

UNIVERSIDADE DO MINHO  
Departamento de Informática

TÓPICOS DE DESENVOLVIMENTO DE SOFTWARE

## Trabalho prático - Parte 2 BraGuia

**Elaborado por:**  
João Alvim Almeida - PG53902  
Simão Barroso - PG54236  
Simão Matos - PG54239

Ano Letivo 2023/24  
17 de junho de 2024

# Índice

<b>1</b>	<b>Introdução</b>	<b>2</b>
<b>2</b>	<b>Detalhes da implementação</b>	<b>2</b>
2.1	Estrutura do projeto . . . . .	2
2.1.1	actions . . . . .	2
2.1.2	backgroundServices . . . . .	2
2.1.3	components . . . . .	2
2.1.4	screens . . . . .	2
2.1.5	state . . . . .	2
2.1.6	theme . . . . .	2
2.2	Soluções de implementação . . . . .	3
2.3	Bibliotecas/dependências utilizadas . . . . .	4
2.4	Padrões de software utilizados . . . . .	5
2.4.1	Provider . . . . .	5
2.4.2	Iterator . . . . .	5
2.4.3	Hooks . . . . .	5
<b>3</b>	<b>Mapa de Navegação</b>	<b>6</b>
<b>4</b>	<b>Funcionalidades</b>	<b>7</b>
<b>5</b>	<b>Discussão de Resultados</b>	<b>14</b>
5.1	Trabalho Realizado . . . . .	14
5.2	Limitações . . . . .	14
5.3	Funcionalidades Extra . . . . .	14
<b>6</b>	<b>Gestão de Projetos</b>	<b>14</b>
6.1	Gestão e Distribuição do Trabalho . . . . .	14
6.2	Eventuais metodologias de controlo de versões usadas . . . . .	14
6.3	Reflexão sobre performance individual . . . . .	14
<b>7</b>	<b>Conclusões</b>	<b>14</b>

# 1 Introdução

No âmbito da unidade curricular ”Tópicos de Desenvolvimento de Software”, propõe-se o desenvolvimento de um guia turístico móvel híbrido denominado ”BraGuia”. Nesta segunda fase do projeto, o objetivo é complementar a aplicação nativa Android desenvolvida na primeira fase, utilizando tecnologia cross-platform, nomeadamente React Native, para estender a compatibilidade a outros sistemas operativos móveis.

A aplicação ”BraGuia” visa substituir um guia turístico tradicional, oferecendo aos utilizadores roteiros turísticos, funcionalidades de localização e navegação geográfica, e reprodução de mídia relacionada com pontos de interesse. A informação é obtida a partir de um backend desenvolvido especificamente para este projeto, garantindo que os utilizadores recebem conteúdos atualizados e relevantes.

A transição para uma aplicação multi-plataforma é um processo comum e necessário para ampliar o alcance e a usabilidade das aplicações móveis. Utilizando a framework React Native, que permite a integração com código nativo, podemos realizar esta transição de forma progressiva e iterativa, maximizando a reutilização de código e a eficiência do desenvolvimento.

## 2 Detalhes da implementação

### 2.1 Estrutura do projeto

Ao desenvolver uma aplicação React Native, é crucial organizar a base de código de forma eficiente para garantir sua manutenção e escalabilidade. Além do arquivo App.jsx, a nossa implementação inclui o código nativo para iOS e Android, a fim de adicionar detalhes específicos para cada plataforma. Também temos uma pasta chamada ”assets” para armazenar os arquivos de multimédia do projeto (como por exemplo o logotipo da aplicação). No entanto, o aspeto principal em que a estrutura pode variar entre projetos React Native é a pasta ”source”(src), na qual separamos os diferentes aspectos da aplicação em diretorias lógicas. Vamos analisar mais de perto cada diretoria e o seu propósito

#### 2.1.1 actions

A Pasta ”actions” contém ficheiros que definem várias ações ou eventos que podem ocorrer dentro da aplicação. Por exemplo, ”appData.js” e ”user.js” podem definir ações relacionadas à gestão dos dados da aplicação e ações específicas do usuário, respectivamente. Ao separar as ações em arquivos dedicados, torna-se mais fácil gerir e manter uma clara separação de responsabilidades. Esta é uma das pastas recomendadas para a utilização da biblioteca Redux.

#### 2.1.2 backgroundServices

A Pasta ”backgroundServices” tem código responsável por lidar com a localização do utilizador.

#### 2.1.3 components

A Pasta ”components” contém todos os componentes React que criamos e que podemos reutilizar por que cada uma responsável por renderizar uma parte independente da interface do usuário.

#### 2.1.4 screens

Na pasta ”screens” colocamos os componentes que representam as diferentes ecras ou páginas do seu aplicativo.

#### 2.1.5 state

Pasta ‘State’ contém tudo relacionado ao controlo de estado da aplicação

#### 2.1.6 theme

A pasta ”theme” inclui ficheiros relacionados com a estética da aplicação em light mode e em dark mode.

## 2.2 Soluções de implementação

- **Persistência de dados:** Tomamos a decisão de carregar toda a informação da API quando fazemos login pela primeira vez. Quando descarregamos toda a informação da API quando fazemos o login temos que ter em conta que isso afeta a economização dos espaço e diminui a eficiência por um instante.
- **Funcionamento offline:** A nossa aplicação foi desenvolvidos para funcionar mesmo sem conexão com a internet, garantindo uma experiência de usuário perfeita. Uma das funcionalidades essenciais é a possibilidade de autenticar o último usuário que acessou um aplicativo naquele dispositivo, independente do estado da conexão com a internet.
- **Navegação com Google Maps:** A aplicação prepara e abre uma rota no Google Maps usando coordenadas de localização. Ele verifica e formata os dados, define waypoints para pontos intermédios da rota, determina o destino final e cria um URL de navegação, que é então aberto no navegador do dispositivo para iniciar a navegação.
- **Pedidos à API:** Todos os pedidos feitos à API utilizam o Fetch dentro de uma função assíncrona de forma a fazer pedidos HTTP sem interromper o utilizador, possibilitando o utilizador de interagir com a aplicação enquanto o pedido executa.
- **Integração de mapa num roteiro:** Para disponibilizar um mapa interativo onde estão marcados os pontos de interesse de um roteiro, é utilizada a biblioteca 'react-native-maps' para disponibilizar um mapa, a partir da qual fornecemos as localizações de cada ponto e configuramos algumas opções da câmara da aplicação do Maps. Este procedimento foi idêntico ao utilizado na fase anterior com a API do Google Maps.
- **Descarregamento de media:** Primeiramente, verifica se o link do arquivo está presente. Em seguida, solicitamos permissão ao utilizador para gravar no armazenamento externo do dispositivo. Se a permissão for concedida, o arquivo é descarregado e salvo no diretório de downloads do dispositivo, com ajuda da biblioteca "RNFS". Caso a permissão seja negada ou bloqueada, o utilizador é informado e orientado a ajustar as configurações do dispositivo para permitir o descarregamento.

## 2.3 Bibliotecas/dependências utilizadas

Neste projeto, foram utilizadas várias bibliotecas para garantir um desenvolvimento eficiente e robusto da aplicação:

```
"dependencies": {  
    "@react-native-async-storage/async-storage": "^1.23.1",  
    "@react-native-community/geolocation": "3.2.1",  
    "@react-navigation/bottom-tabs": "6.5.20",  
    "@react-navigation/material-top-tabs": "6.6.13",  
    "@react-navigation/native": "6.1.17",  
    "@react-navigation/stack": "6.3.29",  
    "@reduxjs/toolkit": "2.2.5",  
    "@supersami/rn-foreground-service": "2.1.1",  
    "expo-device": "6.0.2",  
    "expo-location": "17.0.1",  
    "expo-notifications": "0.28.9",  
    "react": "18.2.0",  
    "react_redux": "1.2.1",  
    "react-native": "0.74.1",  
    "react-native-fs": "2.20.0",  
    "react-native-geolocation-service": "5.3.1",  
    "react-native-gesture-handler": "2.16.2",  
    "react-native-maps": "1.15.6",  
    "react-native-maps-directions": "1.9.0",  
    "react-native-pager-view": "6.3.1",  
    "react-native-permissions": "4.1.5",  
    "react-native-push-notification": "8.1.1",  
    "react-native-safe-area-context": "4.10.4",  
    "react-native-screens": "3.31.1",  
    "react-native-sound": "0.11.2",  
    "react-native-tab-view": "3.5.2",  
    "react-native-vector-icons": "10.1.0",  
    "react-native-video": "6.2.0",  
    "react-navigation": "5.0.0",  
    "react-redux": "9.1.2",  
    "redux": "5.0.1",  
    "redux-persist": "6.0.0",  
    "redux-thunk": "3.1.0"  
},
```

Figura 1: Permissões

Para garantir que a aplicação funcione corretamente e ofereça todas as funcionalidades aos utilizadores, é necessário conceder determinadas permissões. Estas permissões são solicitadas para garantir o acesso a recursos essenciais do dispositivo e à rede.

- **Permissão de Internet:** Esta permissão é necessária para que a aplicação possa se comunicar com a internet e acessar recursos online.
- **Permissão de Localização Precisa e Aproximada:** Estas permissões são solicitadas para permitir que a aplicação obtenha a localização do dispositivo, o que pode ser necessário para funcionalidades como serviços de mapas ou recomendações com base na localização.
- **Permissão de Realização de Chamadas Telefónicas:** Esta permissão é requerida para que a aplicação possa realizar chamadas telefónicas diretamente do dispositivo.
- **Permissão de Foreground Service:** Esta permissão é necessária para permitir que a aplicação execute determinadas tarefas em primeiro plano, mesmo quando não está em uso ativo.
- **Permissão de Acesso ao Estado da Rede:** Esta permissão permite que a aplicação verifique o estado da conexão de rede do dispositivo, o que pode ser necessário para garantir uma experiência de uso contínua e consistente.
- **Permissão para Alterar armazenamento interno:** A permissão para alterar o armazenamento interno permite que uma aplicação móvel possa guardar, modificar ou apagar ficheiros diretamente no armazenamento do dispositivo.

É de mencionar que as bibliotecas Expo não foram utilizadas.

## 2.4 Padrões de software utilizados

### 2.4.1 Provider

O Padrão Provider foi útil no desenvolvimento com o Redux, especialmente quando se deseja persistir o estado da aplicação entre recarregamentos ou sessões. o Redux é integrado com a aplicação React para que qualquer componente possa acessar o estado global gerenciado pelo Redux. Isso é feito utilizando o componente Provider fornecido pelo React-Redux.

```
const App = () => {
  return (
    <Provider store={store}>
      <PersistGate loading={<Text>Loading...</Text>} persistor={persistor}>
        <AppNavigator />
      </PersistGate>
    </Provider>
  );
};

export default App;
```

Figura 2: Padrão Provider.

### 2.4.2 Iterator

Utilizamos este padrão para percorrer os pins e os trails.

### 2.4.3 Hooks

Em vez de utilizar os métodos de ciclo de vida tradicionais, como o construtores, o grupo optou por utilizar hooks por causa da sua simplicidade, reutilização de lógica, facilidade na resolução de problemas de ciclo de vida, melhor performance e compatibilidade com o ecossistema do React.

### 3 Mapa de Navegação

O mapa de navegação da nossa aplicação está apresentado em baixo. O ponto de partida é a página de *login*. Partindo desta, demonstramos vários exemplos das possíveis respostas do sistema perante as várias ações possíveis dos utilizadores.

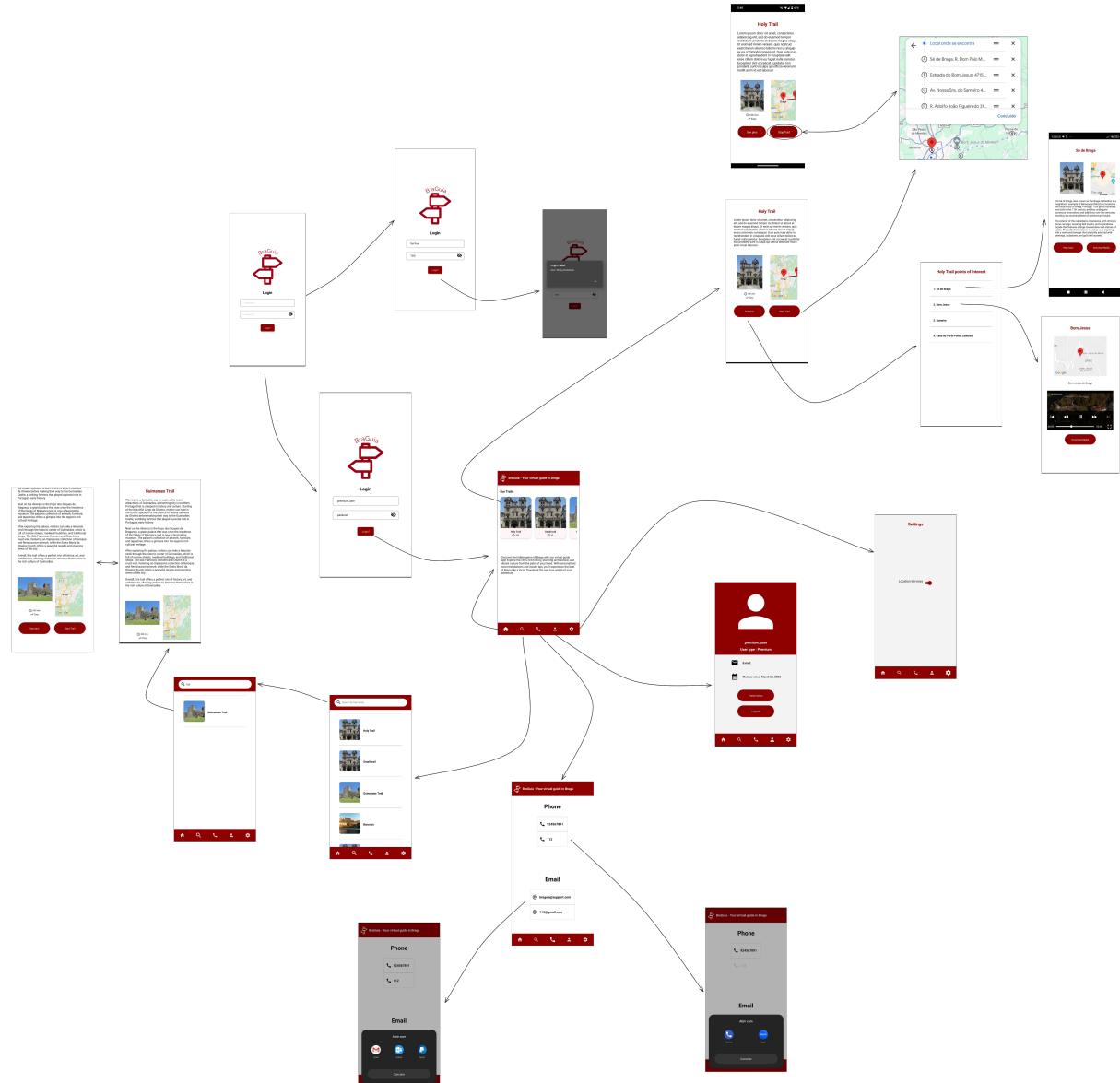


Figura 3: Mapa de Navegação

## 4 Funcionalidades

- A aplicação deve possuir uma página inicial onde apresenta as principais funcionalidades do guia turístico, descrição, etc.

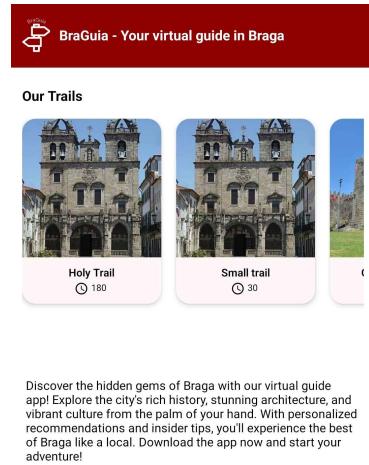


Figura 4: Página inicial

- A aplicação deve mostrar num ecrã, de forma responsiva, uma lista de roteiros disponíveis.

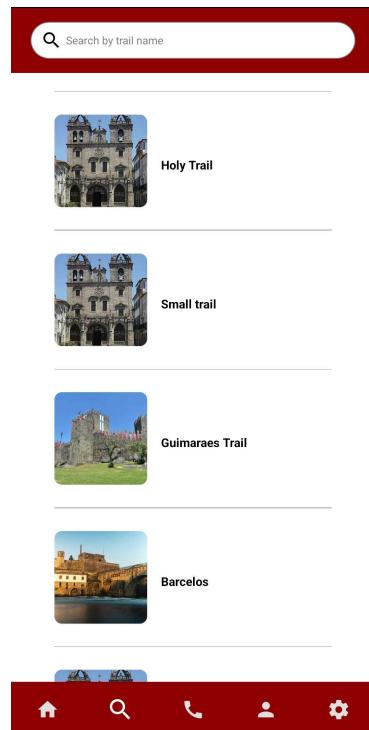


Figura 5: Pesquisa

- A aplicação deve permitir efetuar autenticação.



Figura 6: Login

- A aplicação deve suportar 2 tipos de utilizadores: utilizadores standard e utilizadores premium.

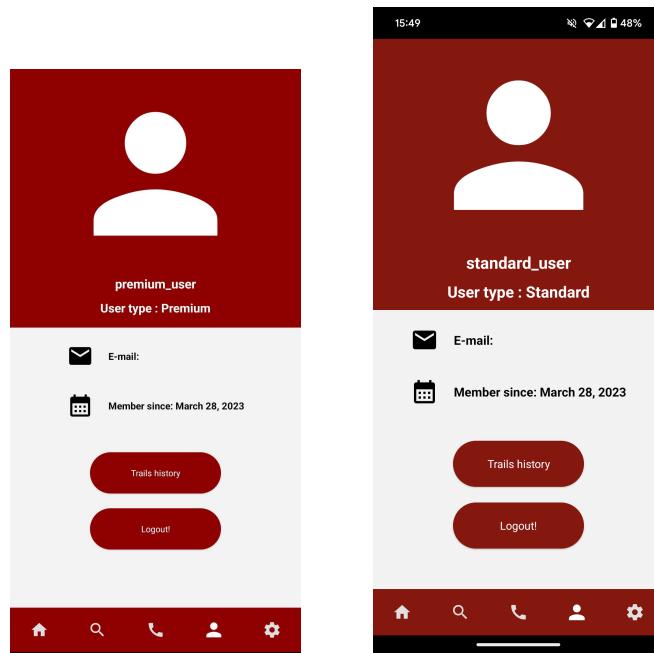


Figura 7: Tipos de utilizador

- A aplicação deve assumir que o utilizador tem o Google Maps instalado no seu dispositivo (e notificar o utilizador que este software é necessário).
- Para utilizadores premium (e apenas para estes) a aplicação deve possibilitar a capacidade de navegação, de consulta e descarregamento de mídia.

- A navegação proporcionada pelo Google Maps deve poder ser feita de forma visual e com auxílio de voz, de modo a que possa ser utilizada por condutores.

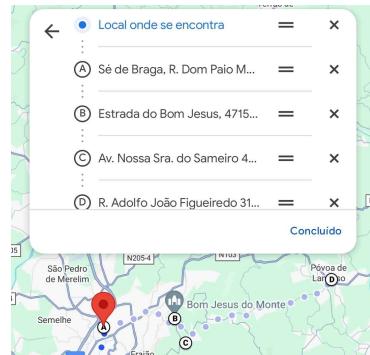


Figura 8: Navegação

- A aplicação deve possuir uma página de informações acerca do utilizador atualmente autenticado.

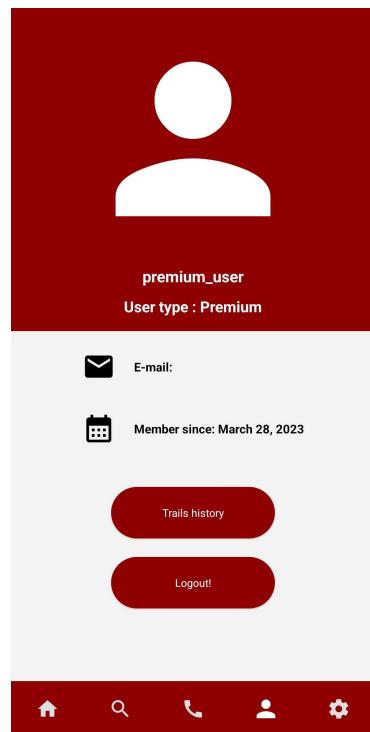


Figura 9: Perfil

- A aplicação deve mostrar, numa única página, informação acerca de um determinado roteiro: galeria de imagens, descrição, mapa do itinerário com pontos de interesse e informações sobre a mídia disponível para os seus pontos.

### Holy Trail

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum



Figura 10: Descrição do roteiro

- A aplicação deve possuir a capacidade de iniciar um roteiro.

### Holy Trail

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum

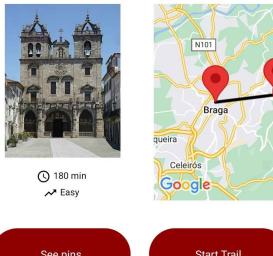


Figura 11: Roteiro

- A aplicação deve possuir a capacidade de emitir uma notificação quando o utilizador passa perto de um ponto de interesse.
- A notificação emitida quando o utilizador passa pelo ponto de interesse deve conter um atalho para o ecrã principal do ponto de interesse.

- A aplicação deve possuir a capacidade de interromper um roteiro.



### Holy Trail

Lorem ipsum dolor sit amet, consectetur adipiscing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam, quis nostrud exercitation ullamco laboris nisi ut aliquip ex ea commodo consequat. Duis aute irure dolor in reprehenderit in voluptate velit esse cillum dolore eu fugiat nulla pariatur. Excepteur sint occaecat cupidatat non proident, sunt in culpa qui officia deserunt mollit anim id est laborum



Figura 12: Stop

- A aplicação deve guardar (localmente) o histórico de roteiros e pontos de interesse visitados pelo utilizado.
- A aplicação deve possuir uma página que mostre toda a informação disponível relativa a um ponto de interesse: localização, galeria, mídia, descrição, propriedades, etc.



### Sé de Braga



The Sé de Braga, also known as the Braga Cathedral, is a magnificent example of Baroque architecture located in the historic city of Braga, Portugal. This grand cathedral was built in the 11th century and has undergone numerous renovations and additions over the centuries, resulting in a stunning blend of architectural styles.

The exterior of the cathedral is impressive, with intricate stone carvings, towering bell towers, and a grandiose facade that features a large rose window and statues of saints. The cathedral's interior is just as awe-inspiring, with a nave and transept that are richly adorned with paintings, sculptures, and gold leaf accents.

Play Audio

Download Media



Figura 13: Pin

- A aplicação deve ter a capacidade de apresentar e produzir 3 tipos de mídia: voz, imagem e vídeo.

---

### Bom Jesus



Bom Jesus de Braga



Download Media

---

Figura 14: Pin

- A aplicação deve possuir um menu com definições que o utilizador pode manipular.

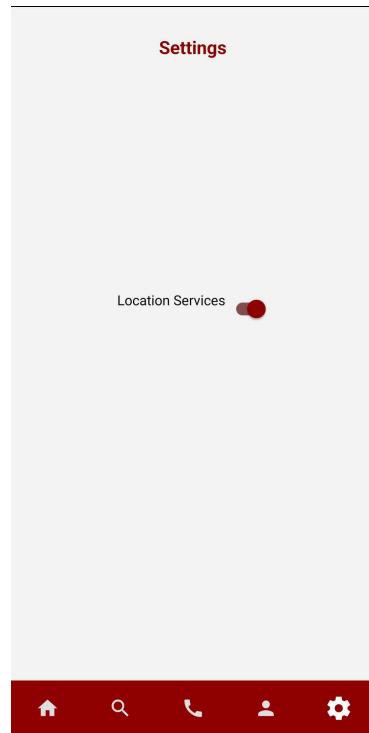


Figura 15: Definições

- A aplicação deve possuir a capacidade de ligar, desligar e configurar os serviços de localização.
- A aplicação deve possuir a capacidade de descarregar mídia do backend e aloja-la localmente, de modo a poder ser usada em contextos de conectividade reduzida.
- A aplicação deve possuir a capacidade de efetuar chamadas para contactos de emergência da aplicação através de um elemento gráfico facilmente acessível na aplicação.

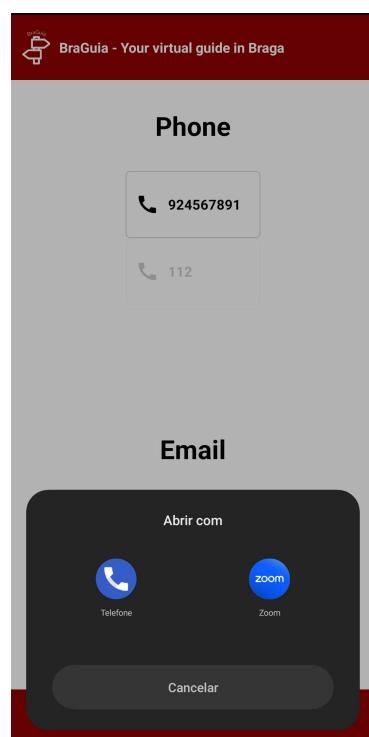


Figura 16: Emergência

## 5 Discussão de Resultados

### 5.1 Trabalho Realizado

Tendo em conta o trabalho realizado, consideramos que fizemos um bom trabalho. Fizemos quase todos os requisitos, como foi visto na imagens em cima.

Nas próximas secções vamos falar em concreto dos aspetos positivos e negativos.

### 5.2 Limitações

Relativamente a limitações, como todos os projetos, não foi possível explorar tudo ao promenor. A nossa aplicação de guia turístico é bastante funcional, mas apresenta algumas limitações importantes. Atualmente, não possui a capacidade de guardar o histórico de roteiros e pontos de interesse visitados, o que impede os utilizadores de rever suas jornadas passadas. Além disso, apesar de utilizar a localização em foreground, a aplicação não emite notificações quando o utilizador passa perto de um ponto de interesse, exigindo que o utilizador mantenha a aplicação aberta para obter informações em tempo real. Por fim, a aplicação não oferece a opção de dark mode, o que poderia melhorar o conforto visual e a eficiência da bateria em dispositivos com ecrãs OLED.

### 5.3 Funcionalidades Extra

Como funcionalidade extra temos a utilização do `Google Maps` embutido na aplicação. Consideramos que isto ajuda para a experiência do utilizador. No entanto é de realçar que é preciso uma `API Key`, que por motivos de não nos cancelarem a conta, não pusemos no código entregue no `github`.

## 6 Gestão de Projetos

### 6.1 Gestão e Distribuição do Trabalho

Devido à natureza mais igualitária deste trabalho, houve mais possibilidades do trabalho ser distribuído de maneira paralela, devido aos ecrãs diferentes poderem ser desenvolvidos ao mesmo tempo. Por causa disto, dividimos em 3 partes o trabalho, cada uma correspondente a um certo número de *screens*, tendo em conta o trabalho que estes causariam. Por exemplo, uma vez que durante o processo de login, este trata de ir buscar os dados ao backend, precisava de um tratamento mais cuidado.

### 6.2 Eventuais metodologias de controlo de versões usadas

Utilizamos principalmente o `github` como modo de guardar versões. Utilizamos também o `discord` para enviar os vários ficheiros de código trocados ao longo durante as reuniões semanais.

### 6.3 Reflexão sobre performance individual

Consideramos que os 3 membros do grupo tiveram a mesmo peso no desenvolvimento do trabalho.

Os 3 contribuímos na melhor das nossas capacidades para levar o projeto a um *bom porto*.

Fizemos reuniões semanais onde todos mostraram-se interessados em contribuir e em saber o que os outros fizeram.

No entanto, é normal, houveram alturas em houve membros que contribuíram mais do que outros, no entanto no final tudo se nivelou.

## 7 Conclusões

A segunda fase deste demonstrou a eficácia da utilização de React Native na transição de uma aplicação nativa Android para uma aplicação multi-plataforma. Com esta abordagem, conseguimos aumentar a compatibilidade da aplicação, permitindo que funcione em múltiplos sistemas operativos móveis sem comprometer a funcionalidade e a experiência do utilizador.

O uso de React Native facilitou a reutilização de código e agilizou o processo de desenvolvimento, evidenciando as vantagens do desenvolvimento cross-platform.

Este projeto não só atingiu os objetivos estabelecidos para a segunda fase, mas também proporcionou uma valiosa experiência prática na integração de tecnologias nativas e multi-plataforma.