkNN94A3kARxgO4HzzBpAGD2ehe	2
5	fb@ex.com.br	Felipe Bartolomeu	\$2a\$10\$C1NTiMYQtWzNY5K109fZke6tXhn0O2217TvL3PjRSWczYVW6.TZ6	2
6	marisae@ex.com.br	Marisa	\$2a\$10\$pwzLVkV7tD6zZSWwXLC8VuK7GbW65n/WUsNAYWikFE2JiCscU/y.y	2
7	sebastiana@ex.com.br	Sebastiana	\$2a\$10\$PZMOL/hH3kWiLkMvLsOyenuq7GycPL8Me6otvs.wwQQShvG21mPa	2
8	fernanda@ex.com.br	Fernanda	\$2a\$10\$mNF4TvbGf8lvdmqWJzz.euQ6keta//DY4GJALfDmVbBV9Tf7Mrcfy	1
9	Teste@feste	Teste	\$2a\$10\$2gn4ei9MKIL/XufS.HSTUum4wy0s.iFLZ7VmcusROBGjQNMw46Qt.	1

Fonte, Autor 2018

### 3.13 Autorização

Ao incluir as dependências no projeto o Spring Security realiza a proteção da aplicação e desta forma devemos informar quais recursos do sistema estarão liberados ou bloqueados por padrão. A Tabela 30 mostra as rotas de acesso aos recursos e quais perfis tem acesso aos serviços apontados pelos endpoints.

**Tabela 9-Endpoints**

Caso de Uso	Recurso	Permissão	Endpoint
Caso de Uso 01- Efetuar acesso	Acessar o sistema	Todos os perfis	/login
Caso de Uso 02 - Alterar senha	Alterar senha no sistema	Todos os perfis	/auth/forgot
Caso de Uso 03 - Recuperar Senha de Usuário	Recuperar o acesso ao sistema	Todos os perfis	/auth /novasenha
Caso de Uso 04 -	Inserir novo usuário no	Perfil Administrador	funcionarios/novo

<sup>19</sup> Disponível em <https://docs.spring.io/spring-security/site/docs/5.0.x/api/org.springframework.security.crypto.bcrypt/BCrypt.html>. Acessado em 08/10/2018

<sup>20</sup> Disponível em [https://www.nist.gov/publication/get\\_pdf.cfm?pub\\_id=911479](https://www.nist.gov/publication/get_pdf.cfm?pub_id=911479). Acessado em 23/10/2018.



## 4- RESULTADOS

Neste capítulo serão apresentados os resultados dos experimentos realizados bem como as soluções implementadas para correções dos erros e propostas de melhorias. Os testes foram definidos com base nos casos de usos para verificar se o sistema atende os requisitos definidos no capítulo dois.

### 4.1 - 1º Experimento- Acesso ao sistema

O objetivo deste experimento é realizar testes na aplicação com o intuito de verificar se a aplicação realiza controle de acesso conforme perfil de usuário.

A bateria de teste foi dividida em casos de teste (CT) seguido de um contador incremental que estão relacionados aos requisitos conforme mostra a tabela 10:

**Tabela 10- Relação Casos de Testes e Requisitos- Acesso ao Sistema**

<b>Caso de teste</b>	<b>Requisito Relacionado</b>
CT01- Login com perfil de cadastrador	R1- O usuário acessará o sistema através de credenciais que são email e uso de senha.
CT02- Login com perfil de agente	R1- O usuário acessará o sistema através de credenciais que são email e uso de senha.
CT03- Login com perfil de administrador	R1- O usuário acessará o sistema através de credenciais que são email e uso de senha.

A Tabela 10 mostra os resultados dos testes.

**Tabela 11- Testes de acesso ao sistema 1º Experimento**

<b>Ação do usuário</b>	<b>Resposta do sistema</b>	<b>Expectativa Atendida</b>
CT01- Logar com perfil de cadastrador e tentar inserir ocorrência.	Sistema verifica perfil de usuário e não permite inserção de nova ocorrência.	Sim
CT02- Logar com perfil de agente e tentar inserir novo funcionário no sistema	Sistema verifica perfil de usuário e não permite inserção de novo funcionário.	Sim
CT03- Logar com perfil de	Sistema verifica perfil de	Sim

administrador e tentar inserir novo funcionário no sistema	usuário e permite inserção de novo funcionário.	
--	---	--

Fonte, Autor 2018.

O 1º experimento mostrou-se satisfatório, pois ao inserir novo usuário ao sistema é necessário informar o perfil do funcionário como podemos ver através da Figura 31 que mostra Tela de Inserção de Novo Funcionário para que o sistema filtre as permissões de acesso.

**Figura 31-Tela Insere Novo Funcionário**

The screenshot shows a mobile application interface for adding a new employee. At the top, there's a status bar with various icons and the time 11:45. Below it, a header bar with a back arrow and the title 'Novo Funcionario'. The form consists of four input fields: 'Nome\*' (Name), 'Email\*', 'Senha\*' (Password), and 'Tipo de Funcionario' (Employee Type) which has a dropdown arrow. At the bottom of the form is a prominent blue button labeled 'CRIAR USUARIO'. The bottom of the screen shows the standard Android navigation bar with back, home, and recent apps buttons.

Fonte, Autor 2018.

Outro exemplo observado no experimento foi a permissão negada ao usuário com perfil cadastrador que tem acesso a lista de ocorrências abertas, mas não pode criar relatórios como podemos ver na Figura 32 que mostra resposta do sistema a restrição de acesso.

**Figura 32-Alerta de permissão negada**



Fonte, Autor 2018.

#### **4.2 - 2º Experimento- Solicitações de ocorrências.**

O objetivo deste teste é verificar se o sistema contempla os requisitos descritos no Caso de Uso 05 fornecendo ao usuário um formulário claro e automatizado para inserção das informações pertinentes as solicitações de ocorrências.

A bateria de testes foi dividida da seguinte forma:

- CT04 – Registro de nova ocorrência com dados corretos.
- CT05 – Registro de nova ocorrência com dados incorretos.

A Tabela 11 mostra os resultados dos testes.

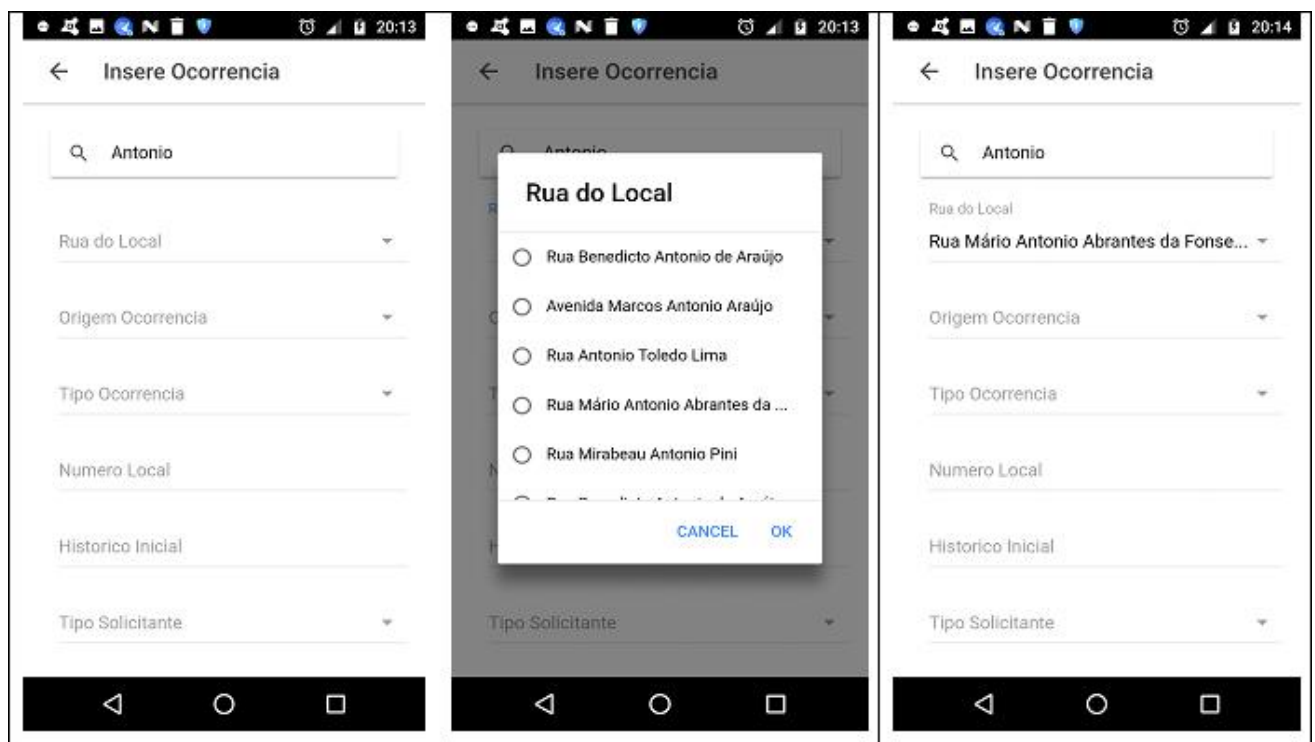
**Tabela 12- Testes de acesso ao sistema 2º experimento**

<b>Ação do usuário</b>	<b>Resposta do sistema</b>	<b>Expectativa Atendida</b>
------------------------	----------------------------	-----------------------------

CT04 – Registro de nova ocorrência com dados corretos.	Mandar os dados para o servidor. Enviar email para o usuário e informar que a ocorrência foi inserida com sucesso.	Sim
CT05 – Registro de nova ocorrência com dados incorretos.	Sistema retorna mensagem de erro informando que não foi possível inserir nova ocorrência.	Sim

.O teste mostrou que o relatório tem boa usabilidade permitindo que uma nova ocorrência seja inserida de forma rápida, pois apresenta formulário automatizado na apresentação das informações. Como exemplo mostramos a Figura 33 que mostra a Tela Insere Ocorrência e como o usuário pode buscar um endereço digitando um trecho do nome da rua. Desta forma a elaboração de uma requisição de vistoria ou atendimento ficou mais rápida comparada com o preenchimento manual do formulário.

**Figura 33-Telas insere rua**



### 4.3 3º Experimento- Atendimento e geração de relatórios pelos agentes de campo.

Este experimento verificou as ações do sistema relacionadas à criação de relatórios das ocorrências. Os Casos de testes foram divididos com o intuito de verificar o caminho do relatório, desde sua inserção, registro fotográfico e disponibilização em formato PDF.

#### 4.4 Bateria de testes:

CT06 – Inserir relatório em ocorrência aberta com perfil de agente ou administrador.

CT07 – Realizar registro fotográfico.

CT09 – Gerar relatório em formato PDF.

**Tabela 13- Testes de acesso ao sistema 3º experimento**

<b>Ação do usuário</b>	<b>Resposta do sistema</b>	<b>Expectativa Atendida</b>
CT06 – Inserir relatório em ocorrência aberta com perfil de agente ou administrador.	Mandar os dados para o servidor. Enviar email para o usuário e informar que o relatório foi inserido com sucesso.	Sim
CT07 – Realizar registro fotográfico.	O sistema deve proporcionar registro da imagem no aparelho, armazenamento em servidor dedicado e recuperação da imagem para confecção de relatório em formato PDF.	Sim
CT09 – Gerar relatório em formato PDF.	O sistema deve possibilitar ao usuário relatório em formato PDF para que possa ser impresso e entregue ao	Sim

	solicitante	
--	-------------	--

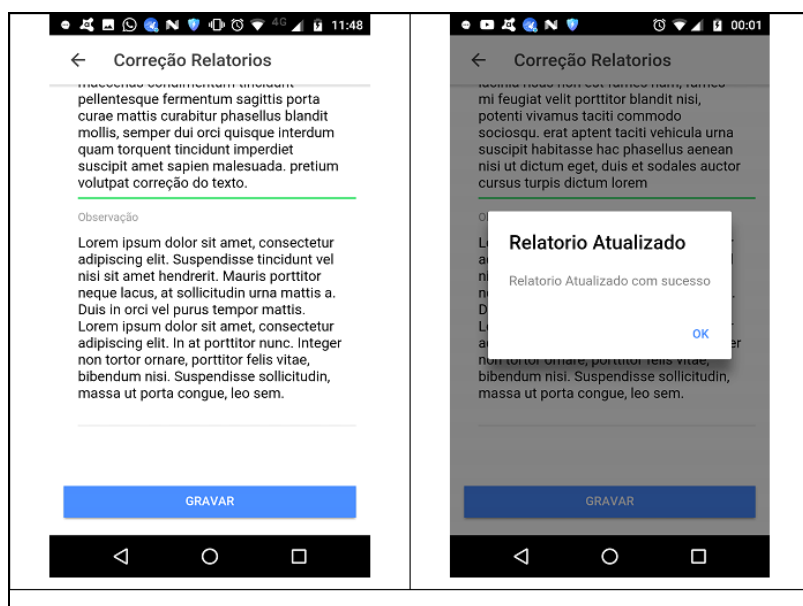
Na realização dos testes foi possível observar o desempenho da aplicação e necessidades para melhor utilização no trabalho diário. Verificou-se demora em abrir a pagina de nova ocorrência sendo que por este motivo foi realizada a seguinte mudança:

- Mudança na recuperação dos dados relacionados à ocorrência que serão armazenados no *Storage*<sup>21</sup> que é uma maneira fácil de armazenar pares chaves/valor e objetos JSON. O armazenamento usa uma variedade de mecanismos de armazenamento, escolhendo o melhor disponível, dependendo da plataforma, com o intuito de diminuir as requisições ao banco de dados visando melhor desempenho. Antes os dados sobre endereços e os que classificam a ocorrência (tipo de ocorrência) eram solicitados ao servidor no momento de carregamento da pagina e após a mudança são recuperados no momento do acesso do usuário ao sistema, mantendo-se durante toda a sessão.

Neste experimento pode-se observar a necessidade do agente em corrigir o texto da vistoria realizada, desta forma foi criada pagina para correção de relatório, onde o agente seleciona o relatório que deseja corrigir ou acrescentar informação e o sistema abre formulário para correção que demonstrou ser funcional e intuitivo.

A Figura 34 mostra as telas que o agente percorre para mudança no texto de um relatório.

**Figura 34-Atualização de relatório**



Fonte, Autor 2018.

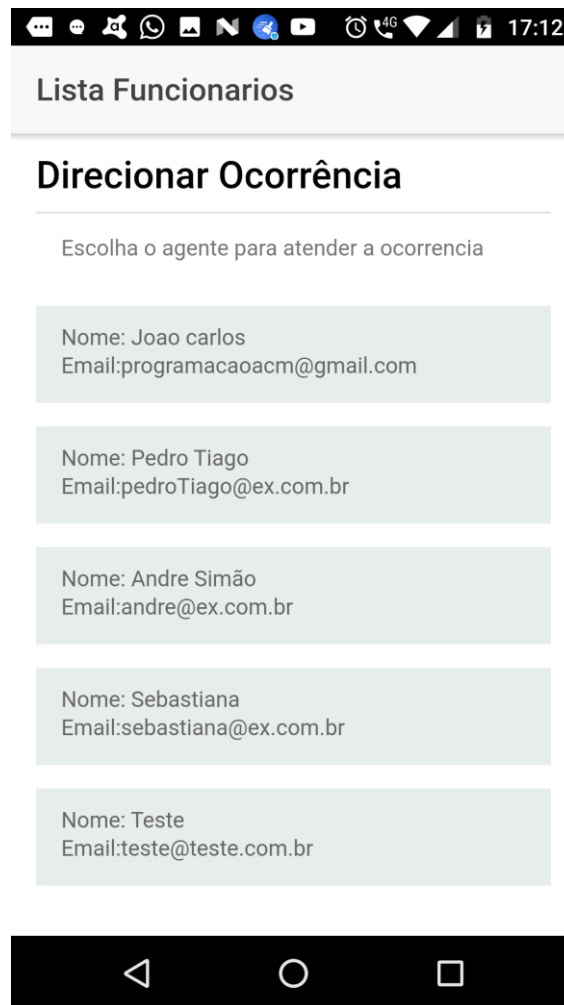
<sup>21</sup> Disponível em <https://ionicframework.com/docs/storage/>



Outra correção que foi realizada após os testes foi a criação de funcionalidade que permite que o cadastrador da ocorrência designe uma nova solicitação de vistoria para determinado agente de campo. Desta forma o agente somente visualizará as ocorrências que foram designadas para ele possibilitando maior organização na distribuição das tarefas diárias.

A Figura 35 mostra tela de designação de ocorrência.

**Figura 35-Tela de Despacho de ocorrência**

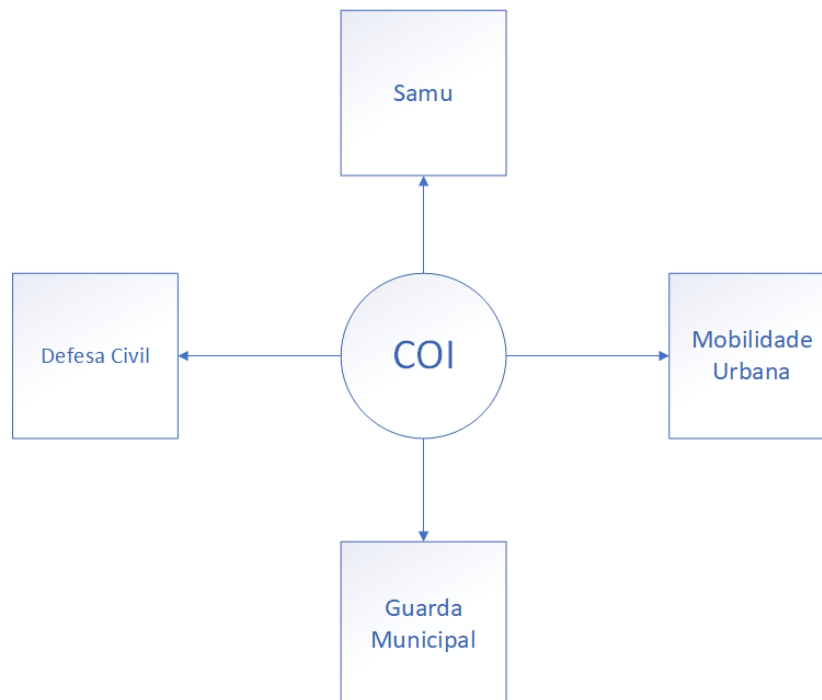


Fonte, Autor 2018.

## 5 - TRABALHOS FUTUROS

Este projeto foi pensado tendo em vista a integração de outros departamentos ao sistema, pois a Defesa Civil de São José dos Campos, que faz parte do Centro de Operações Integradas (COI) realiza muitos dos seus atendimentos de forma conjunta com outros órgãos da Prefeitura Municipal e a inclusão de outros órgãos ao sistema facilitaria os serviços por conta da troca rápida de informações.

**Figura 36– Trabalhos Futuros.**



Fonte, Autor 2018.

Para o sistema desenvolvido identificou-se melhorias que produzirão aperfeiçoamento nas funcionalidades do sistema:

- Elaboração de outros relatórios, como exemplo relatório de quantidade de ocorrências realizadas em determinada região num período de tempo.
- Inclusão de outros módulos gerando integração com os sistemas de outros setores do município.
- Incluir módulo para uso em desktop.
- Incluir módulo com mapas para uso web

Outra melhoria futura seria integrar a base de dados com órgãos do Estado que tem afinidade no tipo de atendimento oferecido, como exemplo Corpo de Bombeiros Militar que poderão usar as informações na elaboração de políticas públicas de prevenção de acidentes.