FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

Bilhética Móvel no contexto DRT

Jorge David Monteiro Pacheco



Mestrado em Engenharia Informática e Computação

Supervisor: Prof. António Miguel Pimenta Monteiro

30 de julho de 2022

Bilhética Móvel no contexto DRT

Jorge David Monteiro Pacheco

Mestrado em Engenharia Informática e Computação

Aprovado em provas públicas pelo Júri:

Presidente: Prof. Carla Teixeira Lopes
Arguente: Prof. Paulo Baltarejo Sousa
Vogal: Prof. António Miguel Pimenta Monteiro

30 de julho de 2022

Resumo

O conceito de Demand-Responsive Transport (DRT), em português Transporte de Passageiros Flexível (TPF) é por vezes utilizado para definir um serviço não regular (não calendarizado), mas que se realiza a pedido do utilizador. Este conceito é especialmente útil em áreas suburbanas, rurais, ou outras cuja densidade populacional é reduzida e onde a procura é dispersa [**?**], o que afasta o investimento dos operadores tradicionais de serviço regulado de operar nessas áreas. A disponibilização de serviços públicos de transporte nestes meios deve ser encarada como fator de integração social e igualdade de oportunidades. É neste contexto que surge a necessidade de implementação de uma aplicação móvel que permita a aquisição de títulos de transporte, integrando-a com a plataforma HeyBus. Esta plataforma, já existente, foi desenvolvida pela OPT-Otimização e Planeamento de Transportes (proponente desta dissertação) e está desenhada especificamente para esta tipologia de transporte. Permite a monitorização por parte das entidades que a tutelam como a sua operacionalização incluindo a interação direta com os passageiros e a utilização de centros de atendimento para o registo de pedidos de marcação. Esta aplicação móvel estará disponível nas suas versões Android e iOS, sendo esperado que a comunicação entre a plataforma HeyBus, a aplicação, e os métodos de pagamento mais comuns seja fluída e permita a escolha e compra de títulos de transporte de forma segura e acessível.

Keywords: Demand-Responsive Transport, Mobile Ticketing, Smartphone App, Transport

Abstract

The concept of Demand-Responsive Transport (DRT) is often used to define a non-regular (not calendarized) service, one that is viable through user requests. This concept is beneficial and meaningful in suburban and rural areas, with a low population density and the demand is sparse [**?**], pushing away traditional transportation operators' investment. The availability of public means of transportations in these areas must be faced as a social integration and equality of opportunities factor. In this context, a need for a mobile app that allows the acquisition of transportation tickets integrated with the HeyBus platform emerges. This platform already exists and was developed by OPT, explicitly designed for this type of transportation (DRT). It allows overseeing entities to monitor the operationalization, which includes their interaction with passengers and the schedulling by call centers. This mobile app will be available in Android and iOS versions, and will integrate the most used payment methods. It is expected that the communication between HeyBus, the app and the payment methods is fluid and allows the schedulling of the trip and the purchase of public transport tickets.

Keywords: Demand-Responsive Transport, Mobile Ticketing, Smartphone App, Transport

Agradecimentos

Gostaria de agradecer ao Prof. António Miguel Pimenta Monteiro por toda a ajuda nesta dissertação, quer pelo aprofundar no conhecimento de certas tecnologias como pela sua disponibilidade durante todo o processo.

Uma palavra também para a OPT, proponente desta dissertação, pelo seu esforço e dedicação no desenvolvimentos que foram necessários para ser usados na aplicação móvel.

Por último um grande agradecimento aos meus amigos e à minha família, que sempre me ajudaram e motivaram para que eu conseguisse dar o meu melhor.

Espero ter conseguido.

Jorge Pacheco

"Tudo é ousado,
a quem nada se atreve."

Fernando Pessoa

Conteúdo

1 Introdução **1**
- 1.1 Contexto . . . 1
- 1.2 Motivação . . . 2
- 1.3 Objetivos . . . 3
  - 1.3.1 Integração com solução atual . . . 3
  - 1.3.2 Facilidade de uso . . . 3
  - 1.3.3 Minimização de custos . . . 3
  - 1.3.4 Multi-dispositivo . . . 3
  - 1.3.5 Privacidade e Segurança . . . 3
- 1.4 Estrutura do Documento . . . 4

2 Revisão da Literatura **5**
- 2.1 Estado da Arte . . . 5
  - 2.1.1 Transporte de Passageiros Flexível . . . 5
  - 2.1.2 Bilhética Móvel . . . 7
  - 2.1.3 Validação de bilhetes . . . 8
  - 2.1.4 Tecnologias . . . 9
- 2.2 Resumo . . . 11

3 Descrição do Problema e Proposta de Solução **13**
- 3.1 Modelos . . . 14
  - 3.1.1 Bilhete . . . 14
  - 3.1.2 Viagem . . . 15
  - 3.1.3 Validação de bilhetes . . . 15

4 Arquitetura & Modelo **17**
- 4.1 Visão Geral . . . 17
  - 4.1.1 Login . . . 17
  - 4.1.2 Compra de bilhetes . . . 18
  - 4.1.3 Validação de bilhetes . . . 18
  - 4.1.4 Esquema de arquitetura . . . 19
- 4.2 Funcionalidades da Aplicação móvel . . . 20
  - 4.2.1 Login . . . 20
  - 4.2.2 Fazer pedido . . . 20
  - 4.2.3 Listar histórico de pedidos . . . 20
  - 4.2.4 Mostrar detalhes de um pedido . . . 21
  - 4.2.5 Fazer o pagamento de um pedido . . . 22
  - 4.2.6 Gerar código-QR . . . 23

4.3 Casos de uso . . . 24
4.3.1 Compra de bilhete . . . 24
4.3.2 Visualização de um pedido . . . 25
4.3.3 Cancelamento de um pedido . . . 25
4.3.4 Pagamento de um pedido . . . 26
4.3.5 Geração de um código-QR . . . 27

5 Realização do Projeto & Implementação **29**
5.1 Visão Geral . . . 29
5.2 Detalhes de implementação . . . 29
5.2.1 Chamadas à API . . . 29
5.2.2 Navegação . . . 30
5.2.3 Dados . . . 30

6 Resultados **31**
6.1 VIsão Geral . . . 31
6.2 Operação & Execução . . . 31
6.3 Interfaces . . . 32
6.3.1 Login . . . 32
6.3.2 Menu Inicial . . . 33
6.3.3 Encontrar viagem . . . 34
6.3.4 Listagem de pedidos . . . 37
6.3.5 Bilhetes . . . 38
6.4 Avaliação . . . 45
6.4.1 Procedimento . . . 45
6.4.2 Questionário . . . 49
6.4.3 Resumo . . . 50

7 Conclusões e Trabalho Futuro **51**
7.1 Trabalho Futuro . . . 52
7.1.1 Login . . . 52
7.1.2 Pagamento . . . 52
7.1.3 Validação dos Bilhetes . . . 52
7.1.4 Lançamento . . . 53

Referências **55**

Lista de Figuras

2.1 Circuito Azul Reguengos de Monsaraz 6
2.2 Aplicação operador HeyBus 7
2.3 Aplicação Anda 8
2.4 Código QR versão 3 10

3.1 Modelo Bilhete 14
3.2 Modelo Viagem 15

4.1 Arquitetura do sistema 19
4.2 Criação de um novo pedido 24
4.3 Visualização de detalhes de um pedido pedido 25
4.4 Cancelamento de um pedido 26
4.5 Pagamento de um pedido 26
4.6 Geração de um código-QR num pedido 'Pago' 27

5.1 API call 29
5.2 Condição de *loading* 30

6.1 Interface da página de Login 32
6.2 Menu Inicial 33
6.3 Interface 'Nova Viagem' 34
6.4 Criação de pedido com sucesso 35
6.5 Erro na criação do pedido 36
6.6 Histórico de Pedidos 37
6.7 Bilhete com estado 'Pedido' 38
6.8 Sucesso no cancelamento de um pedido 39
6.9 Histórico antes e depois do cancelamento do Pedido 40
6.10 Sucesso no cancelamento de um pedido 41
6.11 Bilhete com estado 'Aceite' 42
6.12 Bilhete com estado 'Pago' 43
6.13 QR-Code do bilhete 44
6.14 Facilidade dos utilizadores na criação de um pedido 46
6.15 Percentagem de utilizadores com sucesso no cancelamento de um pedido 47
6.16 Facilidade na procura de um pedido 47
6.17 Facilidade dos utilizadores na utilização da aplicação 49
6.18 Facilidade dos utilizadores na criação de um pedido 49
6.19 Facilidade dos utilizadores na criação de um pedido 50

Acrónimos

API	Application Programming Interface
APP	Application
DRT	Demand Responsive Transport
JS	JavaScript
JSX	JavaScript XML
IMT	Instituto da Mobilidade e dos Transportes
NFC	Near-Field Communication
OPT	Optimização e Planeamento de Transportes
TPF	Transporte de Passageiros Flexível
XML	Extensible Markup Language

Capítulo 1

Introdução

1.1 Contexto

Para melhor compreender a origem desta dissertação, importa compreender o conceito de DRT. DRT não é um conceito recente, tendo aparecido nos anos 90. Significa "Demand-Responsive Transport", ou em português TPF ("Transporte de Passageiros Flexível") e é um tipo de transporte flexível que reage à procura existente e que, por isso, quebra com as formas tradicionais de transportes públicos.

Segundo Jenny Brake, John Nelson e Steve Wright, no seu artigo "Demand Responsive Transport: Towards the emergence of a new market segment", idealmente o transporte público deve ser tão conveniente e prático quanto o transporte privado, o que sugere que todo o transporte público deve ser Demand-Responsive.

> "Ideally, public transport would be as convenient as private transport, suggesting that 'all public transport should be demand responsive."' [4, chap. 2. The demand responsiveness of public transport]

Segundo os mesmos autores, DRT insere-se numa forma de transporte público intermédia, entre um serviço regular que usa pequenos autocarros e um serviço com rotas variadas e altamente personalizado, como o oferecido pelos táxis.

> "An intermediate form of public transport, somewhere between a regular service route that uses small low floor buses and variably routed, highly personalised transport services offered by taxis" [4, chap. 2. The demand responsiveness of public transport]

A definição do IMT(Instituto da Mobilidade e dos Transportes)[11] corrobora o que foi dito anteriormente e vai mais além: "O transporte de passageiros flexível caracteriza-se pela sua adaptabilidade às necessidades dos utilizadores, permitindo a flexibilidade de, pelo menos, uma das dimensões da prestação do serviço:

- Itinerários;

- Horários;
- Paragens;
- Tipologia do veículo rodoviário utilizado.

Aplica-se a situações em que exista uma baixa procura na utilização do transporte público regular ou quando esse transporte não responda às necessidades dos cidadãos, como em regiões de baixa densidade populacional, ou em períodos de menos procura."[11]

1.2 Motivação

Agora que já sabemos o que é o transporte de passageiros flexível, perguntamo-nos: "Mas isso é importante *porquê?*-- a resposta não poderia ser mais simples:

Em zonas com baixa densidade populacional ou baixa procura por transportes coletivos, os transportes tradicionais, muitas vezes, não conseguem servir a população, uma vez que não é rentável fazer mais percursos ou mais horários quando muitas vezes um autocarro de 50 lugares viaja com 10 passageiros. Um exemplo prático seria uma cidade fora das grandes áreas metropolitanas de Portugal (Lisboa e Porto). Tomando o exemplo de Paços de Ferreira, uma cidade com cerca de 56 mil habitantes e a cerca de 30km do Porto, possui 12 freguesias. O transporte intra-município e inter-freguesia é escasso. Existem apenas 2 autocarros por dia para a sede do município a partir das freguesias. Isto causa desigualdade nas populações.

Para ilustrar imaginemos uma pessoa que precise de ir, por exemplo, a uma repartição de finanças, terá de viajar de autocarro de manhã e só poderá voltar da parte da tarde (não podendo excluir o cenário de, aquando da partida do último autocarro, a pessoa ainda não se encontrar na paragem, esta fica com três opções: consegue um meio de transporte particular, aluga um táxi ou um *Uber*, ou espera pelo dia seguinte). Esta situação, com estes constrangimentos, para uma viagem de 3km. Estas situações têm todo o potencial para serem melhoradas com a implementação de um serviço de transporte flexível de passageiros, que serve melhor as necessidades destas populações.

Um estudo realizado em Amesterdão[9] pretendeu verificar quais os efeitos da substituição integral de um serviço regulado ("tradicional"), por um baseado em TPF, e as conclusões não podiam ser mais claras. A substituição de um sistema regular por um sistema TPF corrobora a teoria da melhoria de serviço das populações: apesar da diminuição de passageiros transportados, a lotação média de cada veículo foi superior, a satisfação global das populações aumentou e houve uma diminuição significativa da emissão de gases com efeito de estufa.

Para além deste factor, existe também um outro associado à transição climática. A utilização de transportes coletivos em detrimento de transportes individuais (i.e. carro), inserida na mobilidade inteligente e mobilidade sustentável, é uma das formas mais importantes de reduzirmos emissões e atingirmos a neutralidade carbónica. Neste sentido, já foi criada legislação, no Decreto-lei nº 60/2016, que esclarece a prestação deste tipo de serviços e clarifica as regras necessárias para este tipo de transportes e garante o seu enquadramento legal.

1.3 Objetivos

1.3.1 Integração com solução atual

Um dos principais objetivos desta dissertação é construir uma nova aplicação móvel, mas que integre com a atual solução da HeyBus-que permite cálculo de rotas, custos, horários e toda uma gestão de backoffice. O objetivo será utilizar estes dados (estas funções) na primeira fase de interação com o utilizador, previamente ao momento de pagamento.

1.3.2 Facilidade de uso

A facilidade e acessibilidade da aplicação tem de ser um ponto-chave no desenvolvimento da mesma.

O grande objetivo é que seja uma aplicação passível de ser utilizada por qualquer pessoa, com o mínimo de *'tech-savvy'* possível. Neste sentido, as diferentes páginas, os diferentes ecrãs da aplicação terão em conta isto mesmo, e serão adoptadas as boas práticas para este objetivo.

Por outro lado, é um aspeto importante o segmento de utilizadores a atingir – o máximo possível. Para este efeito, é necessário escolher e adaptar tecnologias de modo a que não sejam restritivas para a maioria do público-alvo.

1.3.3 Minimização de custos

Um forte compromisso desta dissertação é impor o mínimo de alterações possível à infraestrutura existente por parte das empresas de transporte. Não deve ser necessário instalar novos equipamentos, quer sejam de validação de bilhetes, emissão, etc. Com isto em mente foram pesadas as tecnologias para a validação dos bilhetes, nomeadamente NFC vs QR-Codes, sendo a última a que implica menores custos, mais acessibilidade (do lado de quem emite e de quem valida) e, no fundo, mais adequada a esta temática.

1.3.4 Multi-dispositivo

Como referido no ponto 1.3.2, um dos objetivos é a não restrição de utilização com base em especifiações de hardware, tentando restringi-las ao mínimo possível de modo a obter uma maior base de utilizadores. Neste âmbito, a aplicação móvel estará disponível nas suas versões Android e iOS.

1.3.5 Privacidade e Segurança

A questão de privacidade dos dados e segurança dos mesmos é algo que não é alheio. Com a evolução do conhecimento e eventos recentes, torna-se cada vez mais importante. De facto, toda a experiência dentro da aplicação deve ser segura e os dados dos utilizadores não ser comprometidos. Para tal, a informação do passageiro deve ser acessível a ele e apenas a ele, sem haver acessos de qualquer outro – passageiro, motorista, backoffice.

1.4 Estrutura do Documento

O restante do documento organiza-se da seguinte forma: no capítulo 2 é feita uma revisão do estado da arte, construindo um conhecimento sobre projetos relacionados, as temáticas de DRT e bilhética móvel, e mesmo sobre as diferentes tecnologias utilizadas. No capítulo 3 é feita uma reflexão sobre o problema e apresentada uma proposta de solução, bem como as diferentes tarefas e uma vista geral das mesmas. No capítulo 4 é abordada a arquitetura e o modelo da aplicação e dos diferentes componentes, bem como a arquitetura dos mesmos na atual plataforma *web*. Passamos à implementação no capítulo 5, com alguns detalhes específicos a serem referidos aqui. No capítulo 6 é prosseguido este fio condutor, com as diferentes interfaces implementadas a serem expostas e detalhadas. É ainda neste capítulo que são analisados os resultados das experiências realizadas. Por fim, no capítulo 7 são apresentadas as conclusões desta dissertação e o trabalho futuro a desenvolver.

Capítulo 2

Revisão da Literatura

Visão Geral

Neste capítulo serão revisitados trabalhos neste campo, bem como as tecnologias deste domínio a ser utilizadas para alcançar o objetivo da dissertação.

2.1 Estado da Arte

Atualmente em Portugal e com a legislação aprovada, começa a surgir o transporte de passageiros flexível, aos poucos e em algumas cidades. Em termos de trabalho relacionado, irá ser abordado em duas secções: Plataformas DRT e aplicações móveis que permitem aquisição e validação de títulos de transporte.

2.1.1 Transporte de Passageiros Flexível

2.1.1.1 Sit Flexi

A Sit Flexi [6] é uma plataforma criada para TPF em Coimbra, mas que funciona apenas com reservas telefónicas e as suas rotas estão disponíveis em PDFs [7]. A plataforma conta no momento com 16 rotas nos diferentes municípios da região de Coimbra, com diferentes horários. Para efetuar uma reserva, é necessário contactar a central de reservas até às 15h do dia anterior à viagem Esta é a solução mais recorrente em Portugal, principalmente no interior do país (onde há mais potencial para o TPF).

2.1.1.2 Transporte a Pedido-Alentejo

Esta solução de TPF chama-se "Transporte a Pedido"e está implementada na região Alentejana, com o propósito de servir esta mesma região. Os municípios de Vendas Novas, Reguengos de Monsaraz, Moura, Beja, Mértola e Odemira fazem parte da plataforma que pretende alargar o serviço a todos os concelhos do Alentejo. Os transportes são assegurados por parcerias com 15 operadores de transporte da região[1].

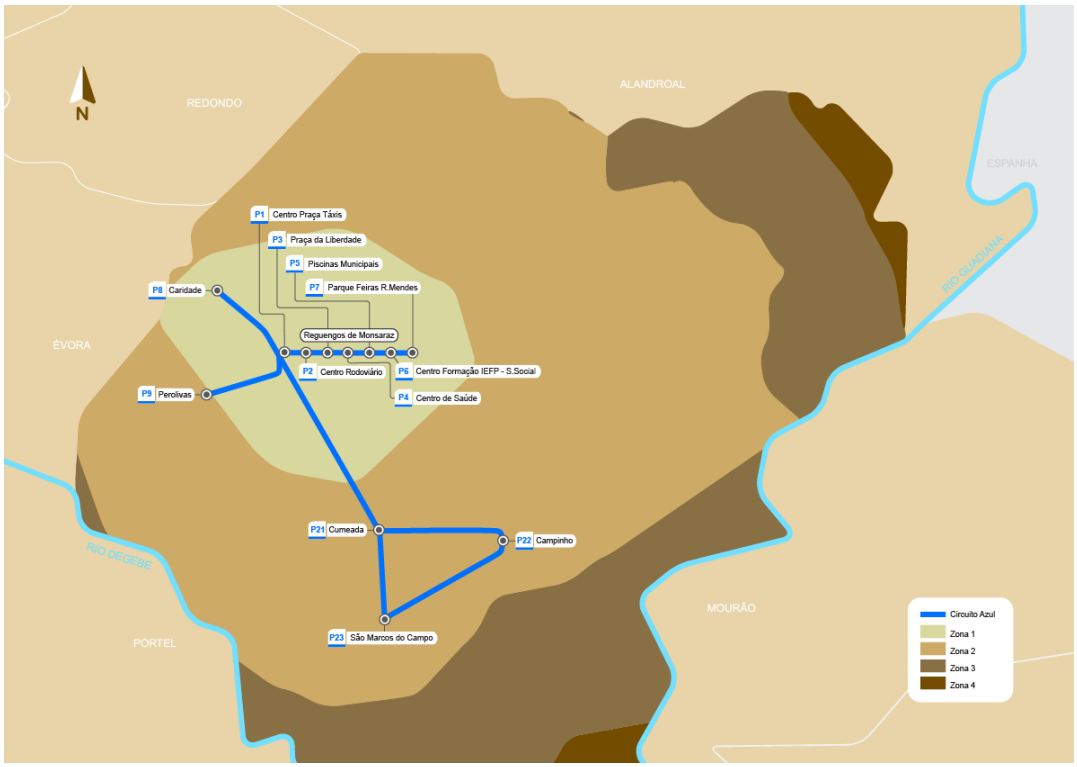


Figura 2.1: Circuito Azul Reguengos de Monsaraz

Os percursos e horários estão disponibilizados numa plataforma web disponível no seu website[2]. Esta plataforma a visualização de diferentes percursos e horários para diferentes concelhos. É possível, escolhendo um determinado concelho, verificar a existência de "Circuitos", ou percursos, diferentes, com alguns concelhos a possuir 9 circuitos diferentes (Odemira).

Depois, para cada circuito, existem vários horários, visualizáveis através da seleção do circuito. Quando o utilizador tiver escolhido qual a conjugação que melhor se adapta às suas necessidades, poderá fazer a sua reserva. Esta, similarmente à solução 'Sit Flexi', terá de ser feita via telefónica e até às 12:30 do dia útil anterior à viagem[1].

2.1.1.3 HeyBus & MOBUS

A OPT (Otimização e Planeamento de Transportes, proponente e parceira externa desta dissertação), desenvolveu a plataforma Hey Bus como solução para TPF. Esta plataforma contém uma panóplia de funcionalidades desde o apoio ao operador que pode inserir os detalhes de zonas, tarifas, paragens etc, o apoio ao cliente com a introdução de origem e destino e o cálculo, pela plataforma, da rota que otimiza custo e tempo.

A MOBUS é a adaptação da plataforma HeyBus para o Grupo Barraqueiro [15], disponível em Mafra e na Maia, apenas na forma de web app, permite fazer reservas mas sem pagamento, sendo este efetuado no veículo de transporte.

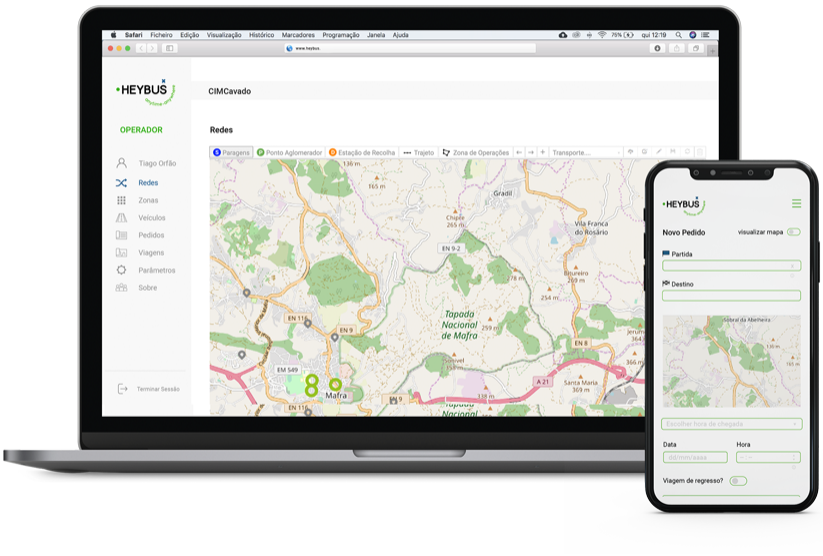


Figura 2.2: Aplicação operador HeyBus

2.1.1.4 Padam

O projeto PADAM, adquirido pela Siemens Mobility, tem as mesmas funcionalidades da MOBUS e está disponível na versão móvel, mas não está disponível em Portugal.

2.1.2 Bilhética Móvel

2.1.2.1 Anda

A Anda[19] é uma aplicação móvel, desenvolvida para os Transportes Intermodais do Porto, que permite viajar nos transportes públicos Andante na região do Grande Porto. Permite a seleção de viagem, aquisição de bilhetes e validação dos mesmos nos validadores existentes (nas estações de metro, autocarros, etc). Esta validação é feita via NFC. O pagamento dos bilhetes é feito via associação a uma de duas formas de pagamento disponíveis: cartão Multibanco ou cartão de crédito/débito.



Figura 2.3: Aplicação Anda

Durante a viagem é possível acompanhar a viagem (o percurso) via aplicação em tempo real.

2.1.2.2 Transdev Go

A Transdev disponibiliza, para algumas cidades inglesas (Lancashire e Yorkshire), um aplicação móvel onde é possível ver acompanhar os autocarros em tempo real, ver percursos e horários, planear viagens e comprar bilhetes. Estes bilhetes podem ser adquiridos via Google Pay [10]. Após a sua aquisição, cada bilhete tem associado um QR-Code que serve para a sua validação no autocarro correspondente.

2.1.3 Validação de bilhetes

A criação de títulos de transporte remonta às origens dos transportes públicos. Da mesma forma, também a sua validação tem início nessa altura. As formas mais tradicionais de validação de títulos de transporte incluem a sua verificação visual, como ainda hoje acontece nos transportes de Paris com os títulos de viagem ocasionais. Estes títulos são mini-cartões, que, ao serem utilizados (passando num pórtico), são carimbados com a data e hora atuais e a zona dos transportes. Quando o agente de transportes quiser validar o bilhete, tudo o que tem a fazer é uma verificação visual.
Esta mesma verificação acontecia também em Portugal, onde os bilhetes eram furados/cortados pelo agente, fazendo assim a sua validação ("picando"o bilhete, dando origem ao apelido desta profissão, i.e., "pica"bem como à expressão "andar à pica", i.e., andar sem bilhete).

Avançando para as formas mais contemporâneas de validação, inserem-se os típicos cartões, como é exemplo o Andante, com a sua validação a ser feita por RFID. O chip passivo no cartão é lido pelo sensor que valida (ou não) a viagem.
Como referido na secção anterior, estes validadores do Andante também podem validar bilhetes comprados na aplicação 'Anda', através da tecnologia NFC, que se relaciona intimamente com RFID, sendo o NFC uma sub-tecnologia RFID.

Aplicado a outras áreas, mas também nos transportes, começa a ser a tecnologia de QR-codes. Estes códigos, que já vinham a ser utilizados, ganharam outra dimensão com a pandemia da COVID-19 e a sua propagação em massa começou. O exemplo mais prático de validação por código-QR dá-se com os certificados digitais da UE, tanto de recuperação, vacinação e testagem. Estes códigos eram gerados pela app MySNS (versão móvel e webapp), e permitiam a circulação de pessoas com base na sua validação, numa aplicação específica para este efeito. Os códigos QR revelaram-se bastante eficientes e mesmo seguros, embora algumas tentativas de usurpação e falsificação ([21] [5]). Em bilhetes de concertos e espetáculos os códigos QR são também cada vez mais utilizados.

2.1.4 Tecnologias

Nas subsecções seguintes serão introduzidos os conceitos relacionados com as tecnologias latentes nesta dissertação, bem como as que serão desenvolvidas para o sucesso da mesma.

2.1.4.1 Sistema Operativo Móvel

Os *smartphones* têm proliferado e há quem os veja como um acessório, como uma peça de vestuário, ou até como algo indispensável. A verdade é que os sistemas operativos para smartphones não se têm diferenciado, principalmente devido a duas grandes competidoras neste mercado: Apple e Google. Estas empresas desenvolveram, respetivamente, iOS e Android, dois sistemas operativos que, em janeiro de 2022, representavam um total de 99.23% de do *market share* à escala global, com o sistema operativo Android a ser o líder do mercado com 69.74% comparativamente aos 29.49% do sistema operativo desenvolvido pela Apple[18].

2.1.4.2 React Native

React Native é uma framework open-source para a linguagem de programação JavaScript, desenvolvida pela Facebook (agora META Inc.) e introduzida em 2015. React Native é semelhante a ReactJS, mas para aplicações móveis. As aplicações são escritas em JSX, um encontro entre XML e JS. Desta forma, torna-se fácil ao programador desenvolver em alto-nível e com muitas bibliotecas e packages para diferentes funcionalidades pretendidas. No seu livro "Learning react native: Building native mobile apps with JavaScript", Eisenman explica que "no *background*, a *"bridge"* de React Native chama as funções nativas de rendering via API em Objective-C (para iOS) e Java (para Android). Assim, a aplicação terá componentes nativos renderizados, e não webviews"[8].

2.1.4.3 Node Package Manager - npm

Node Package Manager, mais conhecido por npm, tem como papel "tornar o desenvolvimento em JavaScript elegante, produtivo e seguro"[13]. npm é um ponto de partilha de código entre desenvolvedores através de *"packages"* e *"modules"*. Em termos práticos, no desenvolvimento de uma aplicação, no caso desta dissertação em React-Native, se por exemplo pretender inserir um calendário para escolha de data, ao invés de desenvolver um calendário de base, posso apenas encontrar um *package* de npm que permite importar esse componente e utilizá-lo, como por exemplo o *@react-native-community/datetimepicker* [16].

2.1.4.4 QR-Code

QR-Code significa Quick-Response Code, foi criado pela empresa automóvel Denso Wave em 1994 e é, basicamente, um código de barras bi-dimensional. Os códigos QR permitem codificar informação nesta espécie de "código de barras", como se fosse uma imagem. Os 3 quadrados distintivos permitem a descodificação do código QR.



Figura 2.4: Código QR versão 3

A utilização de códigos QR tem aumentado, com a sociedade em geral a ter conhecimento dos mesmos e para que servem, com a implementação dos mesmos nos certificados digitais durante a pandemia de COVID-19.

Atualmente no mercado existem soluções de bilhética móvel com validação de bilhetes via código QR. Um exemplo é a Yapsody[22]. A Yapsody é uma plataforma que está principalmente

construída para conferências e eventos deste tipo. Permite a criação de eventos, bilhetes, envio de bilhetes e, com mais relevância para este tema, a validação dos mesmos via QR-Code.

Uma nota importante é relativamente aos riscos da utilização de códigos QR. Um possível risco que acarreta a utilização de QR-Codes é conter conteúdo malicioso, algo que poderia ser alcançado, por exemplo, com a codificação de um URL malicioso ou código JavaScript. Aquando da validação, o dispositivo que valida poderia sofrer, potencialmente, um ataque informático.

2.2 Resumo

Neste capítulo, pudemos aprofundar os diferentes conceitos ao cruzá-los com as diferentes implementações existentes. Como foi possível perceber, esta Dissertação introduz-se num meio novo, uma vez que a todas as plataformas existentes falta algo, que nesta Dissertação estará contemplado: ora falta mobilidade, ora falta a funcionalidade de aquisição de bilhetes, ou ainda falta a aplicação a um contexto DRT. No fundo, falta uma integração com métodos de pagamento para que a experiência seja seamless, fluida, que permita a aquisição dos bilhetes sem sair da plataforma, ligar a um call center e etc.

Capítulo 3

Descrição do Problema e Proposta de Solução

O problema que dá origem a esta Dissertação é a impossibilidade de circulação de pessoas em transportes públicos em meios rurais ou de baixa densidade. Enquanto que a modalidade de transporte flexível de passageiros vem colmatar este problema, o acesso a esta forma inovadora de transporte é ainda escasso, nomeadamente pela inexistência de plataformas móveis acessíveis ao seu público alvo e, no fundo, à grande população. À data deste documento não é possível a uma pessoa que queira realizar uma viagem num contexto TPF/DRT adquirir os seus bilhetes numa aplicação móvel.

A proposta de solução deste mesmo problema passa pela criação de uma aplicação móvel que permita ao utilizador adquirir bilhetes no seu *smartphone*, num contexto TPF. A aplicação deve ser criada e arquiteturada de tal forma que integre com a solução existente da plataforma HeyBus, da OPT. A aplicação permitiria o login de utilizadores, a introdução de origens e destino da viagem, seleção e compra do título de transporte, bem como a gestão do mesmo (cancelamento ou alteração) e visualização do histórico de bilhetes adquiridos e viagens realizadas. A integração é feita com o *backoffice* para obtenção de trajetos e custos, como referido anteriormente, e via *API*. Também haverá lugar a integração com os métodos de pagamento para conclusão da aquisição destes bilhetes.

O *backoffice* da plataforma HeyBus tem diversas funcionalidades, das quais se destacam o cálculo de trajetos com base nas paragens otimizando o custo, distância e tipo de veículo, a comunicação com a base de dados para guardar e buscar dados, que disponibiliza via *endpoints*.

Em termos de tecnologias, a *app* será desenvolvida em *React Native* e a comunicação com a plataforma existente via *API* (dados de trajetos, cálculos dos custos, etc). Quanto aos métodos de pagamento, a OPT está a desenvolver com a *SIBS* uma *gateway* de pagamento específica para o projeto, e nesta fase assume-se a utilização de *mock data* para simular o pagamento. Esta *gateway* permitirá pagamentos via débito direto, cartão crédito/débito, *PayPal* mas está dependente

da evolução do projeto da *SIBS*.

3.1 Modelos

3.1.1 Bilhete

Já relativamente aos bilhetes, estes contêm, obviamente, informação de origem e destino, data e hora, identificador do utilizador, bem como um identificador único para cada bilhete. Contêm ainda informação relativa ao número de passageiros e ao número de passageiros com mobilidade reduzida.

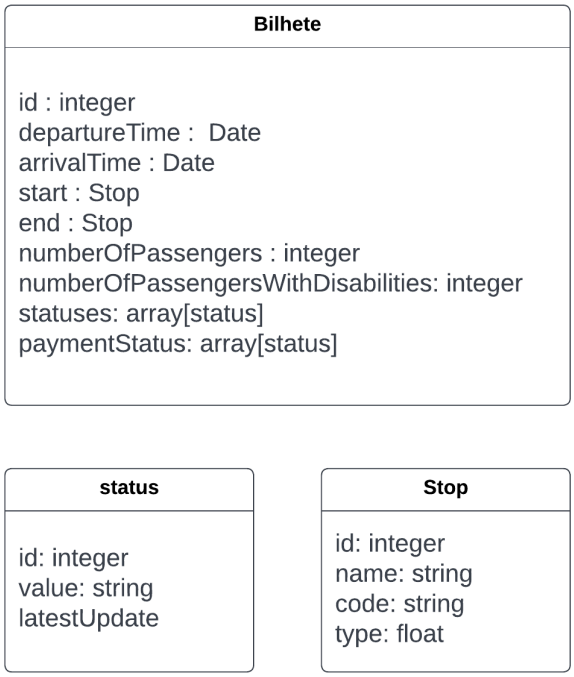


Figura 3.1: Modelo Bilhete

O modelo do bilhete acima apresentado inclui campos como o 'start' e 'end', que consideremos do tipo 'Stops'. Por sua vez, cada 'Stop', ou 'Paragem', tem um identificador, um nome, um código e um tipo associado.
Ainda no modelo do bilhete encontramos 2 'arrays', o de 'statuses' e o de 'paymentStatus', cada um deles compostos por elementos do tipo 'status', ou 'estado'. Um 'estado' é constituído por um id, um *value* e o *latestUpdate*, que é um *timestamp*, o momento do update. Para o campo *statuses*, o *status.value* pode ser um valor de entre 'Requested', 'Rejected', 'Accepted', 'Canceled' ou 'Finished', sendo que todos os pedidos começam com estado 'Requested' aquando da sua criação. Quanto aos elementos do array *paymentStatus*, o seu *status.value* pode ser entre 'Requested' ou 'Paid'.

3.1.2 Viagem

Na realidade da arquitetura da OPT, quando um pedido passa a estado 'Aceite', isto é, a operadora aceitou o pedido de viagem e criou uma viagem com um trajeto específico, então passa a haver um novo modelo, *'Trip'* ou 'Viagem', associado ao bilhete. Além desta associação, o modelo 'Viagem' possui ainda informação do preço a pagar pela viagem e disponibiliza informação relativamente ao veículo que irá realizar a mesma. O sub-modelo veículo, por sua vez, tem informação relativa à marca, modelo e matrículas do mesmo, e ainda indicação sobre a capacidade de passageiros e capacidade para passageiros com mobilidade reduzida. O campo *'Track'* significa neste contexto 'Percurso', e é um *array* de paragens, modelos 'Stop', que indicam o percurso da viagem que será realizada.

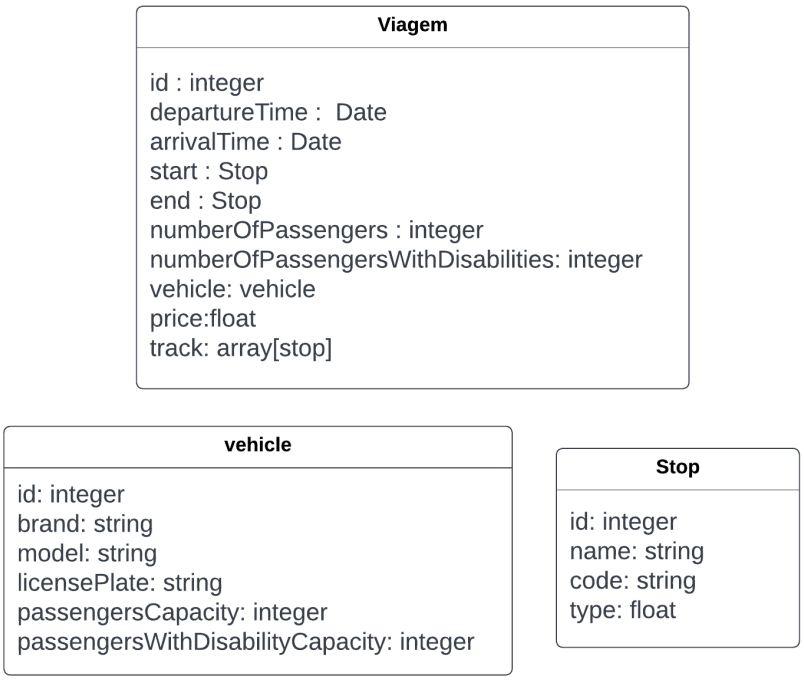


Figura 3.2: Modelo Viagem

3.1.3 Validação de bilhetes

Um ponto vital nesta dissertação, por parte da proponente e desenvolvedora da plataforma web existente, a OPT, é que a validação dos bilhetes não aumente de forma significativa o custo quer do lado do utilizador, quer do lado do 'validador' ou operador de transportes. Ou seja que por um lado não requeira aquisição de equipamentos para os veículos e, por outro lado, que não exclua utilizadores utilizando hardware específico. Este aspeto é de extrema importância e inclusivamente um dos objetivos desta dissertação. Neste sentido, a solução idealizada seria utilizar QR-Codes, gerados pelo identificador único do bilhete, lidos pelo motorista e assim validados. Os QR-Codes

contêm informação relevante para a validação do bilhete, nomeadamente espaço temporal da viagem, número de passageiros e identificador de pagamento. Esta informação seria codificada na geração do código-QR. O código-QR, sendo uma tradução de texto em imagem, teria uma *string* concatenada. Esta *string* conteria então a seguinte estrutura:

'flexibus#id_bil#origem#hora_partida#chegada#hora_chegada#id_pag#estado_pag#matrícula

, em que id_bil é o identificador do bilhete, origem é o nome da paragem de origem bem como chegada é o nome da paragem de destino. id_pag é o identificador do pagamento e estado_pag indica o estado de pagamento.
A descodificação é feita da parte do motorista. O sistema foi pensado e deve ser construído de tal forma que, mesmo em situações onde não exista rede móvel o código possa ser validado.

Capítulo 4

Arquitetura & Modelo

4.1 Visão Geral

O objetivo principal desta dissertação é a implementação de uma aplicação móvel que permita a compra de bilhetes de transporte flexível de passageiros através da plataforma da OPT (heybus+mobus). Nesta dissertação, de modo a implementar a prova de conceito (PoC) foi utilizada, como referido nos capítulos anteriores, a tecnologia React Native. Esta tecnologia foi escolhida em conjunto com a OPT, proponente da dissertação. As razões prendem-se com a sua extensa documentação e suporte, com uma grande comunidade por trás que torna React Native a principal escolha de framework para programação móvel (42% de utilização em 2020, segundo a Statista)[20].

Para efeitos de demonstração, a dissertação assume que os utilizadores dispõem de um telemóvel Android ou iOS, com ligação à internet bem como conta de utilizador na plataforma web da OPT.

O "flow"simplificado da aplicação consiste nos seguintes passos:

1. **Login na aplicação**
2. **Compra de bilhetes**
3. **Validação dos bilhetes**

4.1.1 Login

O protocolo utilizado para o login dos utilizadores na plataforma web da OPT (HeyBus e Mobus) é o OpenID.

4.1.1.1 OpenID

O protocolo OpenID, segundo a sua própria definição [14] funciona como uma camada de identidade por cima do protocolo OAuth2.0.

Um dos objetivos do OpenID é garantir o acesso do utilizador a um conjunto de websites. No caso da plataforma da OPT, esta autenticação permite o acesso à plataforma de utilizador (cujo url é `https://dev8.opt.pt:40008/`), página pessoal (`https://dev4.opt.pt:40004/`), caso seja operador permite o acesso a essa plataforma também (`https://dev4.opt.pt:40004/`). Além disto, claro, a autenticação permite também realizar todas as chamadas necessárias à API (`https://dev9.opt.pt:40009/api/v1/`). Como podemos observar, uma autenticação única facilita a navegação e os pedidos necessários. A utilização de OpenID permite devolver um token. Este token é utilizado nos pedidos feitos à API. Todos os pedidos realizados à API têm Bearer token como validação de autenticação.

4.1.2 Compra de bilhetes

Tal como na plataforma web da OPT, a compra de bilhetes não é imediata. Esta compra é realizada em 2 momentos: num primeiro momento, é feito o pedido de viagem, com os parâmetros adequados (paragem de entrada, saída, numero de passageiros, dia e hora de partida). Depois do pedido ter sido feito, o mesmo fica com estado 'Pedido'. De seguida, após o operador de transportes ter criado as viagens para os vários pedidos, utilizando um algoritmo de otimização de custos e tempo, o pedido do utilizador passa para o estado 'Aceite'. Neste estado, é possível fazer o pagamento.

Após o pagamento, o bilhete passa para o estado 'Pago', estado em qual o utilizador irá realizar a viagem, e pode passar ao próximo passo, a 'Validação dos bilhetes'.

4.1.3 Validação de bilhetes

Aquando da entrada do utilizador no meio de transporte, na paragem selecionada pelo mesmo no momento da realização do pedido, é necessária uma forma de validação do bilhete. Esta validação é garantida através de um QR-code, ou código QR. Este código é gerado com os campos do bilhete (ID do bilhete, número de passageiros, paragem de entrada e de saída, dia e hora da viagem). O motorista pode, então, nesse momento, validar o código QR e seguir a viagem.

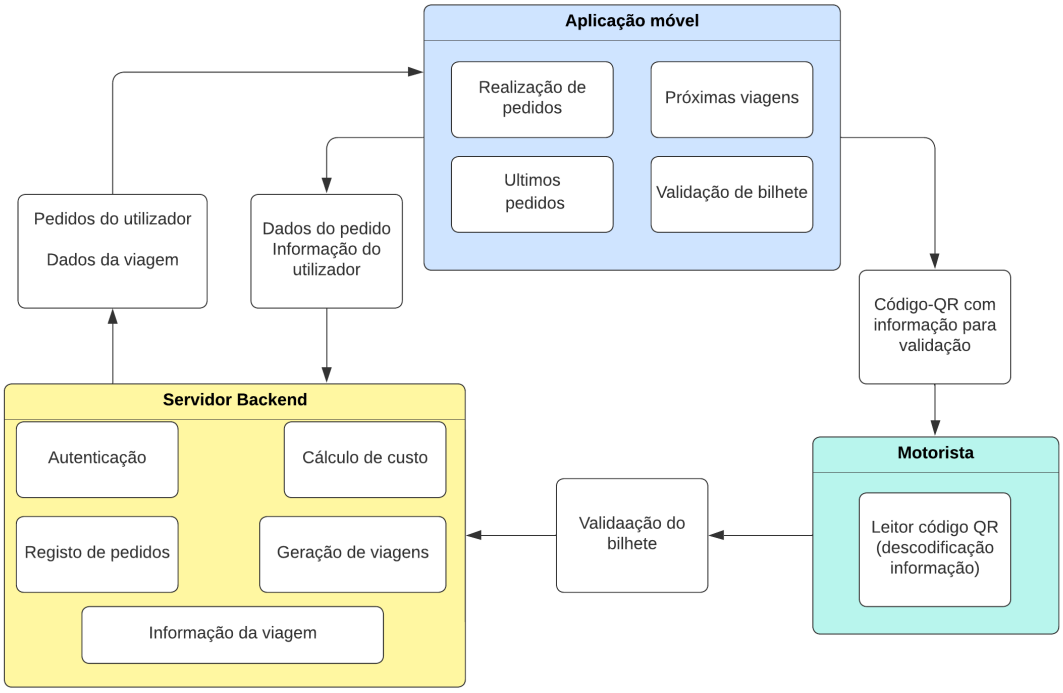


Figura 4.1: Arquitetura do sistema

4.1.4 Esquema de arquitetura

A estrutura apresentada na figura acima representa as trocas entre a Aplicação Móvel, o Servidor Backend e a Aplicação Motorista.

A comunicação entre a aplicação móvel e o servidor backend realiza-se exclusivamente através de chamadas API. O *request* pode ser feito para retornar pedidos de viagem, viagens, um pedido em específico. Também pode retornar as paragens (e as paragens de destino disponíveis com base na selecionada de partida) ou, também, pode ser um *request tipo POST*, para cancelar um pedido, pagar uma viagem.

4.2 Funcionalidades da Aplicação móvel

Nesta secção são apresentadas as diversas funcionalidades a ser criadas e respetivas ligações e/ou interações com outros componentes, nomeadamente servidor ou aplicação motorista.

4.2.1 Login

O *login* é a 'porta de entrada' para a aplicação. Para o funcionamento global da aplicação é necessário a implementação do login, com Open-ID. É possível implementar autenticação com Open-ID em React Native utilizando *packages* npm que permitem utilizar *webviews* ou de outra forma fazendo o login no browser e voltando à aplicação.

Para efeitos desta prova de conceito o *login* não foi implementado com Open-ID uma vez que não foram fornecidos os endpoints de autenticação. Por esta razão, é utilizado o *token* de autenticação da plataforma *web*.

4.2.2 Fazer pedido

A aplicação deve ser capaz de permitir a criação de um novo pedido de viagem. Para este efeito, uma página com vários campos de seleção para os diferentes elementos de um pedido:

- Paragem de origem
- Paragem de destino
- Data de partida
- Hora de partida
- Número de passageiros

Ao clicar no botão de submissão presente nesta página, a aplicação faz um pedido via API ao servidor, no url `https://dev9.opt.pt:40009/api/v1/request`, com o método 'POST' (após realizar uma validação prévia dos dados, obviamente–exemplo: data >= data de hoje). Os *headers* incluem o *token* de autenticação e o *body* do *request* contem os itens descritos na lista acima.
Em caso de resposta positiva, isto é, criação com sucesso de um pedido, é mostrada ao utilizador uma mensagem de sucesso, e o novo pedido já será visível na página de listagem de pedidos. Caso haja algum erro, tanto de *network* como da aplicação, uma mensagem com essa informação é mostrada ao utilizador, para que tente novamente ou entre em contacto com o suporte.

4.2.3 Listar histórico de pedidos

Nesta página deverão estar listados todos os pedidos efetuados pelo utilizador, com a informação básica de cada um, nomeadamente a data, hora de partida e chegada, número de passageiros

e estado do pedido (o estado do pedido pode ser: 'Pedido', 'Cancelado', 'Aceite', 'Pago', 'Finalizado' ou 'Rejeitado'). Esta listagem de pedidos é obtida *on-page-load* através de um pedido à API, com um método *'GET'* ao url `https://dev9.opt.pt:40009/api/v1/request`. Esta página também terá componentes de filtragem, isto é, componentes de seleção de estado ou períodos de tempo para reduzir os resultados da pesquisa. Neste último caso, os pedidos são feitos também por um *'GET' request* ao *url* `https://dev9.opt.pt:40009/api/v1/request?firstDate=_first_date&lastDate=_last_date`

4.2.4 Mostrar detalhes de um pedido

Esta página está acessível pela anterior- Listar Histórico de Pedidos.

Clicando num dos pedidos listados, o utilizador é levado para uma nova página, onde é apresentado o pedido com mais detalhes e um conjunto de ações dependente do estado do pedido. Nesta página são apresentados os detalhes do pedido com uma variabilidade:

4.2.4.1 Estado 'Pedido'

Os pedidos de viagem neste estado aguardam aceitação ou rejeição por parte do operador. Como tal, ainda não há dados de viagem a ser apresentados. O componente desta página deve mostrar os detalhes do pedido, incluindo a data e hora pedida da viagem, o número de passageiros e as paragens de origem e destino. Todavia, existe uma ação possível nesta página para estes pedidos: o cancelamento de um pedido de viagem. Este cancelamento é feito enviando um request tipo *'PATCH'* para o *url* `https://dev9.opt.pt:40009/api/v1/request/idTicket/cancel`, onde *idTicket* é o identificador do pedido de viagem.
Em caso de sucesso no cancelamento, é mostrada ao utilizador uma mensagem indicando o sucesso e que o pedido estará com o estado atualizado na página 'Listagem de pedidos'. Em caso de falha, é mostrada uma mensagem de erro com a indicação de tentar novamente ou contactar o suporte.

4.2.4.2 Estado 'Cancelado'

Os pedidos de viagem neste estado passaram pelo processo de criação e, em algum momento, foram cancelados pelo utilizador antes da viagem ser aceite. Desta feita, não há dados de viagem a ser apresentados. O componente desta página deve mostrar os detalhes do pedido, incluindo a data e hora pedida da viagem, o número de passageiros e as paragens de origem e destino. Nesta página não há ações a ser tomadas sobre o pedido, apenas é possível regressar à pagina 'Listagem de pedidos'.

4.2.4.3 Estado 'Aceite'

Os pedidos de viagem neste estado foram criados pelo utilizador e aceites pelo operador, isto é, foi criada uma viagem pelo operador que permite que o pedido se torne numa viagem. Assim, nesta página é importante que, além de mostrar os detalhes do pedido (incluindo a data e hora

pedida da viagem, o número de passageiros e as paragens de origem e destino), sejam também mostrados detalhes da viagem, nomeadamente a tipologia e identificação do veículo de transporte, no qual se inserem os campos 'Marca', 'Modelo', 'Matrícula', 'Capacidade' e 'Capacidade para pessoas com mobilidade reduzida', nunca olvidando o preço da viagem, que também deve ser apresentado. A ação principal desta página para estes pedidos é o pagamento da viagem, apresentado na próxima secção(4.2.5. O pagamento é feito enviando um request tipo *'POST'* para o *url* `https://dev9.opt.pt:40009/api/v1/request/idTicket/pay`, onde *idTicket* é o identificador do pedido de viagem.
Em caso de sucesso no pagamento, é mostrada ao utilizador uma mensagem indicando o sucesso e que o pedido estará com o estado atualizado na página 'Listagem de pedidos'. Em caso de falha, é mostrada uma mensagem de erro com a indicação de tentar novamente ou contactar o suporte.

4.2.4.4 Estado 'Pago'

Os pedidos de viagem neste estado foram criados pelo utilizador, aceites pelo operador e pagos na aplicação (neste momento a única forma de pagamento é através da aplicação, uma vez que a plataforma web da OPT não tem esta funcionalidade implementada). A informação presente nesta página inclui todos os detalhes do pedido referidos na subsecção anterior (data e hora pedida da viagem, o número de passageiros e as paragens de origem e destino), e apresenta ainda os detalhes da viagem também referidos anteriormente (tipologia e identificação do veículo de transporte – 'Marca', 'Modelo', 'Matrícula', 'Capacidade' e 'Capacidade para pessoas com mobilidade reduzida') com um novo componente referente ao pagamento: uma indicação da data de pagamento e do valor pago. A ação principal desta página para estes pedidos é a geração de um QR-Code, referido com mais detalhe em 4.2.6
Em caso de sucesso no pagamento, é mostrada ao utilizador uma mensagem indicando o sucesso e que o pedido estará com o estado atualizado na página 'Listagem de pedidos'. Em caso de falha, é mostrada uma mensagem de erro com a indicação de tentar novamente ou contactar o suporte.

4.2.5 Fazer o pagamento de um pedido

Em pedidos com estado 'Aceite' surge uma das necessidades vitais que deu origem a esta dissertação: o pagamento de um pedido. Com efeito, o pagamento de um bilhete (e consequente validação) irá facilitar toda a experiência do utilizador. Na página de detalhe de um pedido na aplicação móvel, quando este tem estado 'Aceite', existirá um botão para realizar o pagamento, indicando a quantia a pagar. O processo de pagamento é, em teoria, feito através de uma *gateway* de pagamento desenvolvida pela **SIBS** especificamente para este efeito. Contudo, uma vez que esta *gateway*, à data da escrita deste documento, ainda não está numa fase madura do seu desenvolvimento, o pagamento da aplicação assume dados simulados e aleatórios para simular o pagamento. Como tal, depois desta geração de referência de pagamento na aplicação e simulação de sucesso no pagamento, a informação do pagamento do bilhete é enviada via API para o *url*

`https://dev9.opt.pt:40009/api/v1/request/idTicket/pay` com método *'POST'* onde *idTicket* é o identificador do pedido de viagem.

4.2.6 Gerar código-QR

Quando um pedido foi pago, surge a necessidade do utilizador ter uma forma de o poder validar. Assim, a solução proposta por esta dissertação para este problema é a utilização de um código QR. Na página de detalhe do pedido, quando ele tem estado 'Pago', clicando no botão 'Gerar Código-QR), a aplicação junta os detalhes mais significativos do pedido (id do bilhete, dia, hora, paragens, numero de passageiros), viagem e pagamento (preço, data de pagamento).

De forma a gerar os *qr-codes*, a aplicação utiliza um package *npm*, react-native-qrcode-svg [12].

Estes códigos-QR serão utilizados aquando da entrada no veículo de transporte por parte do utilizador. Com vista à utilização destes códigos e sua validação mesmo em zonas onde não há rede móvel, fenómeno infelizmente ainda presente em Portugal, como podemos verificar na ferramenta desenvolvida pela ANACOM, a plataforma tem.REDE [3]. Nestas situações, deve ser ainda possível validar os bilhetes. Isto é conseguido 'descodificando' o código-QR na aplicação do motorista. Como os dados estão em forma de *string*, e essa mesma *string* contém todos os dados necessários (informação de pagamento, paragem de origem e destino e data e hora da viagem, esta informação poderá ser vista pelo motorista com esta simples descodificação. Porém, caso haja ligação à internet, nesse caso o id do bilhete que vem no código-QR é utilizado para fazer uma chamada à API e, dessa forma, verificar a veracidade e validade do bilhete.

4.3 Casos de uso

Nesta secção são apresentados os diferentes (e principais) casos de uso da aplicação, e exemplificado o seu fluxo de execução. É apresentado um diagrama para cada caso de uso, bem como uma explicação do processo do mesmo, e dos passos necessários.
De forma a não repetir informação, são referenciadas secções anteriores que explicam determinados pormenores em maior detalhe.

4.3.1 Compra de bilhete

O primeiro caso de uso da aplicação, o mais importante, útil e, provavelmente, o mais comum, é o da criação de um novo pedido.
A figura 4.2 ilustra este caso de uso. O primeiro passo é o do *login*, que, em caso de sucesso, redireciona o utilizador para o menu principal da aplicação. Aqui, o utilizador seleciona a primeira opção, 'Próxima Viagem' e é levado para uma nova página, onde seleciona todos os campos referidos na secção 4.2.2.
Após todos os campos inseridos, e clicando em 'Criar Pedido', é feito um *request* ao *backend* através de um pedido à API, e cuja resposta é indicada ao utilizador em forma de *alert*.

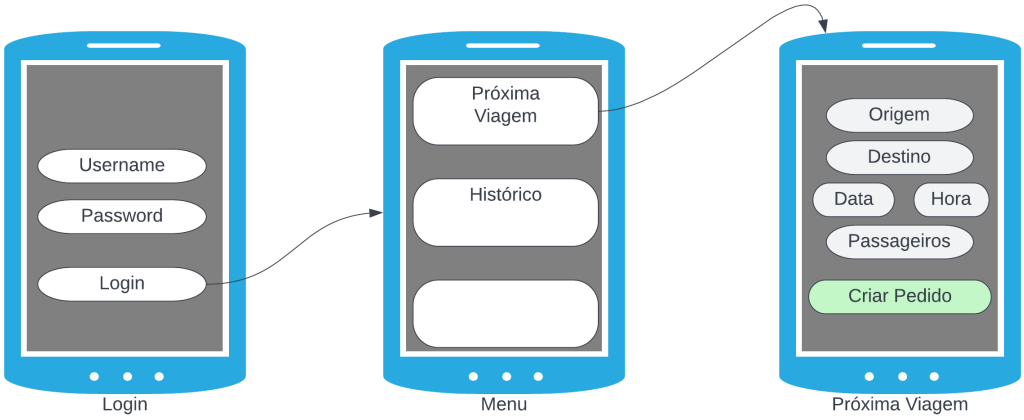


Figura 4.2: Criação de um novo pedido

4.3.2 Visualização de um pedido

Um caso de uso possível na aplicação é a visualização de um pedido. Para isto é necessário que seja feito o *login*, como em todos os casos de uso apresentados. Depois, a seleção no menu principal será a de 'Histórico'.
Ao fazer esta seleção, a aplicação irá fazer uma chamada à API para buscar todos os pedidos, como referido na secção 4.2.3. Os pedidos são apresentados e podem ser filtrados, bastando o utilizador premir a secção 'Data' ou 'Estado', os dois filtros presentes (e independentes). Esta filtragem permite uma maior facilidade na busca de um pedido específico.
Quando o utilizador encontra o seu pedido, na figura representado pelo 'Bilhete 1', basta premi-lo e será levado para a página de 'Detalhes do Pedido', onde poderá ver com mais detalhe o pedido e tomar alguma ação sobre o mesmo, sendo que esta ação depende do 'Estado' do pedido.
Todos os detalhes para os diferentes tipos de pedidos e todas as ações possíveis para cada um deles estão detalhadas na secção 4.2.4.

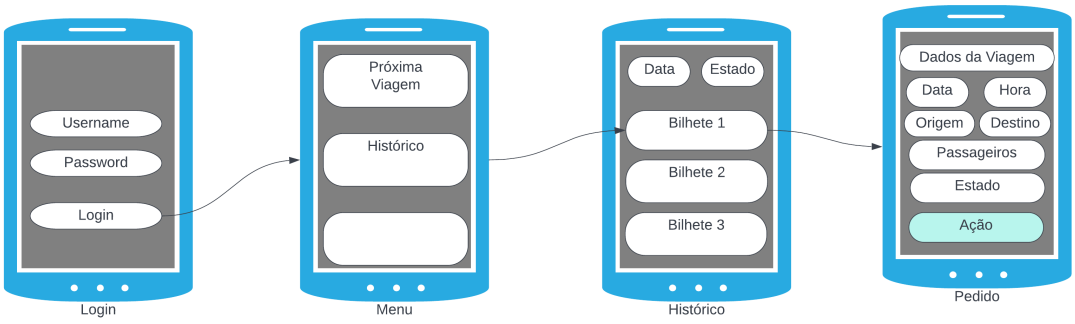


Figura 4.3: Visualização de detalhes de um pedido pedido

4.3.3 Cancelamento de um pedido

O cancelamento de um pedido é uma das ações possíveis sobre pedidos, neste caso sobre pedidos com estado 'Pedido'.
O *flow* da aplicação para cancelar um pedido é semelhante ao anterior, isto é, passando pelo *login*, selecionando a opção 'Histórico' no menu inicial e encontrando o bilhete que se deseja cancelar, somos levados à página 'Detalhes do Pedido' onde podemos então efetuar a nossa ação: cancelar.

Pressionando o botão de cancelamento, é feito um pedido à API para o mesmo efeito e o resultado é apresentado ao utilizador. Mais detalhes poderão ser encontrados em 4.2.4.1.

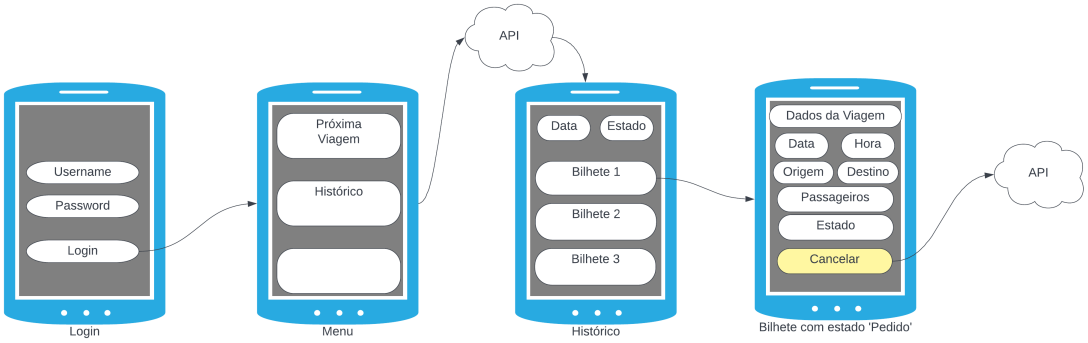


Figura 4.4: Cancelamento de um pedido

4.3.4 Pagamento de um pedido

Uma das ações mais vitais da aplicação é a possibilidade do pagamento de um pedido. Para este efeito, o pedido tem de se encontrar com estado 'Aceite', isto é, a viagem ter sido criada pelo utilizador. O percurso na aplicação passa pelo *login*, seleção da opção 'Histórico' no menu, procura do bilhete que se quer pagar e, pressionando-o, o utilizador chega à página de detalhes do pedido. Nesta página, além dos detalhes de um pedido 'Aceite', descritos em 4.2.4.3, podemos executar a ação de pagamento. Ao pressionar o botão, somos levados para o pagamento que, depois de efetuado, é feita a chamada à API para este comunicar o pagamento do bilhete e o resultado é mostrado ao utilizador na forma de alerta.

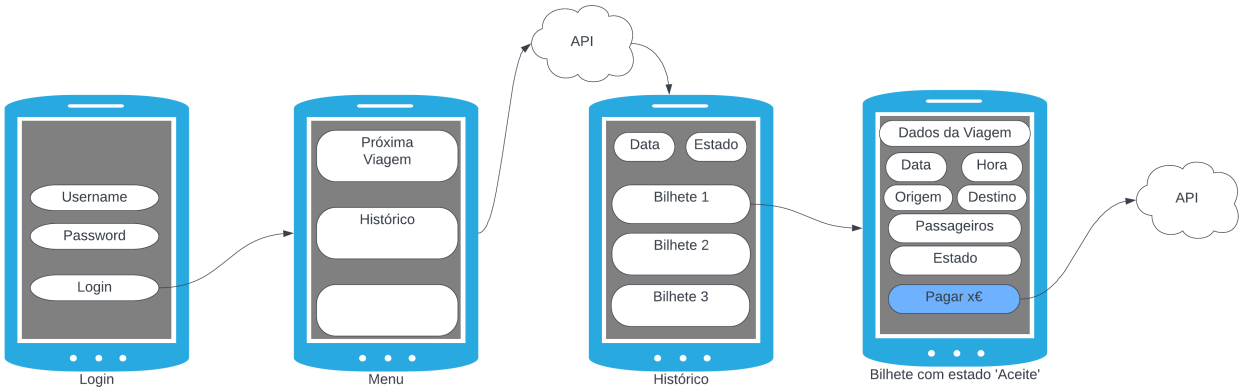


Figura 4.5: Pagamento de um pedido

4.3.5 Geração de um código-QR

De forma a poder validar um bilhete, o utilizador encontrar-se-á munido de um código-QR gerado especificamente para este efeito. Enquanto os pressupostos estão descritos em 4.2.6, o percurso dentro da aplicação é simples. Após o *login*, histórico e seleção do bilhete com estado 'Pago' para o qual o utilizador pretende gerar o código, encontra-se na página de detalhes do pedido. Nesta página, o utilizador pode ver todos os detalhes da viagem e do veículo e pode, também, gerar o código-QR. Para isso basta pressionar 'Gerar QR', que codificará os detalhes essenciais para a validação gerando o código-QR como exemplificado, pronto a ser validado pelo motorista do transporte.

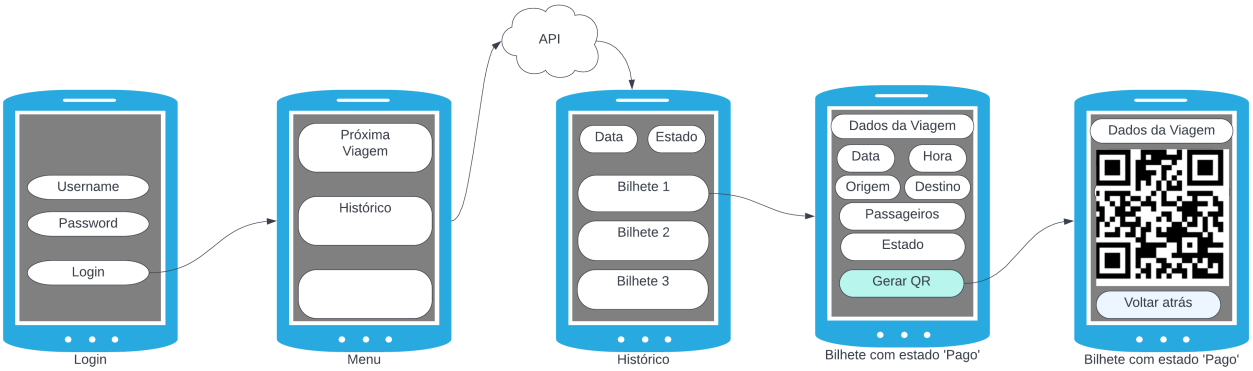


Figura 4.6: Geração de um código-QR num pedido 'Pago'

Capítulo 5

Realização do Projeto & Implementação

5.1 Visão Geral

Neste capítulo são detalhadas as diferentes tecnologias utilizadas no desenvolvimento da aplicação, bem como explicados alguns detalhes importantes na implementação dos diversos itens e páginas. Para além disso, a integração dos diversos componentes será referida neste capítulo.

5.2 Detalhes de implementação

Nesta secção são apresentados os principais detalhes de implementação acompanhados, quando necessário, de capturas de ecrã do código fonte.

5.2.1 Chamadas à API

Um pouco por toda a aplicação, e nomeadamente na página de 'Histórico de Pedidos', 'Gerir as minhas viagens', 'Encontrar uma viagem' e 'Bilhete' (quando estado é aceite ou pago), são feitas chamadas à API da OPT, para diferentes *endpoints*, para obtenção da informação necessária.
A estrutura é a apresentada na figura 5.1.

```
89    const response = await fetch('https://dev9.opt.pt:40009/api/v1/request', {
90      method: 'GET',
91      headers: {
92        Authorization: 'Bearer ' + token,
93      },
94    });
95    const json = await response.json();
96    if (response.status !== 200) {
97      showToast(
98        'error',
99        'Erro ao buscar os pedidos.',
100       'Pode tentar refrescar a página ou contactar o suporte.',
101     );
102   }
```

Figura 5.1: API call

O pedido é feito de forma assíncrona, ao *endpoint* apresentado no *url*. O método do pedido ('POST', 'PATCH', 'PUT', 'GET' ou 'DELETE') depende do tipo de *endpoint* e da ação que queiramos fazer. O exemplo apresenta a chamada feita na página de histórico de pedidos e, como tal, o método é 'GET'.
Quanto à autorização, é utilizado um Bearer token, uma vez que não foi possível implementar o Login com Open-ID.
A resposta é tratada na linha 95, também de forma assíncrona. Isto permite que a aplicação continue a ser renderizada enquanto espera a chamada e atualize depois da resposta sem nunca interromper a execução. Para este efeito, antes da chamada à API é colocada a *true* uma variável, *loading* que indica se a aplicação está à espera da resposta ou não. Em caso positivo, i.e. *loading* == *true*, é renderizado um componente de loading, como podemos ver na imagem seguinte.

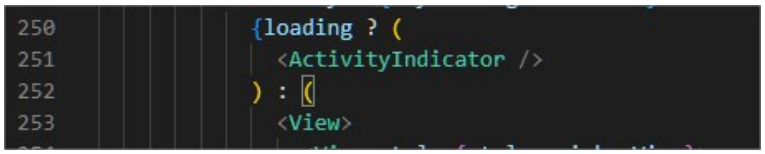


Figura 5.2: Condição de *loading*

5.2.2 Navegação

A navegação na aplicação é feita utilizando *drawer-navigation* (@*react-navigation/drawer*) [17]. Este *package* permite a navegação com uma 'gaveta', para além da navegação por Menu também implementada. A navegação em si, no código, é feita com

```
navigation.navigate('nome_ecra', {params})
```

onde *nome_ecra* é o nome da página para onde se quer navegar e *params* são os parâmetros a enviar, na forma de *nome:valor*.

5.2.3 Dados

A aplicação móvel é alheia aos dados. Isto é não há dados a ser armazenados na aplicação, quer fisicamente no dispositivo ou numa base de dados local. Os dados são temporários durante a utilização da aplicação. Todos os dados presentes são *da* e *para* a *API*, o que permite maior segurança ao utilizador e menos espaço ocupado em armazenamento.

Capítulo 6

Resultados

6.1 VIsão Geral

Neste capítulo são apresentadas as diversas interfaces gráficas da aplicação, indicando o ambiente de execução e instruções de operação e execução. São ainda apresentados e discutidos os testes realizados e os seus resultados.

6.2 Operação & Execução

No desenvolvimento e testes da dissertação, houve dois ambientes de execução da aplicação móvel. A aplicação foi instalada num emulador a correr em Windows10-64bits. O emulador Android utilizado é o modelo Google Pixel 2, cujas especificações detalhadas podem ser encontradas aqui https://www.gsmarena.com/google_pixel_2-8733.php. Outro ambiente de execução foi o Android 11, em execução num Xiaomi Mi 11i (especificações detalhadas: https://www.gsmarena.com/xiaomi_mi_11i-10777.php)

6.3 Interfaces

6.3.1 Login

Na figura abaixo podemos ver a primeira página da aplicação: a página de Login. O seu objetivo é claro: autenticar um utilizador para poder ter a experiência da aplicação.
Contudo, não foram fornecidos os endpoints necessários para poder realizar o protocolo de autenticação na aplicação móvel e, como tal, esta página na prova de conceito não autentica o utilizador como acontece na plataforma web.

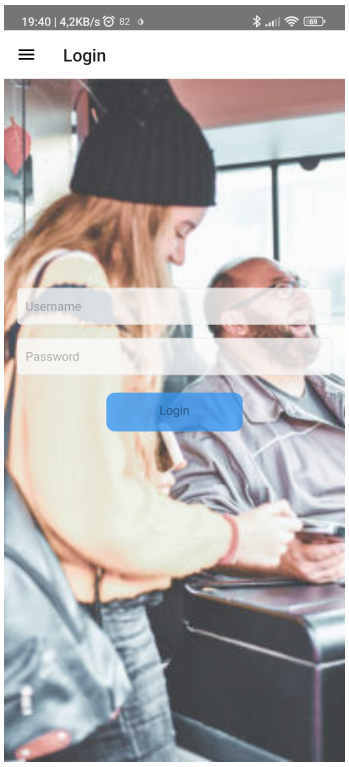


Figura 6.1: Interface da página de Login

Contudo, esta página tem utilidade. Nesta página podemos inserir o *token* que encontramos na plataforma web, inseri-lo no campo 'Password' e assim fazer uma utilização da aplicação.

6.3.2 Menu Inicial

Nesta secção são apresentadas as diversas opções ao utilizador: fazer um novo pedido, gerir as suas viagens, ver o histórico de pedidos ou aceder ao seu perfil. Todas estas opções serão exploradas nas próximas secções.

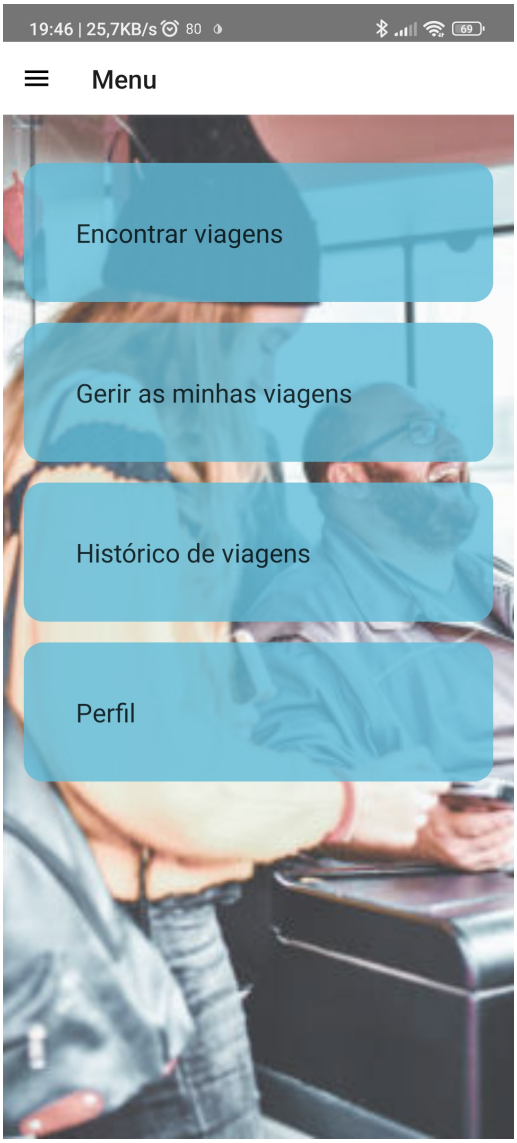


Figura 6.2: Menu Inicial

6.3.3 Encontrar viagem

Começamos pela primeira opção do menu inicial: 'Encontrar Viagens'. Como o nome indica, esta página permite fazer um novo pedido.

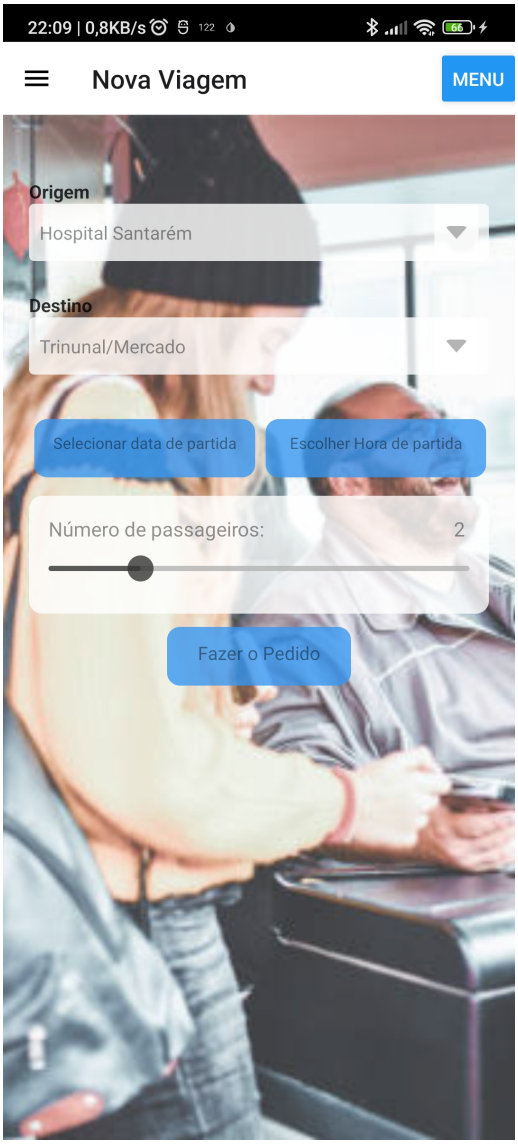


Figura 6.3: Interface 'Nova Viagem'

Para este efeito, é necessário, antes de mais, introduzir a paragem de entrada no veículo, ou a 'Origem'. Selecionando uma das paragens disponíveis no *dropdown*, é feita uma chamada API que retorna as paragens de destino disponíveis com base na *Paragem de Origem* selecionada. Assim, estas paragens são mostradas num novo *dropdown* que aparece ao utilizador, para seleção da *Paragem de Destino*.

Após este passo, é preciso selecionar a *Data da Viagem*. Pressionando o botão é possível selecionar a data num calendário. Para selecionar a hora de partida a seleção é semelhante, havendo um *pop-up* de um modal com o relógio para o utilizador selecionar a hora mais conveniente. Antes

de criar o pedido, é necessário introduzir o número de passageiros da viagem, deslizando o *slider* disponível com esta indicação. Após todos os passos estarem concluídos e verificados, basta clicar em 'Fazer o pedido', para este ser criado.

Dependendo da resposta da API, um alerta é mostrado ao utilizador. Na imagem abaixo podemos ver um pedido criado com sucesso, e com a mensagem a indicar isso mesmo, bem como a possibilidade de vermos esse bilhete na página de listagem.

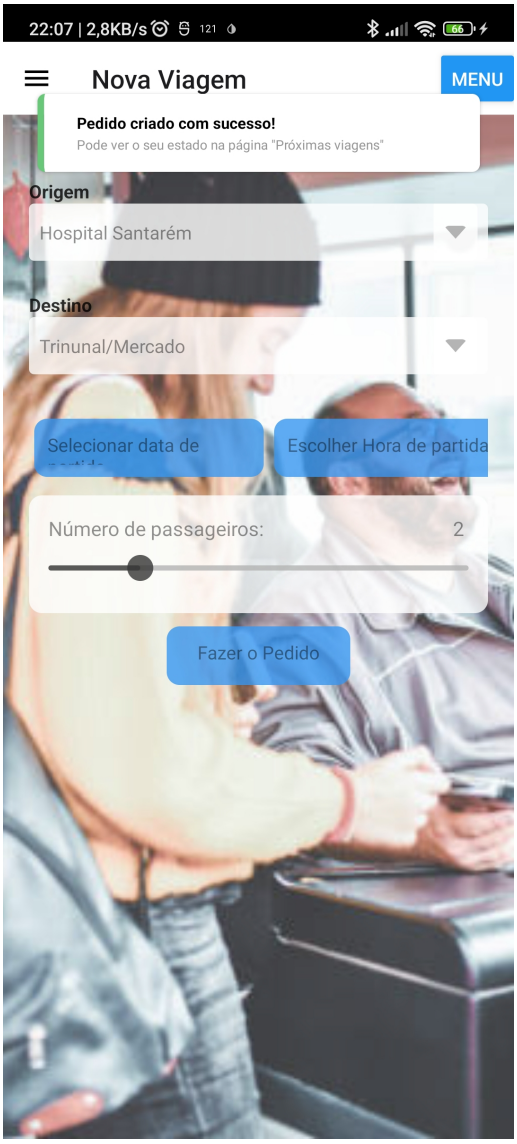


Figura 6.4: Criação de pedido com sucesso

Em caso de erro na criação de pedido, é mostrado um alerta de erro ao utilizador, explicando-lhe as alternativas.

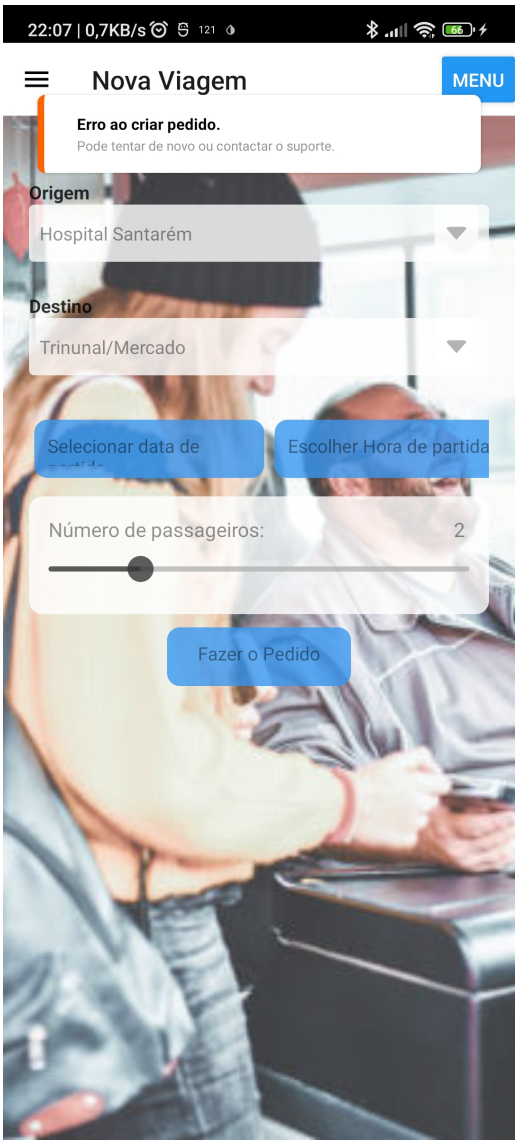


Figura 6.5: Erro na criação do pedido

6.3.4 Listagem de pedidos

Nesta página é apresentada uma listagem dos pedidos, retornados da chamada à API referida nos capítulos anteriores.

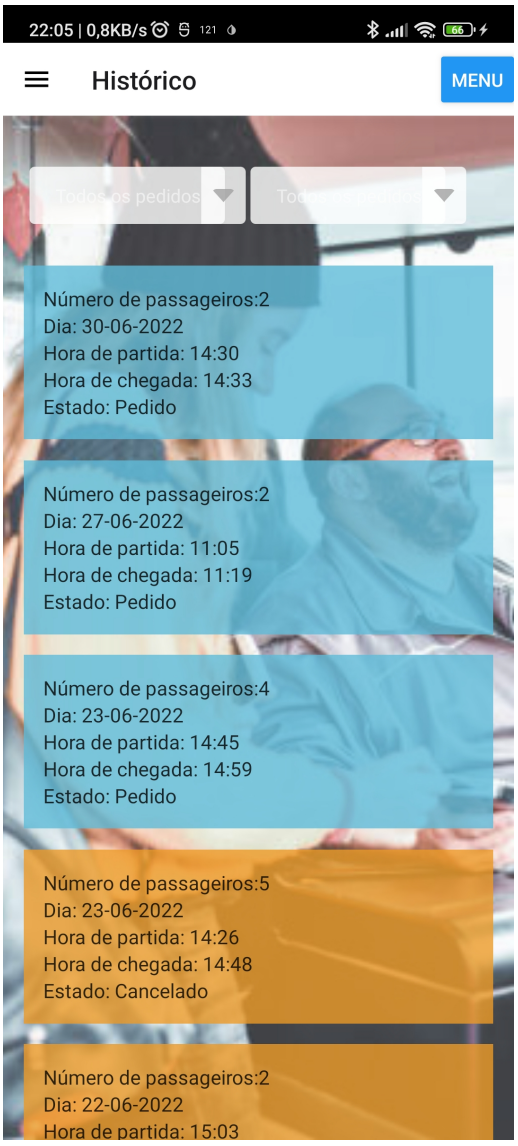


Figura 6.6: Histórico de Pedidos

Os pedidos podem ser filtrados por espaço temporal (últimos 7 dias, último mês, próximos 7 dias, próximo mês) ou por estado do pedido (Apenas pedidos com estado 'Pedido', 'Cancelado', 'Aceite' ou 'Pago'). Uma combinação destes filtros é também possível. Cada item desta lista, cada pedido, tem uma cor correspondente ao seu estado: azul para pedidos com estado 'Pedido', amarelo para pedidos com estado 'Cancelado', verde para pedidos aceites e verde escuro para pedidos pagos. Ao clicar num dos pedidos, o utilizador é levado para a página com mais detalhes sobre esse mesmo pedido. Os detalhes para cada tipo de bilhete são apresentados na próxima secção.

6.3.5 Bilhetes

6.3.5.1 Pedido

Esta página apresenta informações sobre o pedido de viagem que ainda não foi aceite pelo operador. Como tal, são indicados os dados do pedido, nomeadamente a data requerida para a viagem, a hora correspondente à partida e a hora estimada de chegada, com base no percurso. Também é apresentado o número de passageiros.

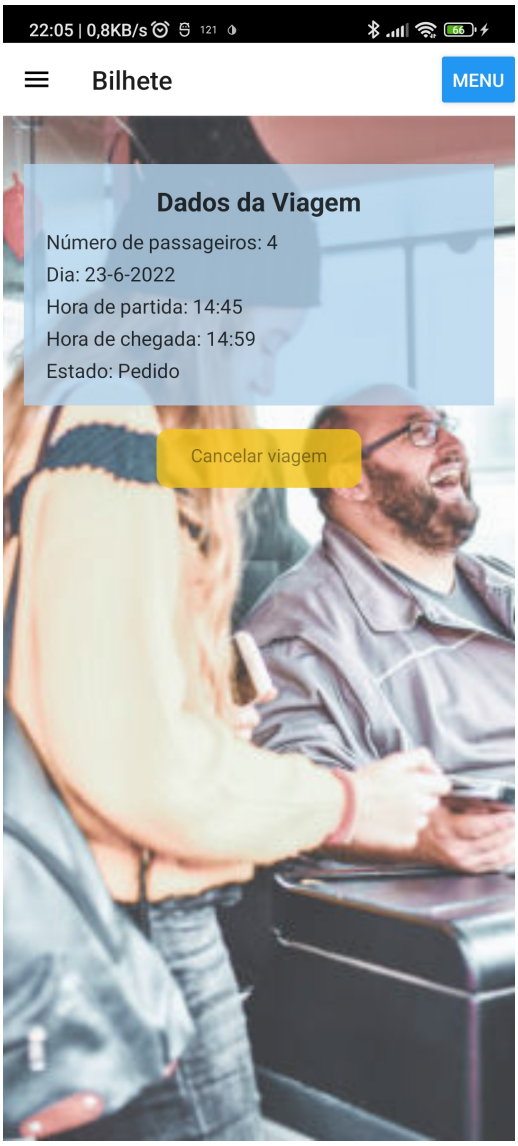


Figura 6.7: Bilhete com estado 'Pedido'

A única ação possível é a de cancelamento, que logicamente cancela o pedido. Dada a relevância desta ação, a cor do botão é de aviso, amarela. Após o bilhete ser cancelado, ao voltar para a página de listagem de pedidos, este terá o seu estado atualizado.

Em caso de sucesso no cancelamento, uma mensagem com essa indicação é apresentada indicando que na página de listagem o estado estará atualizado.

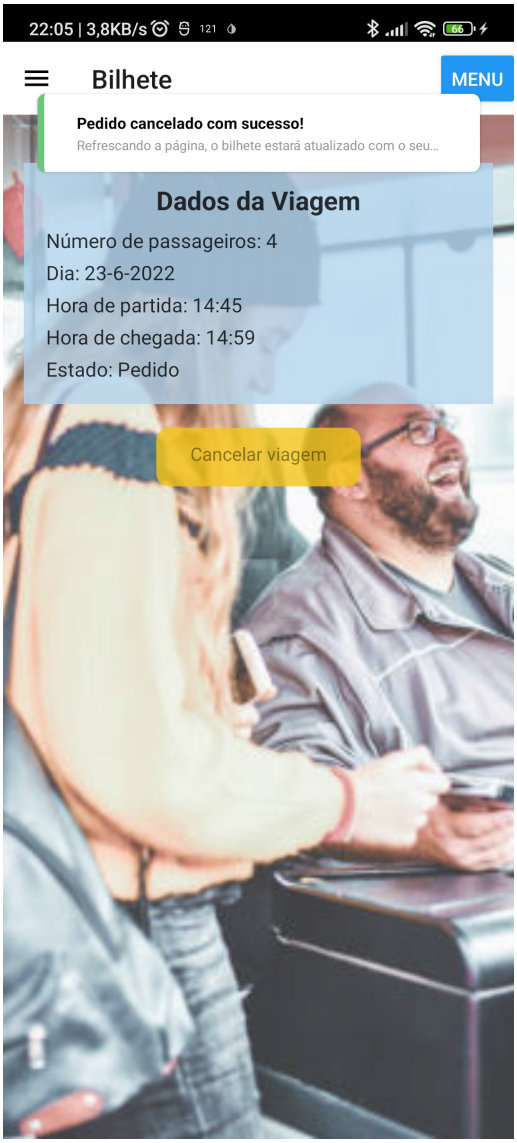


Figura 6.8: Sucesso no cancelamento de um pedido

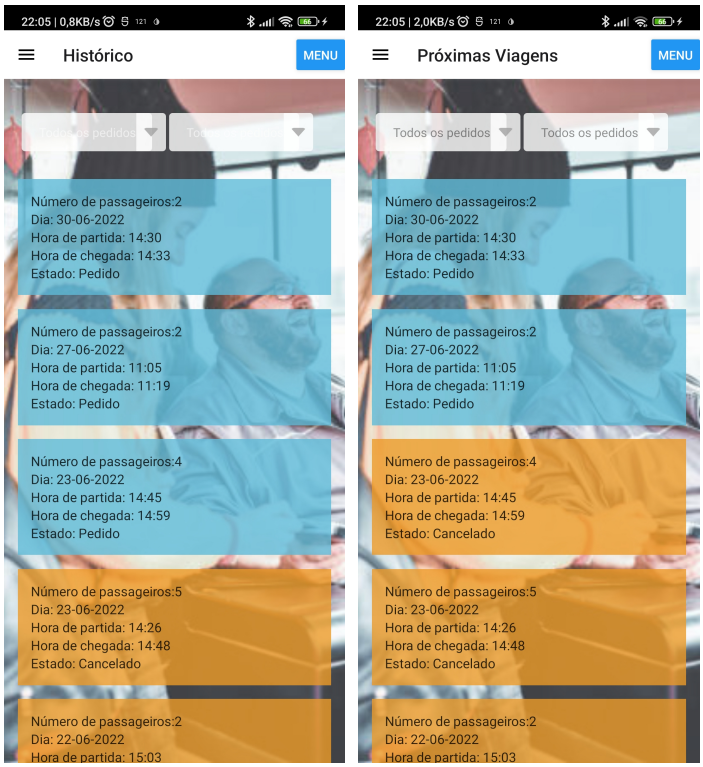


Figura 6.9: Histórico antes e depois do cancelamento do Pedido

6.3.5.2 Pedido Cancelado

Em bilhetes cancelados, são apresentados os dados da viagem como num bilhete com estado 'Pedido', porém, não existe qualquer ação disponível nesta página para além de voltar à listagem de pedidos.



Figura 6.10: Sucesso no cancelamento de um pedido

6.3.5.3 Aceite

Nesta interface, uma vez que a viagem já foi aceite pelo operador, existem duas componentes: os dados da viagem e os dados do veículo. Esta divisão ocorre para uma melhor visualização por parte do utilizador. Nos 'Dados da viagem', são apresentados, similarmente aos bilhetes com estado 'Pedido', os dados temporais da viagem (data & hora) e o número de passageiros estão disponíveis. No entanto, para além destes, o utilizador consegue ver o número de paragens previstas no percurso e o valor a pagar pela viagem.

Na secção de 'Dados do Veículo', o utilizador depara-se com informações técnicas sobre o tipo de transporte que irá realizar a viagem, sejam a marca e modelo do veículo, a sua matrícula e a capacidade do mesmo, quer de passageiros quer de passageiros com mobilidade reduzida.

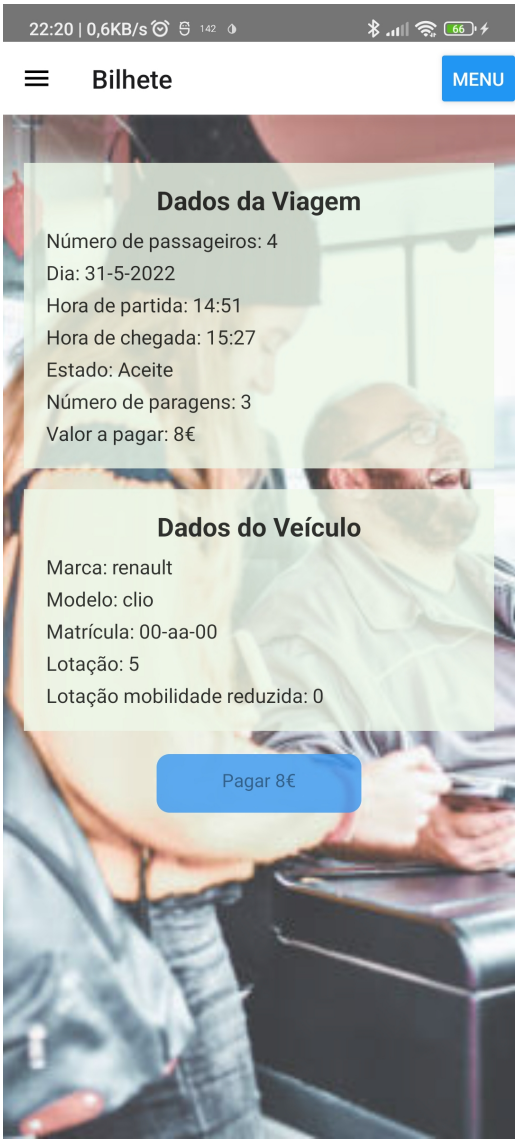


Figura 6.11: Bilhete com estado 'Aceite'

6.3.5.4 Bilhete Pago

Até chegar ao estado 'Pago', um bilhete passa pelo estado 'Pedido' e 'Aceite'. Como tal, os dados apresentados são os presentes nessas mesmas páginas (Dados da viagem + dados do veículo). Todavia, importa referir um novo aspeto: os 'Dados de Pagamento'. Nesse sentido, um novo componente surge com a data de pagamento e a quantia paga, bem como o identificador de pagamento.
A ação possível nesta página é a de geração de código-QR, para validação no veículo de transporte aquando da viagem.

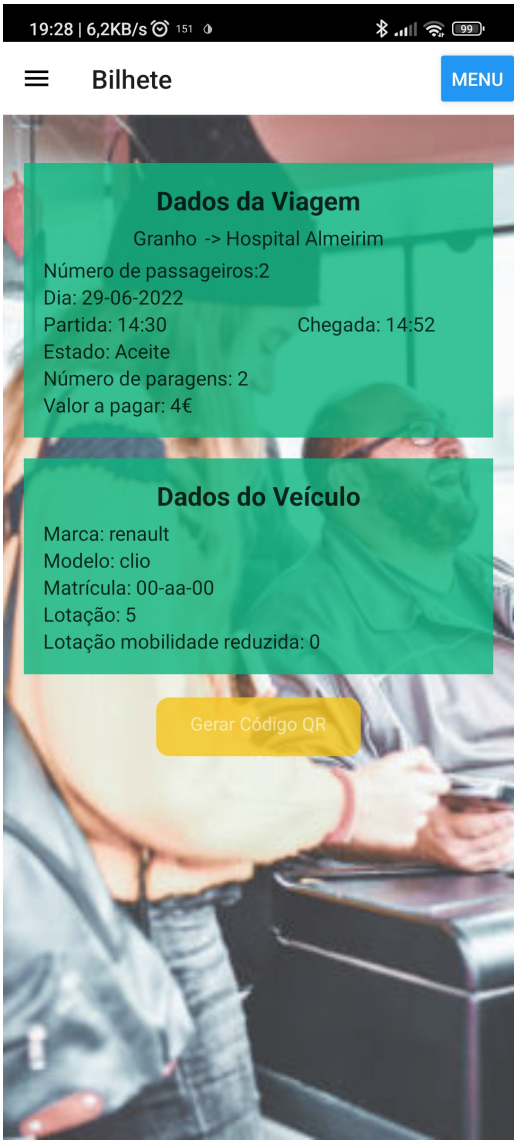


Figura 6.12: Bilhete com estado 'Pago'

6.3.5.5 Código QR

Na página de bilhete pago, pressionando o botão 'Gerar QR-Code' a aplicação gera o mesmo com base nos dados da viagem, concatenando-os numa string e mostrando o código-QR resultante ao utilizador.
Com esta informação o utilizador pode validar o bilhete ao motorista e, caso a zona onde visa tomar o transporte não tenha rede móvel estável, pode optar por guardar o código-QR gerado (com uma captura de ecrã, por exemplo), e desta forma poder viajar sem qualquer perturbação.



Figura 6.13: QR-Code do bilhete

6.4 Avaliação

De modo a percecionar a qualidade e intuitividade da aplicação móvel, a mesma foi posta à prova nas suas diversas vertentes, quer na criação de pedidos, cancelamento e pagamento dos mesmos e na geração de códigos-QR.
Oito pessoas, de diferentes grupos etários fizeram parte destas experiências voluntariamente. 3 pessoas do sexo feminino e 5 do sexo masculino, das quais 2 na faixa etária dos 35-50 e as restantes com menos de 35 anos. As duas pessoas da faixa etária mais elevada tinham menos conhecimento ou facilidade tecnológica, embora não possa ser avaliado quantitativamente. As experiências decorreram utilizando um dispositivo *Xiaomi Mi 11i*.

6.4.1 Procedimento

Os testes foram realizados nos dispositivos referidos em 6.2 e consistiram em:

- Período de habituação
- Criação de um novo pedido
- Cancelamento do pedido criado
- Pesquisa de um pedido específico
- Geração de um código-QR

Após as experiências, os participantes foram convidados a preencher um questionário com algumas perguntas sobre a sua experiência.
Inicialmente, após a explicação prévia da dissertação e sua motivação, era dada aos participantes a oportunidade de 'experimentarem' a aplicação durante cerca de 30 segundos a 1 minuto, com ajuda. Este facto prende-se com a tentativa de simulação do que seria uma experiência real, em que após o *download* de uma aplicação o utilizador tem um tempo de 'habituação' à mesma, explorando as suas funcionalidades.

6.4.1.1 Criação de um novo pedido

Após este período, a primeira tarefa consiste em criar um novo pedido com dados indicados, isto é, eram dados ao participante as paragens de origem e destino, a data da viagem, a hora de partida da mesma e ainda o número de passageiros. Com base nesta informação, o utilizador tem como objetivo criar um novo pedido com estes parâmetros. As medidas de avaliação desta tarefa são a precisão, medida de percentagem de correção dos parâmetros do pedido relativamente aos parâmetros iniciais, e ainda o tempo necessário para a conclusão da tarefa. Quanto à primeira métrica desta tarefa, o que podemos observar é uma taxa de 100% de precisão, na medida em que todos os pedidos cumpriram os requisitos prévios e todos os parâmetros foram os desejados.

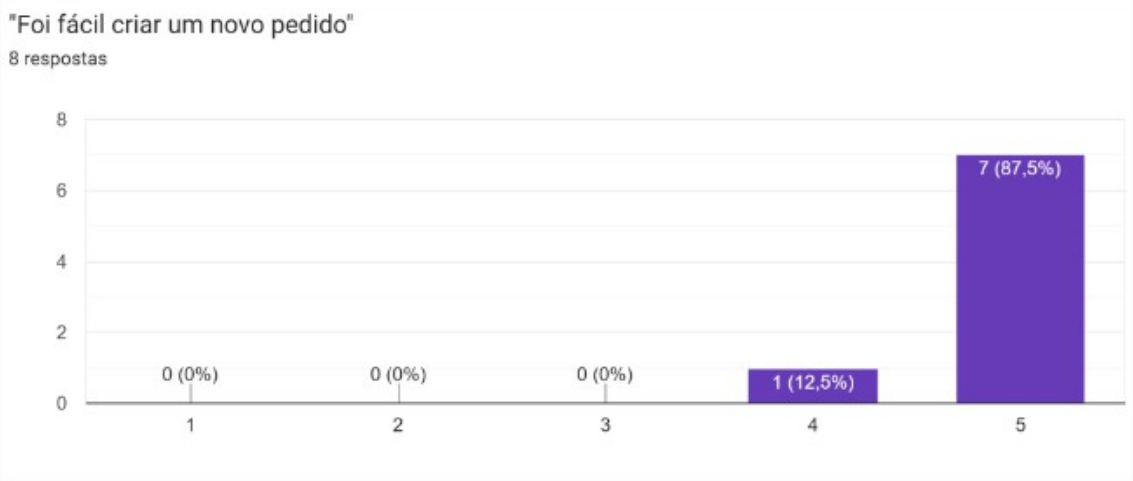


Figura 6.14: Facilidade dos utilizadores na criação de um pedido

Relativamente ao tempo, podemos observar que entre o tempo mínimo (44.604 segundos) e o máximo (80.588 segundos) existe uma diferença de 36 segundo. Parece também existir uma relação entre a faixa etária do participante e o tempo demorado, uma vez que os tempos superiores a 1 minuto se encontram em participantes com mais de 25 anos.
Como podemos ver na figura, os utilizadores concordam ou concordam plenamente com a afirmação "Foi fácil criar um novo pedido". Por este motivo a média de tempo para criar um novo pedido situa-se abaixo de 1 minuto e a precisão é de 100%.

6.4.1.2 Cancelamento do pedido criado

A segunda tarefa dos participantes era encontrar o pedido que haviam acabado de criar, referir alguns detalhes sobre o mesmo e cancelá-lo. Para isto, na maioria dos casos os utilizadores fizeram o percurso 'Menu'->'Histórico de Viagens'-> *"scroll"* na página até encontrarem o pedido-> clique no mesmo-> página 'Detalhes do Pedido'. Nesta página, os utilizadores não tiveram qualquer dificuldade em enumerar todos os pontos da viagem, bem como no cancelamento do pedido.

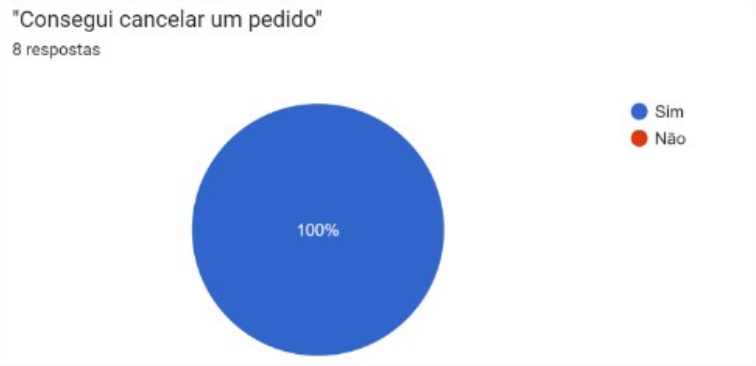


Figura 6.15: Percentagem de utilizadores com sucesso no cancelamento de um pedido

6.4.1.3 Pesquisa de um pedido específico

Os participantes eram então convidados a encontrar um pedido específico, que tivesse estado 'Pago', sendo que em cada experiência só havia 1 pedido com esse estado. Nesta experiência todos os utilizadores foram capazes de encontrar o pedido, embora nem todos fizessem uso dos filtros. Os utilizadores com mais dificuldade em encontrar o seu pedido na experiência anterior foram os mais rápidos a encontrar o pedido 'Pago' nesta tarefa, mostrando uma rápida curva de aprendizagem na aplicação.

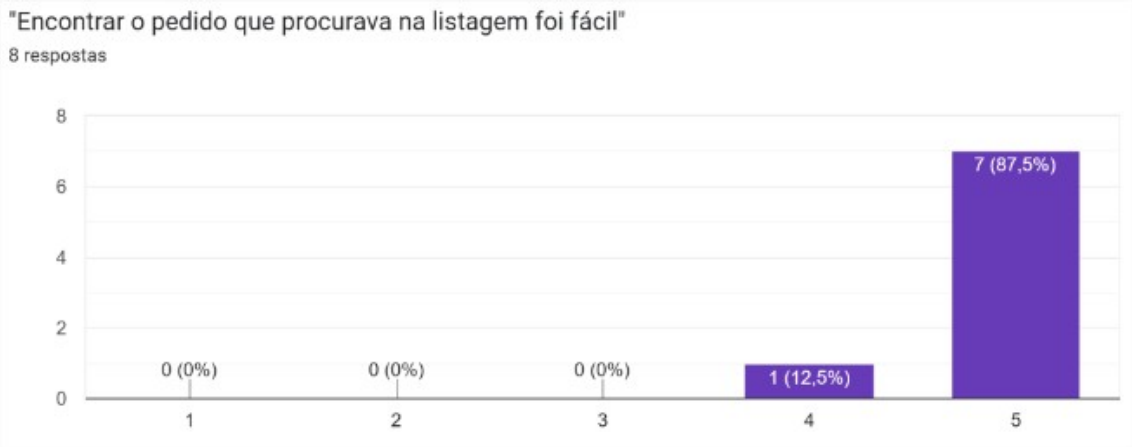


Figura 6.16: Facilidade na procura de um pedido

6.4.1.4 Geração de um código-QR

Entrando nos detalhes de um bilhete 'Pago', os utilizadores selecionaram a informação mais relevante para eles presente nessa página. Os resultados, embora um pouco dispersos, centraram-se nas informações da viagem, mais do que as informações do veículo. O mais referenciado foi o custo da viagem e a hora de partida.
De seguida, era exposta a tarefa de gerar um código-QR. Todos os participantes conseguiram gerar o código-QR relativo ao bilhete sem qualquer dificuldade a registar.

6.4.2 Questionário

Para além das perguntas referidas nos diferentes passos da secção anterior, foram apresentadas mais algumas perguntas, relacionadas com a acessibilidade e usabilidade da app. Um dos principais focos no desenvolvimento da aplicação foi a sua usabilidade. É importante, vital até, que a aplicação móvel seja fácil de usar, e que seja intuitivo o processo de fazer um pedido, pagar um bilhete e gerar o código-QR. Esta é uma das maiores necessidades da aplicação e dos meios na qual será aplicada.
Neste sentido, os resultados afiguram-se animadores.

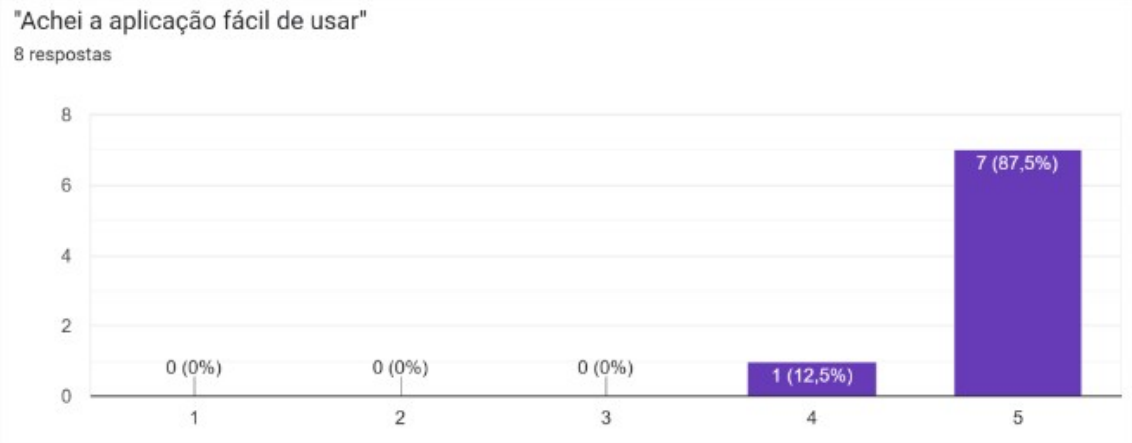


Figura 6.17: Facilidade dos utilizadores na utilização da aplicação

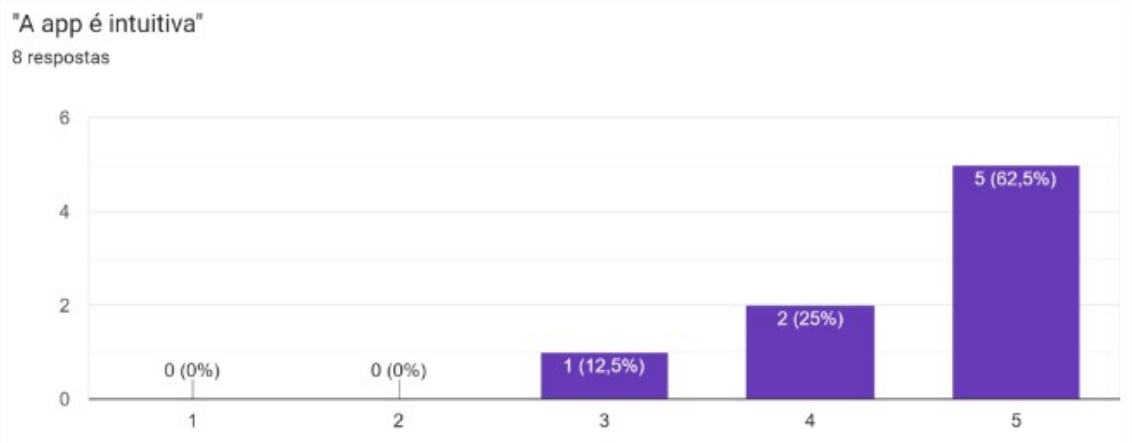


Figura 6.18: Facilidade dos utilizadores na criação de um pedido

Nestas duas questões, podemos ver que as respostas da facilidade de utilização da aplicação se encontram entre 4 e 5. Relativamente à intuitividade da aplicação, quase 60% dos utilizadores colocam o patamar máximo (5). Um ponto adicional nesta questão depara-se com a diferença entre emulador e dispositivo físico. Os utilizadores que percecionaram a aplicação menos intuitiva realizaram as experiências num emulador, o que nos leva a inferir que a experiência de utilização num dispositivo físico é significativamente melhor.

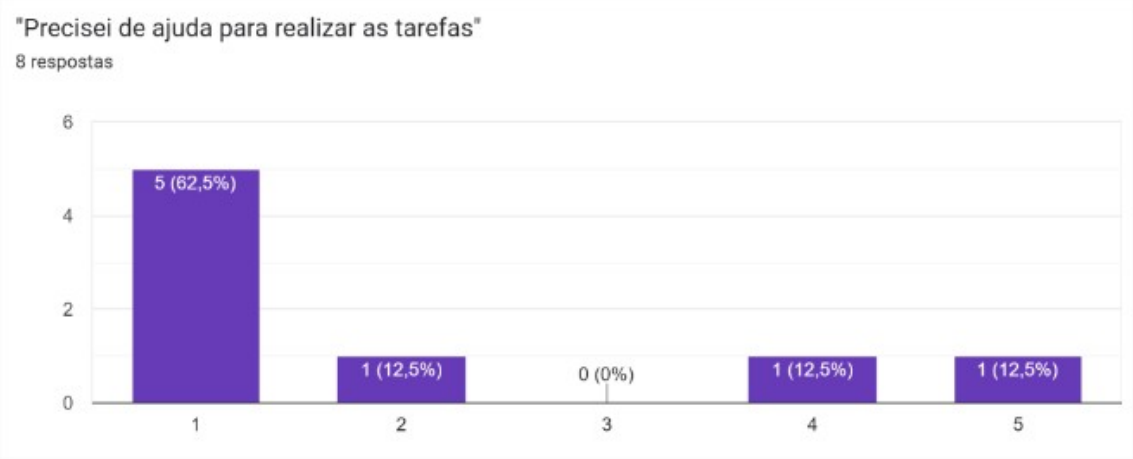


Figura 6.19: Facilidade dos utilizadores na criação de um pedido

Nesta última questão, podemos observar que a maioria dos utilizadores não precisou de ajuda para realizar as tarefas, discordando totalmente ou parcialmente da afirmação da questão.

6.4.3 Resumo

Estes resultados encontram-se na linha da ideia inicial, na medida em que demonstram que a aplicação é intuitiva e fácil de usar.
Globalmente, o que podemos concluir desta experiência é, por um lado, que a criação de um novo pedido não gera obstáculos aos utilizadores, demorando cerca de 1 minuto nessa tarefa.
Por outro lado, encontrar um pedido feito anteriormente, na listagem, quer seja para ver mais detalhes ou efetuar uma ação sobre ele (cancelamento, pagamento ou geração de código-QR), augura-se uma tarefa de relativa facilidade e com uma elevada taxa de sucesso.

Capítulo 7

Conclusões e Trabalho Futuro

Em pleno século XXI, é do conhecimento geral a emergência climática e o esgotar de tempo que estamos a percorrer numa caminhada cujo final parece mostrar-se penoso. O que também é de conhecimento geral, é que a utilização de transportes coletivos em detrimento de transportes individuais é uma das maiores armas de combate face à emissão de gases poluentes, capítulo em que os combustíveis fósseis ainda imperam. Uma das melhores formas de cativar a sociedade civil para a utilização de transportes coletivos é a flexibilização dos mesmos – isto consegue-se com TPF (DRT). A falta de modernização ou flexibilidade tecnológica nas atuais soluções TPF dificulta a massificação das mesmas. O que clama por uma aplicação que permita, ela mesma, a pesquisa, escolha e aquisição de bilhetes para este tipo de transportes – objetivo desta dissertação. Os avanços na tecnologia permitem-no agora, com APIs seguras, protocolos como OAuth2, trocas de dados com a cloud e QR-Codes e seus validadores. Desta forma, o trabalho apresentado pretendeu fazer uma revisão bibliográfica sobre estas temáticas, sobre o que é o TPF e a bilhética móvel, as atuais soluções e de que forma a sua conjunção permite o surgimento do resultado desta dissertação. Um dos desafios foi a definição do espaço de domínio a partir do qual seria desenvolvida a aplicação, bem como a seleção das tecnologias nunca esquecendo os objetivos definidos *a priori*. Com esta dissertação foi possível explorar uma plataforma *Web* desenhada para um contexto TPF, com muito potencial mas ainda sem um grande público utilizador.

A construção da aplicação foi desafiante, mas no geral os objetivos foram alcançados, com as funcionalidades previstas no desenho inicial da proposta de solução a estarem presentes nesta Prova de Conceito – Criação de um novo pedido, visualização de detalhes de um pedido, cancelamento de um pedido, pagamento do mesmo e geração de um código-QR passível de ser validado em diferentes condições (i.e. sem rede móvel por exemplo). Os resultados preliminares demonstram uma forte adesão por parte dos utilizadores à mesma aplicação, e um conjunto de sentimentos positivos perante a mesma. Com uma taxa de 100% de conclusão das tarefas, e uma elevada classificação nos pontos Acessibilidade (4.5) e Intuitividade (4.5), numa escala 0-5, podemos concluir que a aplicação móvel é um trabalho em bruto pronto para ser lapidado.

7.1 Trabalho Futuro

Ao trabalho efetuado falta a maior peça do puzzle: a implementação num sistema real. A aplicação pode ser lançada no final deste trabalho, e as suas funcionalidades correspondem às encontradas na plataforma web. Desta feita, é possível, lançando de imediato a aplicação, que os utilizadores comecem a instalar a aplicação e a fazer pedidos, sem que seja necessário qualquer novo desenvolvimento. Todavia, relativamente à componentes de login, pagamento e validação de bilhetes, há, ainda, trabalho a desenvolver.

7.1.1 Login

Neste momento, não é possível fazer *login* explícito na aplicação. De modo a poder utilizar a aplicação de forma plena, os utilizadores necessitam de fazer *login* na plataforma web, inspecionar o *token* utilizado nas chamadas à API, copiá-lo e inseri-lo na aplicação. No entanto, o protocolo Open-ID utilizado na plataforma web já foi estudado nesta dissertação, e no caso de ser este o protocolo a desenvolver, o caminho está lançado. Existe uma outra possibilidade para o Login a ser considerada e que se prende com o público-alvo desta dissertação, que se prevê com menos conhecimento tecnológico. Assim, importa facilitar o processo de *login*. Uma forma de o fazer seria utilizar processos biométricos ou de *face-ID* uma vez que pode ser complexo para algumas pessoas introduzir as credenciais de utilizador.

7.1.2 Pagamento

Neste momento, na aplicação móvel, pressionando o botão 'Pagar' no bilhete com estado 'Aceite', é simulado o pagamento e enviada a informação ao endpoint da *API*, em *'api/v1/request/requestID/pay'*, em que requestID é o identificador do bilhete. A incompletude deste ponto prende-se com o atraso no desenvolvimento de uma gateway específica para este projeto por parte da SIBS, e, por isso, o trabalho futuro passa pela implementação do pagamento dentro da aplicação e só quando este está concluido enviar a informação para a *API*.

7.1.3 Validação dos Bilhetes

Relativamente à validação dos bilhetes, atualmente são gerados códigos-QR em pedidos com estado 'Pago' para efeitos de validação destes mesmos bilhetes. Contudo, importa desenvolver uma aplicação 'Motorista', para o motorista de cada veículo de transporte de passageiros flexível (TPF) possa validar o mesmo código, para isso apenas necessitando de o ler. O trabalho a ser realizado é a criação desta aplicação, cuja única função é pedir a permissão do utilizador para usar a câmara do dispositivo e ler o código-QR, descodificando-o.
A forma de codificação da informação do lado da aplicação desta dissertação é feita de tal forma que a sua validação possa ser realizada mesmo sem acesso à rede móvel, um ponto importante para o público alvo que mais beneficiará da implementação de um sistema TPF.

7.1.4 Lançamento

O último passo do que pode ser considerado trabalho futuro é o lançamento da aplicação. Este é um passo importante para catapultar o conceito de transporte de passageiros flexível para uma nova dimensão e para a sua normalização como alternativa ao transporte próprio.

Esta dissertação termina como começou, com um apelo à mudança, para uma mobilidade sustentável e dinâmica.

Referências

[1] CCDR Alentejo. FAQs-Plataforma 'Transporte a Pedido'. Available at https://mobilidade.alentejo.pt/DuvidasFrequentes.aspx, Accessed last time in February 2022, fevereiro 2022.

[2] CCDR Alentejo. Plataforma 'Transporte a Pedido'. Available at https://mobilidade.alentejo.pt/PercursoseHorarios.aspx, Accessed last time in February 2022, fevereiro 2022.

[3] ANACOM. tem.REDE um mapa de cobertura de rede móvel em Portugal. Available at https://anacom.maps.arcgis.com/apps/Cascade/index.html?appid=ad3f71dbb09541518f436aa828feb28e, Accessed last time in June 2022, 2022.

[4] Jenny Brake, John D. Nelson, e Steve Wright. Demand responsive transport: towards the emergence of a new market segment. *Journal of Transport Geography*, 12(4):323–337, 2004. State Intervention in Contemporary Transport.

[5] Contacto. Arranjar um certificado Covid falso? Basta perder algumas horas na 'net'. Available at https://www.wort.lu/pt/luxemburgo/precisa-de-um-certificado-covid-falso-basta-algumas-horas-na-net-61fc1394de Accessed last time in June 2022, 2022.

[6] CIMRC Comunidade Intermunicipal Região de Coimbra. Sit Flexi. Available at https://www.cim-regiaodecoimbra.pt/cim-rc/projeto-sitflex/, Accessed last time in February 2022, fevereiro 2022.

[7] CIMRC Comunidade Intermunicipal Região de Coimbra. Sit Flexi PDF Pampilhosa da Serra. Available at https://www.cim-regiaodecoimbra.pt/wp-content/uploads/2021/06/AF_SIT_Flexi_Folheto_Pampilhosa_148x210mm_4.pdf, Accessed last time in February 2022, fevereiro 2022.

[8] Bonnie Eisenman. *Learning react native: Building native mobile apps with JavaScript*. "O'Reilly Media, Inc.", 2015.

[9] Felipe Mariz Coutinho, Niels van Oort, Zoi Christoforou, María J. Alonso-González, Oded Cats, Serge Hoogendoorn. Impacts of replacing a fixed public transport line by a demand responsive transport system: Case study of a rural area in amsterdam. *Research in Transportation Economics*, 83, junho 2020.

[10] Google. G-Pay. Available at https://pay.google.com/, Accessed last time in February 2022, fevereiro 2022.

[11] IMT. TPF segundo IMT. Available at https://www.imt-ip.pt/sites/IMTT/Portugues/TransportesRodoviarios/Transporte-Passageiros-Flexivel/LicenciamentodeEmpresas/Paginas/AcessoaAtividade.aspx, Accessed last time in February 2022, fevereiro 2022.

[12] QRCode npm package. QRCode npm package. Available at https://www.npmjs.com/package/react-native-qrcode-svg, Accessed last time in June 2022, 2022.

[13] npmjs inc. Node Package Manager - npm. Available at https://www.npmjs.com/, Accessed last time in June 2022, 2022.

[14] OpenID. OpenID. Available at https://openid.net/connect/, Accessed last time in June 2022, 2022.

[15] OPT. MOBUS. Available at https://www.opt.pt/mobus/, Accessed last time in February 2022, fevereiro 2022.

[16] @react-native community. @react-native-community/datetimepicker. Available at https://www.npmjs.com/package/@react-native-community/datetimepicker, Accessed last time in June 2022, 2022.

[17] @react navigation/drawer. @react-navigation. Available at https://reactnavigation.org/docs/drawer-based-navigation/, Accessed last time in July 2022, 2022.

[18] StatCounter. Mobile Operating System Market Share Worldwide. Available at https://gs.statcounter.com/os-market-share/mobile/worldwide, Accessed last time in February 2022, fevereiro 2022.

[19] TIP. Guia App Anda. Available at https://www.linhandante.com/uploadFiles/Mini-Guia_1.pdf, Accessed last time in February 2022, fevereiro 2022.

[20] Lionel Sujay Vailshery. Cross-platform mobile frameworks used by developers worldwide 2019-2021. Available at https://www.statista.com/statistics/869224/worldwide-software-developer-working-hours/, Accessed last time in May 2022, maio 2022.

[21] Visão. Cresce o mercado negro de falsos certificados de vacinação e testes negativos em Portugal. Available at https://visao.sapo.pt/atualidade/sociedade/2022-01-18-em-portugal-cresce-o-mercado-negro-de-falsos-certificados-de-vacinaca Accessed last time in June 2022, 2022.

[22] Yapsody. Yapsody. Available at https://www.yapsody.com/ticketing/features/enterprise-grade-features/qr-code-ticket-scanner/, Accessed last time in February 2022, fevereiro 2022.