



4º ANO (417) TÉCNICO EM INFORMÁTICA INTEGRADO AO ENSINO MÉDIO

ARTHUR HENRIQUE MELO DE ALMEIDA
EDUARDO FELICIANO SANTOS ROCHA
HENRIQUE DE OLIVEIRA MENDES PEREIRA
KAYKY DA SILVA MORENO
LUCAS RAFAEL DA SILVA MENDES
MARIANA ALVES FERNANDES
NATHAN GOMES DA SILVA

SISTEMA DE CHAMADA DE EMERGÊNCIA DO SUS - DOCUMENTAÇÃO

Cubatão, 2024

INTRODUÇÃO

O nosso projeto é um aplicativo móvel desenvolvido através da framework Flutter, que é baseada na linguagem de programação Dart e bibliotecas. Como IDE (ambiente de desenvolvimento integrado) foi utilizado o Visual Studio Code.

O aplicativo tem como principal foco agilizar o atendimento de urgência do SAMU, facilitando a localização e o acesso a dados essenciais do usuário para um suporte ideal.

1. Objetivo Geral

Propor melhorias para o sistema de chamada de emergência do SAMU

2. Objetivos específicos

- Compreender o sistema de chamada de emergência do SAMU
- Analisar o seu funcionamento e identificar pontos positivos e negativos
- Elaborar proposta de implementação de melhorias
- Identificar dificuldades na implementação de melhorias

APLICAÇÕES UTILIZADAS

1. Flutter

a. Sobre o Flutter

O Flutter é um framework multiplataforma desenvolvido pela Google e baseado em Dart, que combina as vantagens de linguagens robustas como Java e adaptabilidade do JavaScript. Os blocos de código representam widgets que formarão a interface do usuário.

O Dart é uma linguagem client-side e estruturada segundo o paradigma orientado a objetos, como o Java, buscando ser flexível para servir como base para diversos frameworks de desenvolvimento de aplicativos, embora até o momento o Flutter seja o único que utilize.

Sua primeira versão possuía outro nome, “Sky”, e surgiu com a ideia de construir melhores interfaces para o mobile. Lançado inicialmente em 2014 e apresentado ao público em 2015, estreou no Google IO 2017, já integrado ao Firebase, o sistema de banco de dados unificado da Google. No Google IO 2018 foi lançada a versão 1.0. Ele não traduz o código para o respectivo elemento, mas implementa através de um motor de renderização próprio. Possui compilados em ARM nativ

Possui Hot Reload, ou seja, testa em tempo real, não sendo necessário aguardar a compilação para visualizar o resultado.

A partir de 2019 se tornou uma framework portátil para mobile, web, desktop e embarcados, sendo multiplataforma. Em 2022 o Google lançou o Flutter 3, a versão mais recente da framework que abre espaço para a criação de programas para Linux, macOS e suporte para o Apple MI.

b. Pré-requisitos

- Windows 7, 8, 10 ou 11
- Git instalado no sistema

c. Instalação

1. Baixar a SDK do Flutter

Acesse o site oficial e clique em "Download" para baixar a SDK do Flutter como um arquivo .zip. É importante ressaltar que a SDK do Dart vem junto na instalação, logo não é necessário instalá-la separadamente.

SDK é um kit de desenvolvimento de software, um conjunto de ferramentas oferecidas para os desenvolvedores como depuradores, compiladores e bibliotecas para a criação de códigos

Install the Flutter SDK

To install the Flutter SDK, you can use the VS Code Flutter extension or download and install the Flutter bundle yourself.

[Use VS Code to install](#) [Download and install](#)

Use VS Code to install Flutter

To install Flutter using these instructions, verify that you have installed [Visual Studio Code 1.77](#) or later and the [Flutter extension for VS Code](#).

Prompt VS Code to install Flutter

1. Launch VS Code.
2. To open the **Command Palette**, press **Control** + **Shift** + **P**.
3. In the **Command Palette**, type `flutter`.
4. Select **Flutter: New Project**.
5. VS Code prompts you to locate the Flutter SDK on your computer.
 1. If you have the Flutter SDK installed, click **Locate SDK**.
 2. If you do not have the Flutter SDK installed, click **Download SDK**.

This option sends you the Flutter install page if you have not installed Git for Windows as directed in the [development tools prerequisites](#).
6. When prompted **Which Flutter template?**, ignore it. Press **Esc**. You can create a test project after checking your development setup.

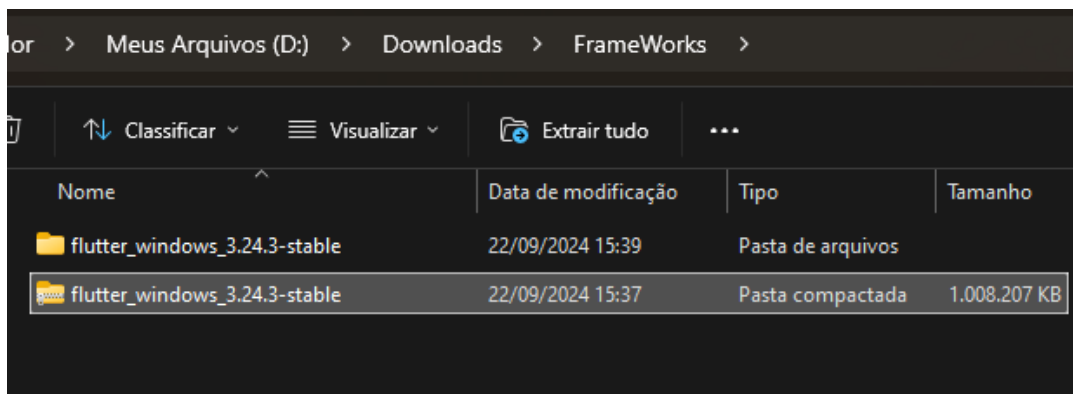
Download the Flutter SDK

1. When the **Select Folder for Flutter SDK** dialog displays, choose where you want to install Flutter.

VS Code places you in your user profile to start. Choose a different location.
Consider `%USERPROFILE%` or `C:\dev`.

2. Extrair para o disco local

Com o arquivo já baixado, extraia para o Disco Local C:

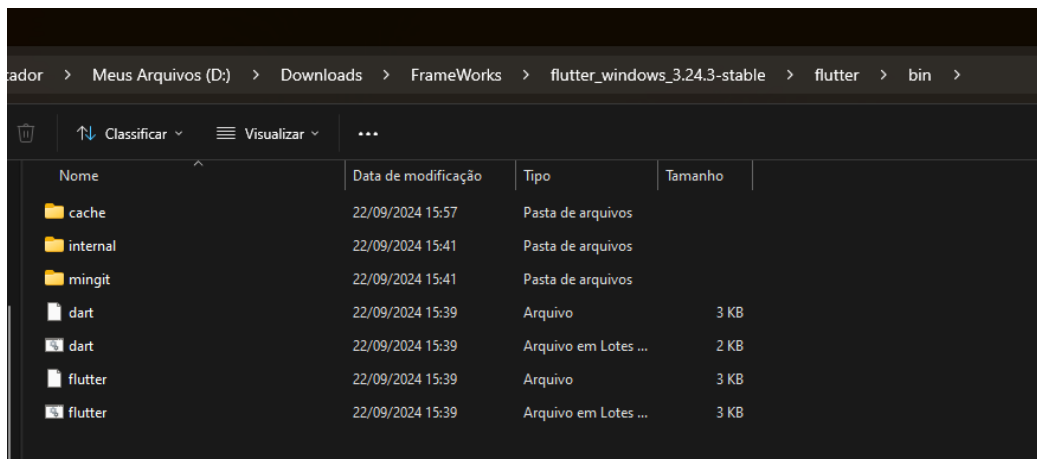


3. Copie o caminho da pasta "flutter"

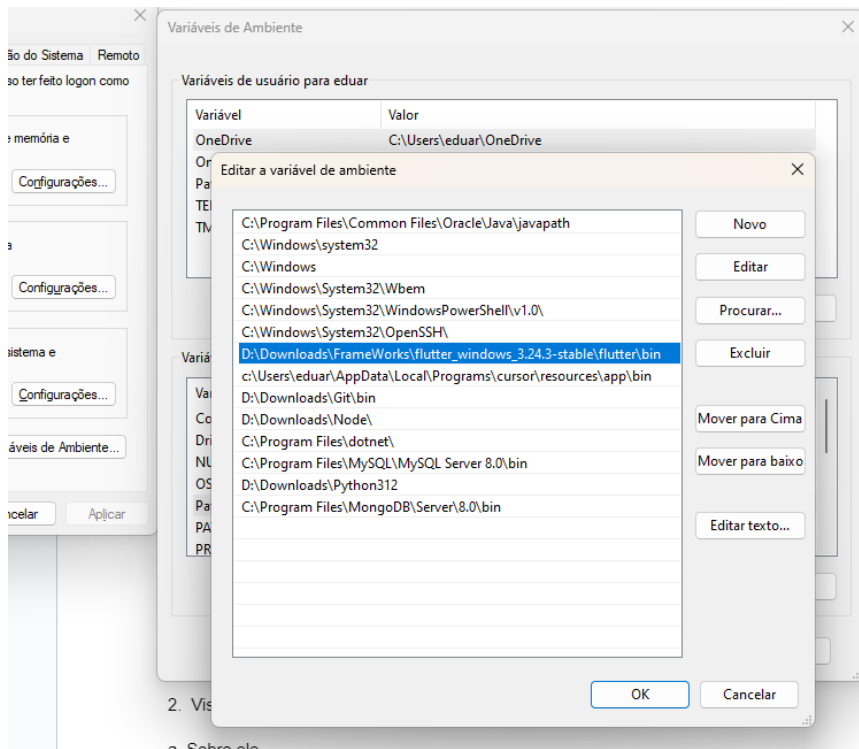
Para que os comandos funcionem diretamente pelo terminal é necessário inserir o diretório do "flutter/bin" nas variáveis de ambiente do Windows, PATH. Em seguida, abra o editor de variáveis de ambiente do sistema

PATH é uma variável de ambiente no Windows que especifica uma lista de diretórios onde os arquivos executáveis (.exe) estão localizados. Ao inserir um comando no terminal (CMD) o sistema busca no diretório um executável com o nome correspondente, caso não seja encontrado, busca no PATH.

Adicionar o Git ao Path permite que os comandos sejam realizados a partir do próprio terminal, sem precisar estar no diretório onde ele foi instalado.



4. Adicione o caminho da pasta flutter na variável PATH



5. Verifique se a instalação foi feita corretamente

Para verificar se a instalação foi realizada com sucesso, utilize o comando `-flutter doctor`.

```

Doctor summary (to see all details, run flutter doctor -v):
[✓] Flutter (Channel stable, 3.24.3, on Microsoft Windows [vers#o 10.0.22631.4460], locale pt-BR)
[✓] Windows Version (Installed version of Windows is version 10 or higher)
[!] Android toolchain - develop for Android devices (Android SDK version 35.0.0)
    X cmdline-tools component is missing
      Run `path/to/sdkmanager --install "cmdline-tools;latest"`
      See https://developer.android.com/studio/command-line for more details.
    X Android license status unknown.
      Run `flutter doctor --android-licenses` to accept the SDK licenses.
      See https://flutter.dev/to/windows-android-setup for more details.
[X] Chrome - develop for the web (Cannot find Chrome executable at .\Google\Chrome\Application\chrome.exe)
    ! Cannot find Chrome. Try setting CHROME_EXECUTABLE to a Chrome executable.
[X] Visual Studio - develop Windows apps
    X Visual Studio not installed; this is necessary to develop Windows apps.
      Download at https://visualstudio.microsoft.com/downloads/.
      Please install the "Desktop development with C++" workload, including all of its default components
[✓] Android Studio (version 2024.1)
[✓] VS Code (version 1.95.3)
[✓] Connected device (2 available)
[✓] Network resources

! Doctor found issues in 3 categories.
  
```

Caso esteja instalado corretamente, instale o editor de código, no caso o Visual Studio Code.

2. Visual Studio Code

a. Sobre ele

O Visual Studio Code é um editor de código aberto desenvolvido pela Microsoft

disponível para Windows, Mac e Linux. Não deve ser confundido com o Visual Studio, também criado pela Microsoft e dedicado ao .NET Framework e linguagens como C e C#.

O VS Code possui funcionalidades mais simples e é mais customizável, podendo editar programas e script de diferentes linguagens de acordo com as extensões baixadas.

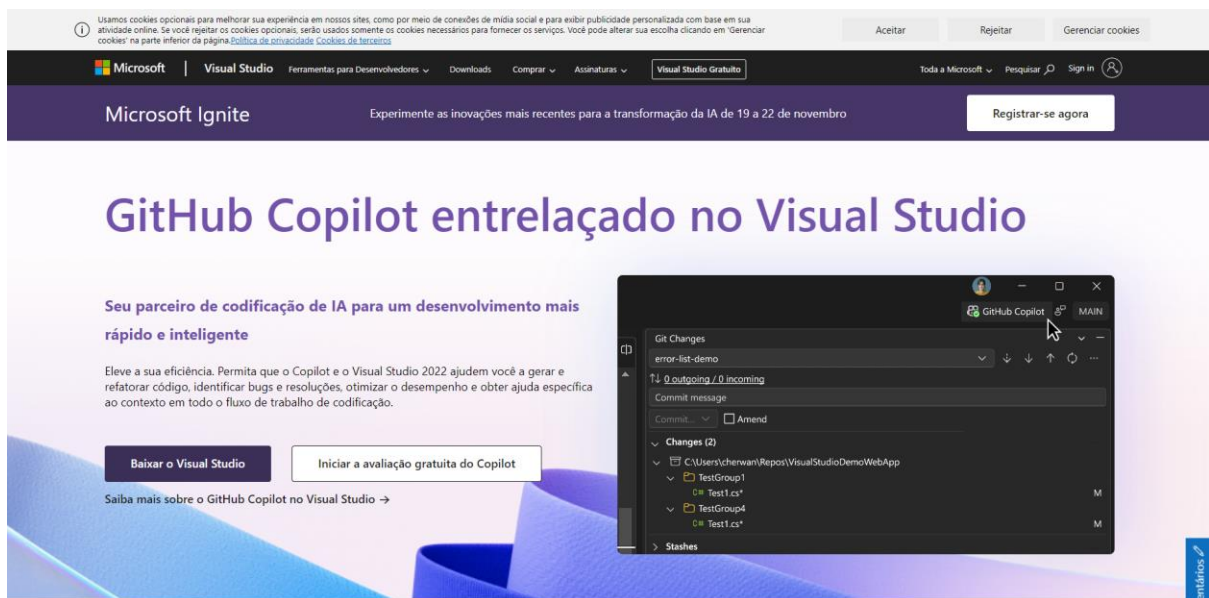
b. Pré-requisitos

- Sistema Operacional: Windows 10, 8.1 ou 7 (32 ou 64 bit)
- Espaço em disco: 50 GB de espaço livre recomendado
- RAM: 8 GB de RAM recomendado (mínimo de 4GB)

c. Instalação

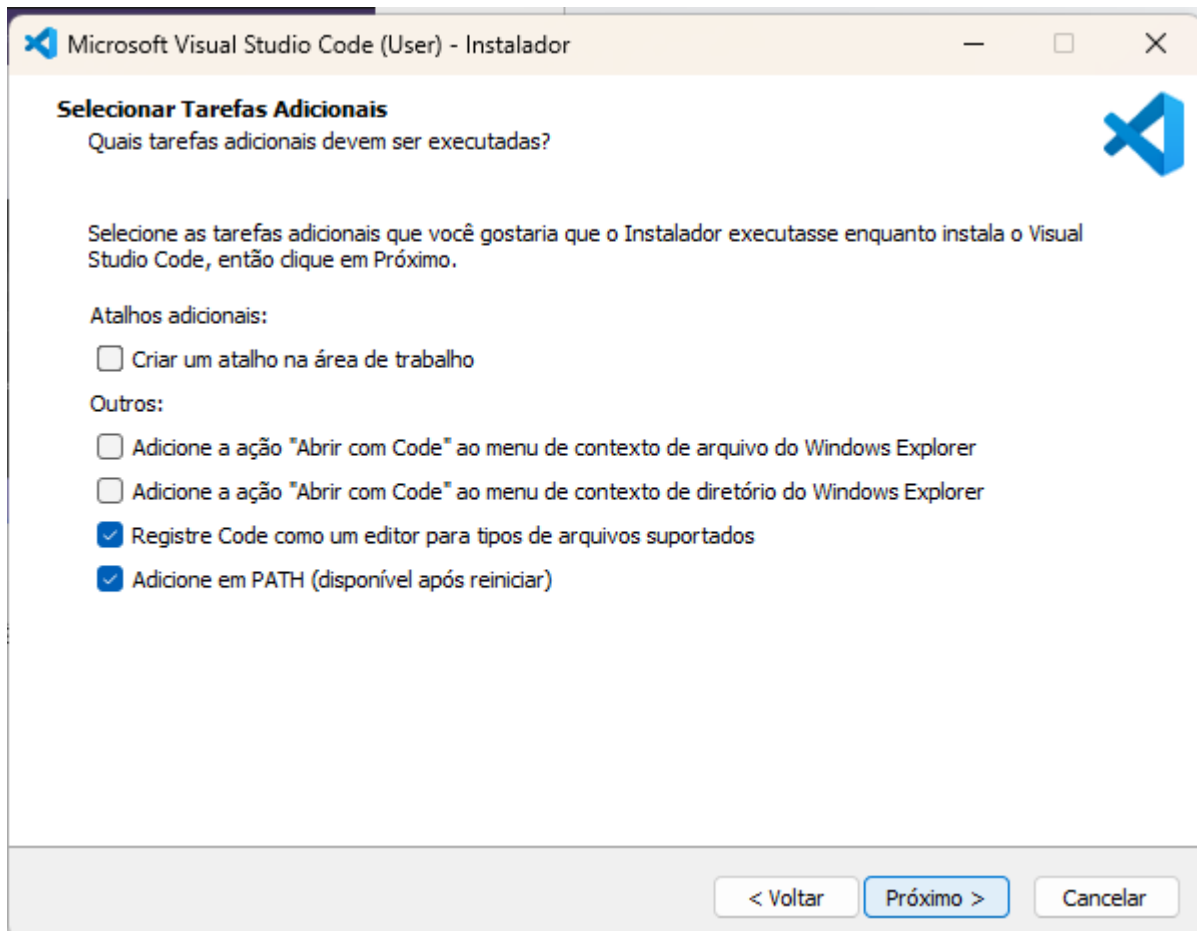
1. Download do Instalador

No site <https://visualstudio.microsoft.com/pt-br/> baixar o instalador do Visual Studio Code.



2. Executar o Instalador

Abrir o arquivo baixado do instalador e definir a pasta em que será instalado, por definição uma subpasta na AppData

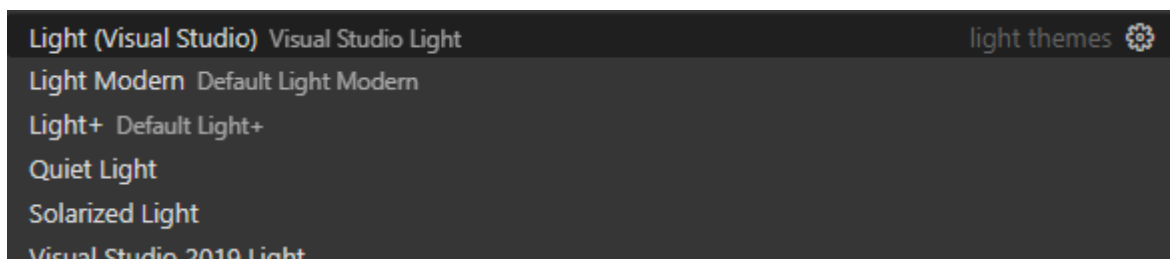


3. Configurar a instalação

Definir as opções de “Tarefas Adicionais” segundo a necessidade de instalação, nesse caso, as já selecionadas são suficientes.

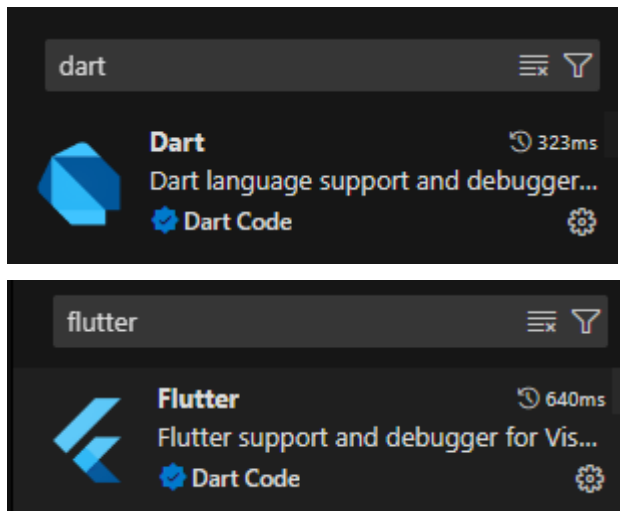
4. Configuração inicial

Denir o tema da IDE (branco ou escuro) e sincronizar com outros dispositivos são as opções disponíveis para essa etapa



5. Baixar as extensões de Flutter e Dart

Na aba de “Extensões” (ou apertando Ctrl+Shift+X), pesquisar “Flutter” e “Dart” e instalar ambas as extensões



Pronto! O seu Visual Studio Code já está instalado e devidamente configurado

CONCEITOS ESSENCIAIS

1. Widgets

Os widgets são os elementos básicos da interface gráfica do projeto. Exemplos: Botão, TextBox, Sliders, entre outros. Apesar de básicos, são fundamentais para a interação do usuário com o aplicativo.

2. Programação Orientada a Objetos

É um paradigma de programação baseado no conceito de "objetos". Os objetos podem conter dados ou códigos.

A POO possui conceitos como encapsulamento, herança, polimorfismo e abstração, necessários para uma melhor forma de desenvolver um código.

O principal recurso da POO é a fácil reutilização do código e facilidade em modificá-lo caso necessário. Algumas linguagens populares que utilizam a POO são Java, C++, C# e Python.

3. Classes

As classes são modelos dos quais objetos são criados. Elas definem atributos e métodos padrões para todos os objetos gerados a partir dela.

METODOLOGIA - DESENVOLVIMENTO

-O que vamos usar

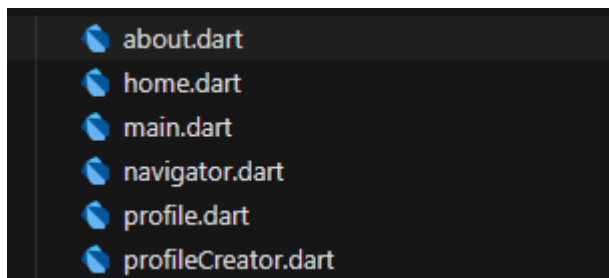
Bibliotecas

```
cupertino_icons: ^1.0.8
url_launcher: ^6.3.1
geolocator: ^13.0.1
shared_preferences: ^2.3.2
image_picker: ^1.1.2
```

Dos icons usados, para envio do sms, para pegar latitude e longitude, para compartilhar os dados com a outra página, para enviar a imagem para o perfil

PÁGINAS

Cada arquivo representa uma página diferente.



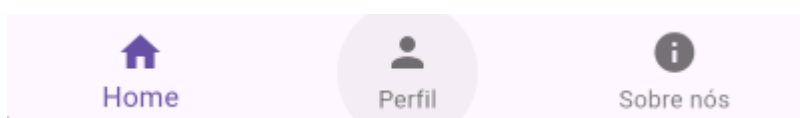
CÓDIGO

Permissões da biblioteca de geoprocessamento, necessário colocar no AndroidManifest

```
home.dart AndroidManifest.xml X
android > app > src > main > AndroidManifest.xml
1 <manifest xmlns:android="http://schemas.android.com/apk/res/android">
2   <application
6     <activity
19       <meta-data
21         android:resource="@style/NormalTheme"
22       />
23     <intent-filter>
24       <action android:name="android.intent.action.MAIN" />
25       <category android:name="android.intent.category.LAUNCHER" />
26     </intent-filter>
27   </activity>
28   ← Don't delete the meta-data below.
29   This is used by the Flutter tool to generate GeneratedPluginRegistrant.java →
30   <meta-data
31     android:name="flutterEmbedding"
32     android:value="2" />
33 </application>
34 ← Required to query activities that can process text, see:
35 https://developer.android.com/training/package-visibility and
36 https://developer.android.com/reference/android/content/Intent#ACTION_PROCESS_TEXT.
37
38 In particular, this is used by the Flutter engine in io.flutter.plugin.text.ProcessTextPlugin. →
39 <queries>
40   <intent>
41     <action android:name="android.intent.action.PROCESS_TEXT" />
42     <data android:mimeType="text/plain" />
43   </intent>
44 </queries>
45 <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
46 <uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION" />
47 <uses-permission android:name="android.permission.ACCESS_BACKGROUND_LOCATION" />
48 </manifest>
49
```

Barra de navegação do app

```
navigator.dart X
lib > navigator.dart > ...
1  import 'package:flutter/material.dart';
2  import 'about.dart';
3  import 'profile.dart';
4  import 'home.dart';
5
6  class MainNavigator extends StatefulWidget {
7    final List<String> usuario;
8
9    const MainNavigator({super.key, required this.usuario});
10
11  @override
12  MainNavigatorState createState() => MainNavigatorState();
13 }
14
15 class MainNavigatorState extends State<MainNavigator> {
16   int selectedIndex = 0;
17
18   @override
19   Widget build(BuildContext context) {
20     List<Widget> screens = [
21       const Home(),
22       ProfilePage(usuario: widget.usuario),
23       About(),
24     ];
25
26     return SafeArea(
27       child: Scaffold(
28         body: screens[selectedIndex],
29         bottomNavigationBar: BottomNavigationBar(
30           items: const [
31             BottomNavigationBarItem(icon: Icon(Icons.home), label: 'Home'),
32             BottomNavigationBarItem(icon: Icon(Icons.person), label: 'Perfil'),
33             BottomNavigationBarItem(
34               icon: Icon(Icons.info), label: 'Sobre nós'), // BottomNavigationBarItem
35           ],
36           onTap: (i) {
37             setState(() {
38               selectedIndex = i;
39             });
40           },
41           currentIndex: selectedIndex,
42         ), // BottomNavigationBar
43       ), // Scaffold
44     ); // SafeArea
45   }
46 }
```



Configuração da barra

```

class Page extends StatelessWidget {
  const Page({super.key});

  @override
  Widget build(BuildContext context) {
    return MaterialApp(
      home: MainNavigator(usuario: []),
      debugShowCheckedModeBanner: false,
    ); // MaterialApp
  }
}

```

Estruturação da página “sobre nós”

```

children: [
  const Center(
    child: Icon(
      Icons.emergency,
      size: 80,
      color: Colors.white,
    ), // Icon
  ), // Center
  const SizedBox(height: 20),
  Card(
    elevation: 5,
    shape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(15),
    ), // RoundedRectangleBorder
    child: Padding(
      padding: const EdgeInsets.all(20.0),
      child: Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          const Text(
            'Sistema de Chamada de Emergência',
            style: TextStyle(fontSize: 22, fontWeight: FontWeight.bold, color: Color(0xFF0F9652)),
          ), // Text
          const SizedBox(height: 15),
          const Text(
            'Este aplicativo é um sistema de chamada de emergência de ambulância, projetado para garantir sua segurança e facilitar o atendimento',
            style: TextStyle(fontSize: 16, height: 1.5),
          ), // Text
          const SizedBox(height: 20),
          const Text(
            'Funcionalidades:',
            style: TextStyle(fontSize: 18, fontWeight: FontWeight.bold, color: Color(0xFF0F9652)),
          ), // Text
          const SizedBox(height: 10),
          _buildFeatureItem('Coleta de dados para sua segurança'),
          _buildFeatureItem('Ativação rápida de chamada'),
          _buildFeatureItem('Registro de dados do ocorrido'),
          _buildFeatureItem('Captura de localização em tempo real'),
        ],
      ), // Column
    ), // Card
  ],
), // Column

```

```

class AboutState extends State<About> {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: const Color(0xFF6EEEBBA),
      appBar: AppBar(
        backgroundColor: const Color.fromARGB(255, 15, 150, 82),
        elevation: 0,
        title: const Text('Sobre o Aplicativo', style: TextStyle(color: Colors.white)),
      ), // AppBar
      body: SafeArea(
        child: SingleChildScrollView(
          child: Padding(
            padding: const EdgeInsets.all(20.0),
            child: Column(
              crossAxisAlignment: CrossAxisAlignment.start,
              children: [
                const Center(
                  child: Icon(
                    Icons.emergency,
                    size: 80,
                    color: Colors.white,
                  ), // Icon
                ), // Center
                const SizedBox(height: 20),
                Card(
                  elevation: 5,
                  shape: RoundedRectangleBorder(
                    borderRadius: BorderRadius.circular(15),
                  ), // RoundedRectangleBorder
                  child: Padding(
                    padding: const EdgeInsets.all(20.0),
                    child: Column(

```



Sistema de Chamada de Emergência

Este aplicativo é um sistema de chamada de emergência de ambulância, projetado para garantir sua segurança e facilitar o atendimento médico rápido.

Funcionalidades:

- ✓ Coleta de dados para sua segurança
- ✓ Ativação rápida de chamada
- ✓ Registro de dados do ocorrido
- ✓ Captura de localização em tempo real



Home



Perfil



Sobre nós

Dados da página de perfil

```
class ProfilePageState extends State<ProfilePage> {
  String _nome = '';
  String _cpf = '';
  String _telefone = '';
  String _observacao = '';

  @override
  void initState() {
    super.initState();
    _loadUserData();
  }

  Future<void> _loadUserData() async {
    SharedPreferences prefs = await SharedPreferences.getInstance();
    setState(() {
      _nome = prefs.getString('nome') ?? '';
      _cpf = prefs.getString('cpf') ?? '';
      _telefone = prefs.getString('telefone') ?? '';
      _observacao = prefs.getString('observacao') ?? '';
    });
  }
}
```

Estruturação página de perfil

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: const Color(0xFF6EEBBA),
    appBar: AppBar(
      backgroundColor: const Color.fromARGB(255, 15, 150, 82),
      elevation: 0,
      title: const Text('Perfil do Usuário',
        style: TextStyle(color: Colors.white)), // Text
    ), // AppBar
    body: SafeArea(
      child: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(20.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.center,
            children: [
              const CircleAvatar(
                radius: 60,
                backgroundColor: Colors.white,
                child: Icon(
                  Icons.person,
                  size: 80,
                  color: Color(0xFF6EEBBA),
                ), // Icon
              ), // CircleAvatar
              const SizedBox(height: 20),
              Card(
                elevation: 5,
                shape: RoundedRectangleBorder(
                  borderRadius: BorderRadius.circular(15),
                ), // RoundedRectangleBorder
                child: Padding(
                  padding: const EdgeInsets.all(20.0),
                  child: Column(
                    children: [
                      _buildInfoItem('Nome', _nome),
                      _buildInfoItem('CPF', _cpf),
                      _buildInfoItem('Telefone', _telefone),
                      _buildInfoItem('Observações', _observacao),
                    ],
                  ),
                ),
              ),
            ],
          ),
        ),
      ),
    ),
  );
}
```

Alterar dados

```
const SizedBox(height: 20),
ElevatedButton(
  onPressed: () {
    Navigator.of(context).push(
      MaterialPageRoute(
        builder: (context) =>
          ProfileCreatorPage(usuario: widget.usuario),
      ), // MaterialPageRoute
    );
  },
  style: ElevatedButton.styleFrom(
    backgroundColor: const Color.fromARGB(255, 15, 150, 82),
    foregroundColor: Colors.white,
    padding: const EdgeInsets.symmetric(
      horizontal: 30, vertical: 15), // EdgeInsets.symmetric
    shape: RoundedRectangleBorder(
      borderRadius: BorderRadius.circular(10),
    ), // RoundedRectangleBorder
  ),
  child: Text('Alterar Dados'),
), // ElevatedButton
```

Perfil do Usuário



Nome:

CPF:

Telefone:

Observações:

Alterar Dados



Home



Perfil



Sobre nós

Página de alteração de dados


```
class ProfileCreatorPageState extends State<ProfileCreatorPage> {
  final _textNome = TextEditingController();
  final _textCPF = TextEditingController();
  final _textTelefone = TextEditingController();
  final _textObservacao = TextEditingController();
  File? _image;
  final picker = ImagePicker();

  @override
  void initState() {
    super.initState();
    _loadUserData();
    // Preencher os campos com os dados existentes do usuário
    _textNome.text = widget.usuario.isNotEmpty ? widget.usuario[0] : '';
    _textCPF.text = widget.usuario.length > 1 ? widget.usuario[1] : '';
    _textTelefone.text = widget.usuario.length > 2 ? widget.usuario[2] : '';
    _textObservacao.text = widget.usuario.length > 3 ? widget.usuario[3] : '';
  }
}
```



Criação de Perfil



 Nome

 CPF

 Telefone

Observações -



Salvar Perfil

Cancelar

Processamento dos dados salvos

```
Future<void> _loadUserData() async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  setState(() {
    _textNome.text = prefs.getString('nome') ?? '';
    _textCPF.text = prefs.getString('cpf') ?? '';
    _textTelefone.text = prefs.getString('telefone') ?? '';
    _textObservacao.text = prefs.getString('observacao') ?? '';
    String? imagePath = prefs.getString('imagePath');
    _image = File(imagePath!);
  });
}

Future<void> _saveUserData() async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  await prefs.setString('nome', _textNome.text);
  await prefs.setString('cpf', _textCPF.text);
  await prefs.setString('telefone', _textTelefone.text);
  await prefs.setString('observacao', _textObservacao.text);
  if (_image != null) {
    await prefs.setString('imagePath', _image!.path);
  }

  // Atualiza a lista de usuário
  widget.usuario
    ..clear()
    ..addAll([_textNome.text, _textCPF.text, _textTelefone.text, _textObservacao.text]);
}
```

```
    await _saveUserData();

    // Retorna os dados atualizados
    Navigator.of(context).pop(widget.usuario);
  },
```

Envio imagem

```
Future getImage(ImageSource source) async {
  final pickedFile = await picker.pickImage(source: source);

  setState(() {
    if (pickedFile != null) {
      _image = File(pickedFile.path);
    }
  });
}
```

Estruturação da página de alteração de perfil

```

Card(
  elevation: 5,
  shape: RoundedRectangleBorder(
    borderRadius: BorderRadius.circular(15),
  ), // RoundedRectangleBorder
  child: Padding(
    padding: const EdgeInsets.all(20.0),
    child: Column(
      children: [
        _buildTextField(_textNome, 'Nome', Icons.person),
        const SizedBox(height: 20),
        _buildTextField(_textCPF, 'CPF', Icons.credit_card),
        const SizedBox(height: 20),
        _buildTextField(_textTelefone, 'Telefone', Icons.phone),
        const SizedBox(height: 20),
        _buildTextField(
          _textObservacao, 'Observações -', Icons.note,
          maxLines: 3, ),
      ],
    ),
  ),
)

```

```

@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: const Color(0xFF6EEBBA),
    appBar: AppBar(
      backgroundColor: const Color.fromARGB(255, 15, 150, 82),
      elevation: 0,
      title: const Text('Criação de Perfil',
        style: TextStyle(color: Colors.white)), // Text
    ), // AppBar
    body: SafeArea(
      child: SingleChildScrollView(
        child: Padding(
          padding: const EdgeInsets.all(20.0),
          child: Column(
            crossAxisAlignment: CrossAxisAlignment.stretch,

```

Página Home (Principal)

Inicialização e carregamento dos dados

```

class _HomeState extends State<Home> {
  String dropdownvalue = 'Localização atual';
  String? latitude, longitude;
  String nome = '', cpf = '', telefone = '', observacao = '';

  @override
  void initState() {
    super.initState();
    _loadUserData();
  }
}

```

```

Future<void> _loadUserData() async {
  SharedPreferences prefs = await SharedPreferences.getInstance();
  setState(() {
    nome = prefs.getString('nome') ?? '';
    cpf = prefs.getString('cpf') ?? '';
    telefone = prefs.getString('telefone') ?? '';
    observacao = prefs.getString('observacao') ?? '';
  });
}

```

Pegando dados de geoprocessamento

```

Future<void> pegarPosicao() async {
  // Solicitar permissão de localização
  LocationPermission permission = await Geolocator.requestPermission();
  if (permission == LocationPermission.denied ||
      permission == LocationPermission.deniedForever) {
    // Mostrar uma mensagem de erro ou guia para o usuário
    print('Permissão de localização negada');
    return;
  }

  try {
    Position posicao = await Geolocator.getCurrentPosition();
    setState(() {
      latitude = posicao.latitude.toString();
      longitude = posicao.longitude.toString();
    });

    // Certifique-se de que os valores de latitude e longitude não sejam null
    if (latitude != null && longitude != null) {
      String mensagem = '''
Preciso de ajuda;
Nome: $nome
CPF: $cpf
Telefone: $telefone
Observação: $observacao
Latitude: $latitude
Longitude: $longitude
''';

```

Envio da mensagem (Substitui os ***** por um número de telefone)

```

final Uri smsUri = Uri(
  scheme: 'sms',
  path: '*****',
  queryParameters: {'body': mensagem},
);

// Tentar enviar o SMS e verificar se é possível lançar o URL
if (await canLaunchUrl(smsUri)) {
  await launchUrl(smsUri);
} else {
  print('Não foi possível enviar o SMS');
}
} catch (e) {
  print('Erro ao obter localização: $e');
}
}

```

Construção da página

```

backgroundColor: const Color.fromARGB(255, 15, 150, 82),
elevation: 0,
title: const Text('Emergência',
  style:
    TextStyle(color: Colors.white, fontWeight: FontWeight.bold)), // Text
), // AppBar
body: SafeArea(
  child: Padding(
    padding: const EdgeInsets.all(20.0),
    child: Column(
      crossAxisAlignment: CrossAxisAlignment.stretch,
      children: [
        Card(
          elevation: 5,
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(15),
          ), // RoundedRectangleBorder
          child: Padding(
            padding: const EdgeInsets.symmetric(horizontal: 15),
            child: DropdownButtonHideUnderline(
              child: DropdownButton<String>(
                isExpanded: true,
                value: dropdownvalue,
                onChanged: (String? newValue) {
                  setState(() {
                    dropdownvalue = newValue!;
                  });
                },
                items: <String>['Localização atual']
                  .map<DropdownMenuItem<String>>((String value) {
                    return DropdownMenuItem<String>(
                      value: value,
                      child: Text(value),

```

Construção do botão chamar

```
), // Padding
), // Card
const SizedBox(height: 50),
Expanded(
  child: Center(
    child: AspectRatio(
      aspectRatio: 1,
      child: ElevatedButton(
        onPressed: pegarPosicao,
        style: ElevatedButton.styleFrom(
          backgroundColor: const Color(0xFF1E8449),
          shape: RoundedRectangleBorder(
            borderRadius: BorderRadius.circular(20),
          ), // RoundedRectangleBorder
          elevation: 10,
        ),
      ),
    ),
    child: Column(
      mainAxisAlignment: MainAxisAlignment.center,
      children: [
        Icon(Icons.emergency,
          size: 60, color: Colors.white), // Icon
        SizedBox(height: 10),
        Text(
          'CHAMAR',
          style: TextStyle(
            fontSize: 30,
            fontWeight: FontWeight.bold,
            color: Colors.white,
          ), // TextStyle
        ), // Text
      ],
    ),
  ),
),
```

Emergência

Localização atual



CHAMAR



Home



Perfil



Sobre nós