Desenvolvimento de Aplicativo para Cadastro e Divulgação de Veículos Suspeitos

Gustavo Gonçalves Queiroz
Universidade Estadual do Oeste do Paraná – Unioeste
Foz do Iguaçu, Brasil
gusgonque@gmail.com

***Abstract*— The "Intelligent Wall" project is a partnership between Itaipu Binacional, Fundação Parque Tecnológico Itaipu - Brasil and the Federal Revenue Service, which aims to develop and implement intelligent technological solutions aimed at increasing security in border regions in Brazil.**

An application is being developed for Android/IOS cell phones, whose purpose is to associate data from suspicious vehicles to alert agents in the field, in the vicinity of the vehicle, through notifications on their cell devices, with the option to view vehicle information with a map, open the route application with the path to the indicated location, as well as register a new suspicious vehicle.

The application was developed using Flutter technology, and Firebase Messaging technology was used to send notifications. Several Flutter language libraries were used to determine the location of the device, show a map of the location of the suspicious vehicle and manage the notifications received.

***Resumo*— O projeto "Muralha Inteligente" é uma parceria entre a Itaipu Binacional, Fundação Parque Tecnológico Itaipu - Brasil e Receita Federal, que tem o objetivo de desenvolver e implementar soluções tecnológicas inteligentes visando o aumento da segurança em regiões de fronteira no Brasil.**

Está sendo desenvolvido um aplicativo para telefones celulares Android/IOS, cuja finalidade é associar dados de veículos suspeitos para alertar agentes em campo, nas proximidades do veículo, por meio de notificações em seus dispositivos celulares, com opção de visualizar as informações do veículo com um mapa, abrir o aplicativo de rotas direto para a localização indicada, como também registrar um novo veículo suspeito.

O aplicativo foi desenvolvido utilizando a tecnologia Flutter, e foi utilizado a tecnologia Firebase Cloud Messaging para envio das notificações. Foram utilizadas várias bibliotecas da linguagem Flutter para determinar a localização do dispositivo, mostrar um mapa da localização do veículo suspeito e gerenciar as notificações recebidas.

***Palavras-chave*— Aplicativo; Flutter; Firebase Messaging; Android; IOS.**

I. Introdução

O projeto Muralha Inteligente consiste em integrar o monitoramento por veículos remotamente controlados, e o monitoramento de veículos através de softwares de processamento de imagens, de modo a automatizar e melhorar a eficiência do processo manual atual.

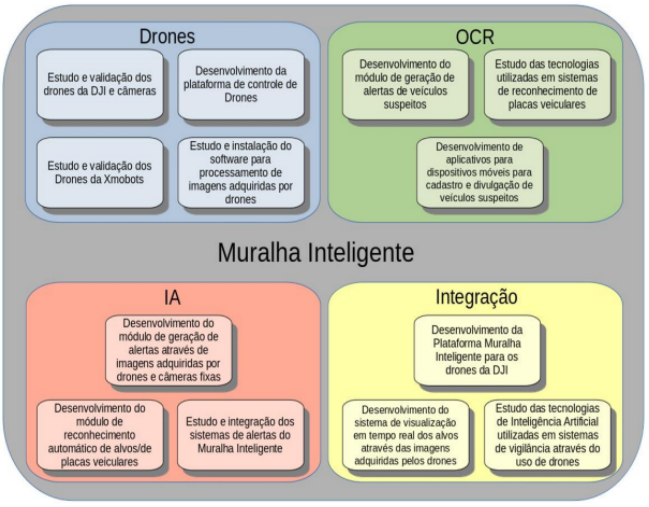

Fig 1. Módulos de pesquisas que compõem o projeto Muralha Inteligente.

Esse projeto faz parte de um conjunto de pesquisas que auxiliarão o desenvolvimento do projeto Muralha Inteligente (Fig. 1), especificamente na parte OCR (Optical Character Recognition ou Reconhecimento Ótico de Caracteres), o "desenvolvimento de módulo de geração de alertas de veículos suspeitos". Com isso está sendo desenvolvido um software e seu manual, que será integrado ao projeto Muralha Inteligente, capaz de cadastrar e divulgar dados de veículos considerados suspeitos.

II. Flutter

Flutter[1] é o kit de ferramentas de interface do usuário portátil do Google, com capacidade de criar softwares aplicativos compilados nativamente para dispositivos móveis, web e desktop a partir de uma única base de código.

A partir da plataforma "Pub.dev"[2], repositório de bibliotecas oficial para aplicações Dart e Flutter, foram usadas as bibliotecas "Awesome Notifications"[5], "http"[6], "flutter_map"[7], "url_launcher"[8], entre outras, para desenvolvimento do aplicativo

III. Firebase Cloud Messaging e como as Notificações Funcionam

O Firebase Cloud Messaging (FCM)[3] é uma solução para envio de mensagens entre plataformas que permite a entrega confiável de mensagens sem custo.

É utilizado o FCM para enviar mensagens de dados para todos os dispositivos telefone com o aplicativo instalado,

contendo as informações de um veículo suspeito recém adicionado, por uma aplicação servidor.

Então, cada telefone irá determinar sua distância do veículo, em background, e se essa distância for menor que a distância pré-determinada pelo desenvolvedor de cinco quilômetros, é criada uma notificação no telefone, pela biblioteca "Awesome Notifications", como apresentado na figura 2. Dessa forma, não é necessário saber para qual dispositivo enviar a notificação. Também é possível separar os telefones por grupos que serão enviados as mensagens, por exemplo, por níveis de sigilo.

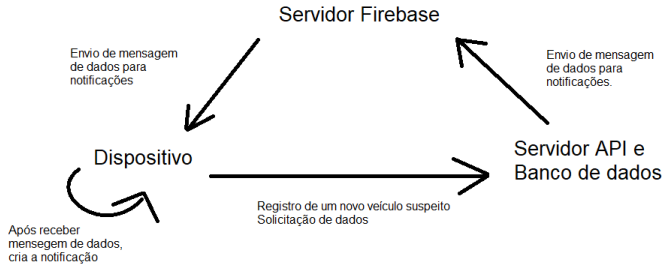

Fig 2. Representação do funcionamento do aplicativo.

IV. Telas

Foi decidido que o aplicativo teria quatro telas, sendo elas a tela principal do aplicativo, uma tela para a lista de veículos, uma tela contendo as informações de um veículo específico da lista e um formulário para registro de novos veículos.

A. Dashboard

É na tela inicial do aplicativo que o usuário acessa as telas de lista de veículos e de registro de um novo veículo, por meio de botões no inferior da tela. Ao acessar essa tela, a localização do dispositivo é requisitada.

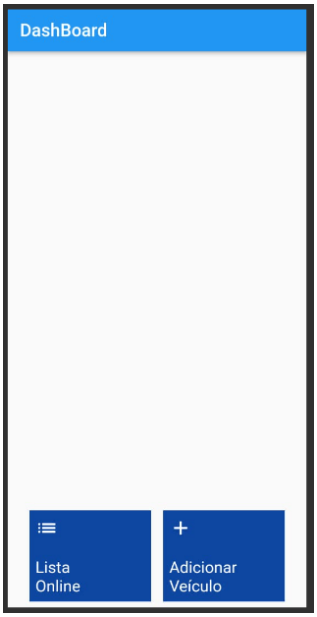

Fig 3. Tela do Dashboard do aplicativo.

B. Lista de Veículos

Ao acessar esta tela, o aplicativo solicita ao banco de dados uma lista com todos os veículos registrados e a apresenta. A lista está formada por quadros com o nome e descrição dos veículos. Ao clicar em um veículo específico, acessa a tela de informações daquele veículo. A lista é carregada de um banco de dados, utilizando funções da biblioteca "http" para carregar as informações de cada veículo.

Fig 4. Tela da Lista dos Veículos Suspeitos

C. Tela de Informações

Ao acessar a Tela de Informações, o aplicativo solicita ao banco de dados as informações de um veículo específico e as apresenta, juntamente com um mapa da localização do veículo (ícone vermelho), em relação ao usuário (ícone azul), como apresentado na figura 5. Inferior ao mapa, está um botão que solicita o telefone a abrir um aplicativo de rotas com as informações para iniciar uma rota para a localização do veículo suspeito.

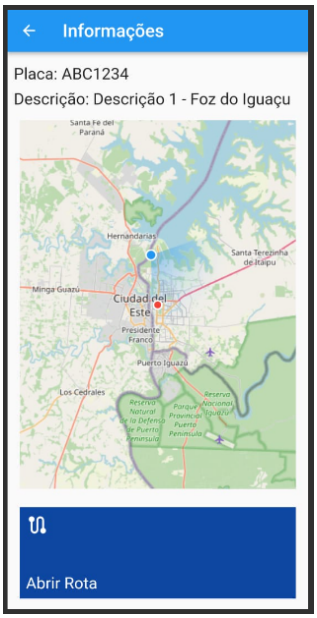

Fig 5. Tela das informações de um veículo suspeito em específico.

O mapa é feito utilizando a biblioteca "flutter_map", plotando o mapa fornecido pelo "Open Street Map"[4]. O método para abrir a rota no aplicativo de navegação é implementado com a biblioteca "url_launcher".

D. Registro de Veículo Suspeito

Ao acessar essa tela, é apresentado um formulário para cadastrar o veículo suspeito no banco de dados. O formulário apresenta dois campos, sendo estes para o usuário inserir a placa do veículo e sua descrição.

Fig 6. Formulário de cadastro de um veículo suspeito.

Assim como para as telas anteriores, são utilizadas funções da biblioteca "http" para implementar o método de enviar os dados do veículo para o banco de dados.

Até o momento, o aplicativo usaria a localização atual do dispositivo para efetuar o registro do veículo suspeito.

V. Testes e Discussões

Os testes foram realizados em dispositivos Android e IOS. É mostrado a seguir uma notificação no sistema operacional Android. Para fins de testes, no lugar de um banco de dados dedicado foi utilizado um repositório do Github, e uma API local para enviar dados para notificações.

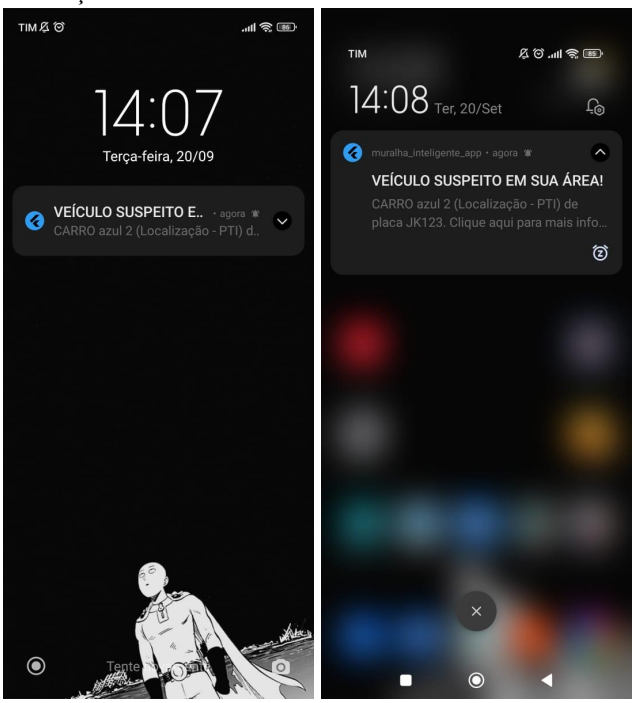

Fig. 7. Notificação na tela de bloqueio do telefone e a mesma Notificação expandida.

As notificações funcionam com o aplicativo fechado, em background, e com a tela do dispositivo bloqueada, como mostrado na figura 7. Vale notar que o aplicativo deve ser aberto uma vez, para ser salva a localização do dispositivo, antes de receber as notificações, e enquanto fechado ou em background, o aplicativo não terá acesso à localização atual, então usará a última localização salva enquanto aberto para realizar o método de criar ou não a notificação.

Ao testar em diferentes localidades, com dados de veículos em distâncias maiores que a definida, não é recebida a notificação.

Ao estudar artigos referentes a aplicativos multiplataformas, foi escolhida a ferramenta Flutter por possibilitar o desenvolvimento do aplicativo para as plataformas Android e IOS, compilação nativa, os aplicativos serem escritos na linguagem Dart, linguagem orientada a objetos com sintaxe estilo C desenvolvida pela Google, e ter um grande número de bibliotecas disponíveis, para diversas finalidades.

Caso continuado o desenvolvimento do aplicativo, será adicionada a opção de determinar a localização do veículo

suspeito, na tela de Registro, além do desenvolvimento de recursos para melhorar a qualidade do aplicativo como aperfeiçoamento da interface e

VI. Conclusão

É esperado que o aplicativo sirva de ajuda para agentes nas fronteiras do país, visando divulgar e notificar sobre a presença de veículos suspeitos na sua proximidade.

Após testes, foi confirmado que o aplicativo funciona com o dispositivo em diferentes localidades, desde que a distância seja menor que cinco quilômetros.

Referências

[1] Flutter. Disponível em: https://flutter.dev/. Acesso em 27/09/2022.

[2] Pub.dev. Disponível em: https://pub.dev. Acesso em 27/09/2022.

[3] Documentação do Firebase Cloud Messaging. Disponível em: https://firebase.google.com/docs/cloud-messaging. Acesso em 27/09/2022.

[4] Open Street Map. Disponível em: https://www.openstreetmap.org/. Acesso em 27/09/2022.

[5] Biblioteca "Awesome Notifications". Disponível em https://pub.dev/packages/awesome_notifications. Acesso em 27/09/2022.

[6] Biblioteca "http". Disponível em https://pub.dev/packages/http. Acesso em 27/09/2022

[7] Biblioteca "flutter_map". Disponível em https://pub.dev/packages/flutter_map. Acesso em 27/09/2022.

[8] Biblioteca "url_launcher". Disponível em https://pub.dev/packages/url_launcher. Acesso em