**Modelagem da aplicação de E-Gov**

**Aplicação smartphone para registro de ocorrências públicas**

A aplicação servirá para pessoas civis registrarem irregularidades que ocorrem no meio em que vivem. Irregularidades como carros estacionados em local proibido, assaltos em andamento, crimes contra o patrimônio público, entre outros crimes que são observados no dia-dia das pessoas e ficam sem ação de prosseguir a denúncia. O tratamento do registro da denúncia será efetuado por uma autoridade mais próxima. A autoridade será um policial, guarda municipal ou responsável de manter a order na região, que terá acesso especial ao aplicativo para visualizar e adotar as ocorrências, assim ser solucionadas e encaminhadas com o registro do BO (boletim de ocorrência). O registro da ocorrência será feito de forma simplificada e rápida para evitar prolongamento e talvez a perda de informação do que ocorreu. O usuário registrador irá acessar o aplicativo via *smartphone*, terá a opção de tirar até quatro fotos do ocorrido, descrever com poucas palavras a ocorrência, modificar a localização que foi capturada automaticamente pelo GPS do dispositivo e enviar para a central de registros. É previsto que este procedimento de registro da ocorrência tenha um tempo de duração de 5 min. Em comparação com o processo de denuncia atual serpa muito mais rápido e preciso, pois dará a localização exata do local e as autoridades conseguiram com o link fazerem a rota até o local.

**Arquitetura da aplicação**

O aplicativo será baseado à serviço. Sendo assim, a aplicação *front-end* (parte visual da aplicação: páginas, componentes visuais, estilo de cores e botões) ficará armazenada no dispositivo do usuário e as informações que populam o aplicativo ficarão armazenadas em um servidor disponibilizado na internet. As duas partes, assim chamas de camadas, se comunicarão via *HTTP*, utilizando seus métodos(*GET, SET, PUT, DELETE*). Porém, temos uma exceção neste arquitetura de serviço por termos o mecanismo de autenticação contido também no cliente(dispositivo *smartphone*) com suas credenciais.

O Front-End será desenvolvendo com as linguagens estáticas HTML e CSS para definir layouts e estilo das páginas. Para o controle das ações de tela será utilizado a linguagem Javascript. Para facilitar o desenvolvimento será utilizado framework IONIC desenvolvido em Javascript dedicado para criação de aplicativos para plataformas Android e IOS. O IONIC é uma evolução do Cordova, desenvolvido em java e javascript, a vantagem de utilizá-lo é podermos criar plugins e integrar às plataformas, no caso da plataforma Android desenvolvido em java e IOS desenvolvido em Objective-C. Para solução da autenticação, será desenvolvido o plugin que conterá a implementação da autenticação. O IONIC é integrado com o framework AngularJS e segue suas arquitetura no projeto inicial. O AngularJS é implementado em Javascript e facilita na controle das páginas com eventos, listagens, encapsulamento de objetos seguindo padrões do formato JSON, separação de camadas, acesso à camada HTTP, entre muitas facilidades disponíveis.

A back-end será desenvolvido e disponibilizado vias serviços REST. Para isso, será desenvolvido em java utilizando o frameword Spring.io e ficará sendo executada em um servidor de aplicação, como um Tomcat. O Spring possui métodos implícitos que faz o REST ser implementado com mais facilidade e de claro entendimento na manutenção. O armazenamento será feito em um base de dado, com a utilização do SGBD MySQL.

**Requisitos funcionais:**

* RF001 - O sistemas deverá conter cadastro de usuários
  + RN001 - Informações do usuário: nome, email, senha
  + RN002 - Confirmação de senha para registro do usuário
  + RN003 - O sistemas conterá três tipos de perfis:
    - Administrador - Criado somente por usuários administrador no ambiente administrador.
    - Registrador - Criado somente no acesso público, primeira na página inicial do aplicativo.
    - Autoridade - Criado somente por administradores no ambiente administrador
* RF002 - O sistemas deverá conter tela principal de acesso ao sistema
* RF003 - O sistemas deverá conter recuperação da senha do usuário
* RF004 - O sistemas deverá conter página principal do ambiente registrador
* RF005 - O sistemas deverá conter página principal do ambiente autoridade
* RF006 - O sistemas deverá conter página principal do ambiente administrador
* RF007 - O sistemas deverá conter cadastro de ocorrências
  + RN004 - Informações da ocorrência: descrição, localização e fotos
* RF008 - O sistemas deverá conter visualização da ocorrências
* RF009 - O sistemas deverá filtrar as ocorrências
  + RN005 - Tipos de filtros: por período, por status
* RF010 - O sistemas deverá conter atribuição de ocorrências
* RF011 - O sistemas deverá conter resolução de ocorrências
  + RN006 - Informação da resolução: conclusão da ocorrência

**Requisitos não funcionais:**

* RNF001 - Linguagem de programação Java para desenvolvimento da camada de serviço e implementação do plugin de autenticação
* RNF001 - Framework Spring para desenvolver métodos REST
* RNF001 - Maven para gerar o .war que será implamntado no servidor de aplicação
* RNF001 - Servidor de aplicação Tomcat para executar camada de serviço
* RNF001 - Linguagem de programação Javascript para desenvolvimento da camada de controle das páginas.
* RNF001 - Framework IONIC para desenvolvimento do aplicativo para plataforma Android
* RNF001 - AngularJS para fazer o controle dos dos evento e encapsulamento das informações da tela
* RNF001 - Plugman será utilizado para geração de plugins para plataforma Cordova (IONIC implementa)