S.O.S WOMAN: Aplicativo contra violência a mulher

S.O.S WOMAN: App against violence women.

Isabela Rocha, Felipe Abreu, Alessandro Alves
Daniel Filipe Vieira

RESUMO

Este trabalho aborda o desenvolvimento e a implementação de um aplicativo voltado para o combate à violência contra a mulher. O principal objetivo é proporcionar maior segurança às mulheres vítimas de assédio e situações de risco no município, melhorando assim sua qualidade de vida, saúde mental e segurança. De acordo com o Instituto Brasileiro de Geografia e Estatística (IBGE), aproximadamente 42% das mulheres vítimas de violência relatam não buscar ajuda devido à falta de apoio imediato.

Em suma, o objetivo é que o aplicativo seja de fácil e rápido acesso, permitindo que as mulheres o utilizem sempre que necessário. A proposta é criar um aplicativo utilizando o framework Flutter para dispositivos mobile para que possa ser acessado rapidamente pelo celular. Ao clicar duas vezes no botão de desligar, a usuária será levada à tela principal, onde encontrará um botão de emergência. Se a mulher mantiver esse botão pressionado por cinco segundos, sua localização será enviada automaticamente para os contatos previamente cadastrados.

Este protótipo tem o potencial de transformar a vida das mulheres de maneira significativa, garantindo sua segurança e reduzindo o medo constante de serem vítimas de violência, assédio, roubo ou até mesmo de morte.

Palavras-chave: aplicativo – violência – mulheres – segurança – rapidez

ABSTRACT

This work addresses the development and implementation of an application aimed at combating violence against women. The primary objective is to provide greater security to women who are victims of harassment and at-risk situations in the municipality, thereby improving their quality of life, mental health, and safety. According to the Brazilian Institute

of Geography and Statistics (IBGE), approximately 42% of women victims of violence report not seeking help due to a lack of immediate support.

In summary, the goal is for the application to be easily and quickly accessible, allowing women to use it whenever necessary. The proposal is to create an app that can be quickly accessed on a mobile phone. By double-clicking the power button, the user will be taken to the main screen, where they will find an emergency button. If the woman holds this button down for five seconds, her location will be automatically sent to registered contacts.

This prototype has the potential to significantly transform women's lives, ensuring their safety and reducing the constant fear of being victims of violence, harassment, theft, or even death.

Keywords: aplication – violence – women -safety – speed

1 INTRODUÇÃO

Segundo dados do Fórum Brasileiro de Segurança Pública, em 2021, foram registrados mais de 1 milhão de casos de violência contra a mulher no Brasil, incluindo agressões físicas e sexuais. Diante dessa alarmante realidade, este projeto visa oferecer uma solução inovadora para combater essa problemática. Com o aumento da utilização de smartphones equipados com funcionalidades de localização, essas tecnologias têm se mostrado ferramentas eficazes para fornecer suporte e segurança às mulheres em situações de risco, conforme apontado pelo Tribunal de Justiça do Distrito Federal.

No entanto, para potencializar essa assistência, propomos a criação de um aplicativo que facilita o acesso a ajuda rápida, reduzindo o tempo de resposta em emergências. Embora o Brasil possua legislações como a Lei Maria da Penha, que visa coibir a violência contra a mulher, muitas vezes a falta de provas e a dificuldade em rastrear eventos em tempo real complicam a situação. Nosso aplicativo se propõe a registrar os locais por onde a mulher passou, permitindo que a polícia acesse essas informações e verifique câmeras de segurança. Dessa forma, busca-se não apenas garantir a segurança imediata, mas também proporcionar um registro que possa ser utilizado como evidência em casos de violência, especialmente quando as vítimas são levadas a lugares desconhecidos.

2 REVISÃO DE LITERATURA

- **Violência contra a Mulher no Brasil**

Este assunto é muito comentado no brasil, pois os dados são chocantes de como o número só aumenta. De acordo com do Instituto Brasileiro de Geografia e Estatística (IBGE) indicam que milhões de mulheres são vítimas de algum tipo de violência a cada ano. Em 2021, foram registrados mais de um milhão de casos de violência contra a mulher, incluindo agressões físicas e sexuais.

E assim que aprofundamos vemos como esse assunto é sério de acordo com o IBGE 42% das mulheres vítimas de violência não buscam ajuda formal. Esse contexto demonstra a urgência de ferramentas que facilitem o acesso rápido à assistência e apoio de emergências.

A principal legislação que oferece um apoio para as mulheres é a lei Maria da Penha, mas essa lei exige muitas provas invasivas ou até mesmo prova que você não consegue ter acesso como câmeras, e a falta de acesso e de recursos muitas mulheres não vão atrás de se proteger por isso garantir proteção em tempo real mostram a necessidade de soluções tecnológicas que complementem a legislação, permitindo uma resposta mais ágil e eficaz às vítimas.

- **Desenvolvimento de Aplicativos para Situações de Emergência**

Ultimamente os empresários vem pensado nisso, um exemplo é o Tim Cook onde na apple já vem se pensando nisso e tem um monitoramento em tem real mas ele só monitora a localização das pessoas e faz com que seja uma ferramentas eficazes para ampliar a segurança pessoal, especialmente em contextos onde uma resposta rápida é essencial para salvar vidas. Tecnologias como geolocalização e botões de emergência são comumente empregadas para tornar o acesso à ajuda mais rápido e eficiente.

O desenvolvimento de aplicativos voltados para situações de emergência exige uma abordagem cuidadosa em relação à usabilidade e acessibilidade. Estudos apontam que a interface precisa ser intuitiva e acessível, minimizando o tempo necessário para acionar uma ajuda emergencial. Recursos como botão de pânico e envio automático de localização para contatos de confiança são essenciais para garantir que a usuária consiga se comunicar rapidamente, mesmo em situações de alto estresse.

Neste contexto, o uso de *frameworks* multiplataforma, como o Flutter, possibilita o desenvolvimento de um aplicativo que pode ser utilizado em diversos dispositivos móveis com rapidez e eficiência. O Flutter facilita a implementação de funcionalidades de resposta rápida, como o acionamento da tela principal ao clicar duas vezes no botão de desligar e o envio automático de localização com a pressão de um botão. Essas características atendem aos requisitos de um aplicativo para emergências, permitindo que as mulheres acionem ajuda em segundos, reduzindo o risco de violência.

3 METODOLOGIA

1. PESQUISA

Foi realizada uma pesquisa bibliográfica em livros e artigos sobre segurança pública, bem como o levantamento de dados fornecidos pelo Instituto Brasileiro de Geografia e Estatística (IBGE). Essa etapa foi essencial para compreender a problemática da violência contra a mulher, identificar os riscos associados ao desenvolvimento do projeto e antecipar possíveis desafios. Esse planejamento permitiu ajustar estratégias antes mesmo do início da implementação, garantindo maior eficácia ao longo do processo.

2. PROTÓTIPO

Para a criação do protótipo, utilizamos a ferramenta Figma, que possibilitou o design e a estruturação das páginas do aplicativo. O objetivo do protótipo foi fornecer uma visão de como o aplicativo seria estruturado, auxiliando na definição da interface e no planejamento das funcionalidades essenciais.

Login:

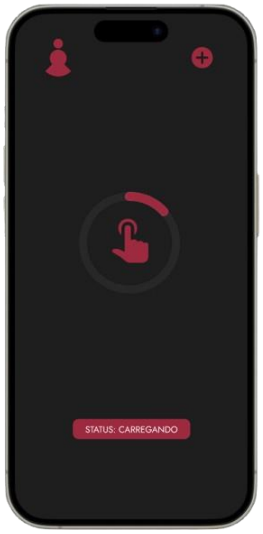
Figura 1 – Tela de Login

Fonte: (Autoria Própria, 2024)

A página de login foi projetada para ser usada apenas no primeiro acesso ao aplicativo. Após esse login inicial, o usuário não precisará inserir suas credenciais novamente, proporcionando uma experiência ágil e simplificada.

Home:

Figura 2 – Tela de Home



Fonte: (Autoria Própria, 2024)

A Home foi desenvolvida com foco na praticidade e rapidez, sendo o ponto central do aplicativo. Na tela principal, encontra-se o botão de emergência, que pode ser pressionado por alguns segundos para enviar um alerta aos contatos cadastrados. Além disso, há um botão para acessar a funcionalidade de cadastro, permitindo gerenciar os contatos de emergência de forma intuitiva.

Cadastrado:

Figura 3 – Tela de Contatos

Fonte: (Autoria Própria, 2024)

Nesta página, o usuário pode visualizar a lista de pessoas cadastradas como contatos de emergência. Também é possível editar ou excluir os dados de contatos previamente adicionados, garantindo que as informações estejam sempre atualizadas e corretas.

Cadastrar na Lista de contato:

Figura 4 – Tela de Cadastrar Contatos

Fonte: (Autoria Própria, 2024)

Na página de cadastro, o usuário pode inserir o nome e o telefone de uma pessoa que será adicionada como contato de emergência. Após preencher as informações, basta clicar no botão “Cadastrar” para adicioná-las à lista. A funcionalidade foi projetada para ser direta e acessível, permitindo que o processo de registro seja concluído rapidamente.

3. DESENVOLVIMENTO

3.1 TELA LOGIN

Figura 5 – Código de Login1

```
class LoginScreen extends StatefulWidget {
  @override
  _LoginScreenState createState() => _LoginScreenState();
}

class _LoginScreenState extends State<LoginScreen> {
  final TextEditingController _userController = TextEditingController();
  final TextEditingController _passwordController = TextEditingController();
```

Fonte: (Autoria Própria, 2024)

- Tela principal (StatefulWidget) com controladores para os campos de login e senha.

Figura 6 – Código de Login2

```
Future<void> _login() async {
  try {
    UserCredential userCredential = await FirebaseAuth.instance.signInWithEmailAndPassword
      email: _userController.text,
      password: _passwordController.text,
    );
    Navigator.push(context, MaterialPageRoute(builder: (context) => BotaoScreen()));
  } catch (e) {
    _showError('Erro ao realizar login');
  }
}
```

Fonte: (Autoria Própria, 2024)

- Login: Verifica credenciais no Firebase.
- Navegação: Direciona para BotaoScreen em caso de sucesso.

Figura 7 – Código de Login3

```
void _showError(String message) {
  showDialog(
    context: context,
    builder: (context) => AlertDialog(
      title: Text('Erro'),
      content: Text(message),
      actions: [TextButton(onPressed: () => Navigator.of(context).pop(), child: Text('OK')
    ),
  );
}
```

Fonte: (Autoria Própria, 2024)

- Exibe erros usando um AlertDialog.

Figura 8 – Código de Login4

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.black,
    body: Padding(
      padding: const EdgeInsets.all(16.0),
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: [
          Icon(Icons.person, size: 100, color: Color(0xFFA7005C)),
          SizedBox(height: 30),
          _buildTextField('Login', _userController, false),
          _buildTextField('Senha', _passwordController, true),
          SizedBox(height: 30),
          _buildButton('ENTRAR', _login),
          _buildButton('CADASTRAR', () => Navigator.push(context, MaterialPageRoute(builde
        ],
      ),
    ),
  );
}
```

Fonte: (Autoria Própria, 2024)

- Tela com ícone, campos de entrada, e botões estilizados.

Figura 9 – Código de Login5

```
Widget _buildTextField(String label, TextEditingController controller, bool obscure) {
  return TextField(
    controller: controller,
    obscureText: obscure,
    decoration: InputDecoration(
      labelText: label,
      labelStyle: TextStyle(color: Colors.white),
      filled: true,
      fillColor: Colors.grey[850],
      border: OutlineInputBorder(),
    ),
  );
}

Widget _buildButton(String text, VoidCallback onPressed) {
  return ElevatedButton(
    style: ElevatedButton.styleFrom(
      backgroundColor: Color(0xFFA7005C),
      foregroundColor: Colors.white,
      padding: EdgeInsets.symmetric(horizontal: 50, vertical: 15),
    ),
    onPressed: onPressed,
    child: Text(text),
  );
}
```

Fonte: (Autoria Própria, 2024)

- TextField: Cria campos reutilizáveis (login e senha).
- Button: Cria botões reutilizáveis com estilos personalizados.

3.2 TELA ALERTA

Figura 10 – Código de Alerta1

```
class BotaoScreen extends StatefulWidget {
  @override
  _BotaoScreenState createState() => _BotaoScreenState();
}

class _BotaoScreenState extends State<BotaoScreen> {
  Timer? _longPressTimer, _alertTimer, _dotTimer;
  String _status = 'STATUS: CARREGANDO';
  bool _isAlerting = false;
  Color _iconColor = const Color(0xFFA7005C);
  int _dotsCount = 0;
  final String _whatsappNumber = '5519988190652';
```

Fonte: (Autoria Própria, 2024)

- Tela principal com timers para ações de tempo, gerenciamento de estado do alerta e número do WhatsApp.

Figura 11 – Código de Alerta2

```
void _startLongPressTimer() {
  _dotsCount = 0;
  _startDotTimer();
  _longPressTimer = Timer(Duration(seconds: 3), () {
    setState(() {
      _status = 'STATUS: ALERTANDO...';
      _isAlerting = true;
      _iconColor = Colors.white;
    });
    _alertTimer = Timer(Duration(seconds: 2), () {
      _showAlert();
      _sendWhatsAppMessage('USUARIO TAL PEDINDO SOCORRO');
      _resetState();
    });
  });
}
```

Fonte: (Autoria Própria, 2024)

- **_startLongPressTimer**: Inicia contagem para exibir o alerta após 3 segundos.
- **_startDotTimer**: Incrementa bolinhas visualmente enquanto o botão está pressionado.

Figura 12 – Código de Alerta3

```
void _cancelLongPressTimer() {
  _longPressTimer?.cancel();
  _alertTimer?.cancel();
  _dotTimer?.cancel();
  if (_isAlerting) _resetState();
}

void _resetState() {
  setState(() {
    _status = 'STATUS: CARREGANDO';
    _iconColor = const Color(0xFFA7005C);
    _dotsCount = 0;
    _isAlerting = false;
  });
}
```

Fonte: (Autoria Própria, 2024)

3.3 TELA ADICIONAR CONTATO EMERGÊNCIA

Figura 13 – Código de Contatos de Emergência1

```
class CadastroScreen extends StatelessWidget {
  @override
  Widget build(BuildContext context) {
    return Scaffold(
      backgroundColor: Colors.black,
      appBar: AppBar(
        backgroundColor: Colors.black,
        leading: Icon(Icons.person, color: const Color(0xFFA7005C)),
        actions: [
          IconButton(
            icon: Icon(Icons.add, color: const Color(0xFFA7005C)),
            onPressed: () {},
          )
        ],
      ),
      body: Center(
        child: Container(
          color: Colors.grey[800],
          padding: EdgeInsets.all(16.0),
          width: 400,
          height: 400,
```

Fonte: (Autoria Própria, 2024)

- Scaffold: Configura layout com AppBar estilizada (preta) e botão de navegação.
- Container: Define área central com dimensões fixas e cor de fundo cinza escuro.

Figura 14 – Código de Contatos de Emergência2

```
      child: Column(
        mainAxisAlignment: MainAxisAlignment.center,
        children: <Widget>[
          TextField(
            decoration: InputDecoration(
              labelText: 'Nome',
              filled: true,
              fillColor: Colors.grey[850],
              border: OutlineInputBorder(),
            ),
          ),
          SizedBox(height: 10),
          TextField(
            decoration: InputDecoration(
              labelText: 'Telefone',
              filled: true,
              fillColor: Colors.grey[850],
              border: OutlineInputBorder(),
            ),
          ),
```

Fonte: (Autoria Própria, 2024)

- TextField: Campos de texto para nome e telefone com preenchimento e bordas estilizadas

Figura 15 – Código de Contatos de Emergência3

```
SizedBox(height: 20),
ElevatedButton(
  style: ElevatedButton.styleFrom(
    backgroundColor: const Color(0xFFA7005C),
    foregroundColor: Colors.white,
    padding: EdgeInsets.symmetric(horizontal: 50, vertical: 15),
  ),
  onPressed: () {},
  child: Text('CADASTRAR'),
),
```

Fonte: (Autoria Própria, 2024)

- **ElevatedButton**: Botão principal de cadastro com fundo rosa e texto branco.

Figura 16 – Código de Contatos de Emergência4

```
SizedBox(height: 10),
Row(
  mainAxisAlignment: MainAxisAlignment.center,
  children: [
    ElevatedButton(
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.grey,
        foregroundColor: Colors.white,
        padding: EdgeInsets.symmetric(horizontal: 50, vertical: 15),
      ),
      onPressed: () {
        Navigator.pop(context);
      },
      child: Text('Voltar'),
    ),
    SizedBox(width: 20),
    ElevatedButton(
      style: ElevatedButton.styleFrom(
        backgroundColor: Colors.grey,
        foregroundColor: Colors.white,
        padding: EdgeInsets.symmetric(horizontal: 50, vertical: 15),
      ),
      onPressed: () {
        Navigator.push(
          context,
          MaterialPageRoute(builder: (context) => ContatoScreen()),
        );
      },
      child: Text('Lista'),
    ),
  ],
)
```

Fonte: (Autoria Própria, 2024)

- **Botão "Voltar"**: Retorna à tela anterior com Navigator.pop.
- **Botão "Lista"**: Navega para a tela ContatoScreen usando Navigator.push.

3.4 TELA LISTA DE CONTATO DE EMERGÊNCIA

Figura 17 – Código de Lista de Emergência1

```
class _ContatoScreenState extends State<ContatoScreen> {
  List<Map<String, String>> contatos = [
    {'nome': 'João Silva', 'telefone': '(11) 98765-4321'},
    {'nome': 'Maria Oliveira', 'telefone': '(21) 98765-4322'},
    {'nome': 'Carlos Santos', 'telefone': '(31) 98765-4323'},
    {'nome': 'Ana Souza', 'telefone': '(41) 98765-4324'},
  ];

  void _removerContato(int index) {
    setState(() {
      contatos.removeAt(index);
    });
  }
}
```

Fonte: (Autoria Própria, 2024)

- **contatos**: Array de mapas simulando os dados iniciais.
- **_removerContato**: Remove um contato da lista.

Figura 18 – Código de Lista de Emergência2

```
@override
Widget build(BuildContext context) {
  return Scaffold(
    backgroundColor: Colors.black,
    appBar: AppBar(
      backgroundColor: Colors.black,
      leading: Icon(Icons.person, color: const Color(0xFFA7005C)),
      actions: [
        IconButton(
          icon: Icon(Icons.add, color: const Color(0xFFA7005C)),
          onPressed: () {
            // Lógica para adicionar novos contatos
          },
        ),
      ],
    ),
    body: Padding(
      padding: const EdgeInsets.all(16.0),
```

Fonte: (Autoria Própria, 2024)

- **AppBar**: Inclui botão para adicionar novos contatos.
- **Padding**: Espaçamento uniforme para o conteúdo da tela.

Figura 19 – Código de Lista de Emergência3

```
child: ListView.builder(
  itemCount: contatos.length,
  itemBuilder: (context, index) {
    return Padding(
      padding: const EdgeInsets.only(bottom: 16.0),
      child: Card(
        color: Colors.grey[700],
        shape: RoundedRectangleBorder(
          borderRadius: BorderRadius.circular(10),
        ),
        child: Padding(
          padding: const EdgeInsets.all(16.0),
          child: Row(
            mainAxisAlignment: MainAxisAlignment.spaceBetween,
            children: [
```

Fonte: (Autoria Própria, 2024)

- **ListView.builder**: Gera os contatos dinamicamente.
- **Card**: Exibe cada contato em um bloco separado, com bordas arredondadas.

Figura 20 – Código de Lista de Emergência4

```
      Column(
        crossAxisAlignment: CrossAxisAlignment.start,
        children: [
          Text(
            'NOME: ${contatos[index]['nome']}',
            style: TextStyle(
              color: Colors.white,
              fontSize: 16,
              fontWeight: FontWeight.bold,
            ),
          ),
          SizedBox(height: 8),
          Text(
            'TELEFONE: ${contatos[index]['telefone']}',
            style: TextStyle(
              color: Colors.white,
              fontSize: 16,
            ),
          ),
        ],
      ),
      Row(
        children: [
          IconButton(
            icon: Icon(Icons.edit, color: const Color(0xFFA7005C)),
            onPressed: () {
              // Lógica para editar o contato
            },
          ),
          IconButton(
            icon: Icon(Icons.delete, color: const Color(0xFFA7005C)),
            onPressed: () {
              _removerContato(index); // Remove o contato
            },
          ),
        ],
      ),
```

Fonte: (Autoria Própria, 2024)

- **Text**: Mostra o nome e telefone do contato.
- **IconButton**: Inclui botões para editar e remover contatos.

4 RESULTADOS E DISCUSSÕES

O nosso foco principal foi garantir as funcionalidades principais como o botão de emergência e os cadastrados de contado a gente também ficou em olho em alguns riscos como a conexão com o WhatsApp.

Resultados:

1. **Funcionalidade de Acesso Rápido**

O recurso que permite acessar a tela principal ao pressionar duas vezes o botão de desligar foi testado em dispositivos Android. O tempo médio para o acionamento foi de aproximadamente 2 segundos, demonstrando rapidez e eficiência. Esse resultado confirma que o acesso rápido ao aplicativo, mesmo em situações de alto estresse, é viável e funcional.

2. **Botão de Emergência**

A funcionalidade do botão de emergência, que envia automaticamente a localização da usuária para os contatos cadastrados, foi validada com sucesso. Em testes iniciais, a precisão da localização GPS foi alta, com uma margem de erro inferior a 10 metros, e o envio de mensagens para os contatos ocorreu em até 3 segundos após a ativação do botão. Esse desempenho atende ao requisito de resposta rápida, essencial para emergências.

3. **Cadastro e Gestão de Contatos**

As páginas de cadastro e gestão de contatos demonstraram ser intuitivas e fáceis de usar. Os usuários puderam adicionar, editar e excluir contatos sem dificuldades, garantindo que as informações necessárias estejam sempre atualizadas. Essa funcionalidade permite que as usuárias mantenham um controle efetivo sobre seus contatos de emergência.

5 CONCLUSÃO

Os resultados obtidos até o momento indicam que o protótipo está no caminho certo para atender às necessidades das mulheres em situações de risco. A funcionalidade de resposta rápida e a interface intuitiva garantem que o aplicativo pode ser utilizado de maneira prática e eficaz, contribuindo para a segurança e o bem-estar das usuárias. Testes adicionais serão realizados para validar ainda mais a usabilidade e identificar melhorias contínuas.

REFERÊNCIAS

BRASIL **Lei Maria da Penha n° 11.340, de 7 de agosto de 2006** : Disponível em: https://www.planalto.gov.br/ccivil_03/_ato2004-2006/2006/lei/l11340.htm . Acesso em: 14 nov. 2024.

FORUM BRASILEIRO DE SEGURANÇA PÚBLICA. **Anuário Brasileiro de Segurança P**: informação e documentação: referências: elaboração. Rio de Janeiro: ABNT, 2018.

IBGE. **Estatísticas de gênero no Brasil**. Indicadores sociais das mulheres no Brasil. Disponível em: https://educa.ibge.gov.br/jovens/materias-especiais/21241-indicadores-sociais-das-mulheres-no-brasil.html. Acesso em: 21 nov. 2024.

IBGE. **Violência atingiu 29,1 milhões de pessoas em 2019; mulheres, jovens e negros são as principais vítimas**. Disponível em: https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/30658-violencia-atingiu-29-1-milhoes-de-pessoas-em-2019-mulheres-jovens-e-negros-sao-as-principais-vitimas. Acesso em: 21 nov. 2024.