INSTITUTO SUPERIOR DE ENGENHARIA DO PORTO
MESTRADO EM ENGENHARIA INFORMÁTICA




Desenvolvimento de aplicação móvel e backend de gestão de serviços no âmbito da Faculdade de Letras da Universidade do Porto

JOSUÉ LUÍS MILHINHOS BAPTISTA LAPA
Outubro de 2022

POLITÉCNICO
DO PORTO

Instituto Superior de Engenharia do Porto

DESENVOLVIMENTO DE APLICAÇÃO MÓVEL E BACKEND DE GESTÃO DE SERVIÇOS NO ÂMBITO DA FACULDADE DE LETRAS DA UNIVERSIDADE DO PORTO

Josué Luís Lapa

Dissertação para obtenção do Grau de Mestre em Engenharia Informática, Área de Especialização em Sistemas Gráficos e Multimédia

Orientador: João Paulo Pereira

Júri:

Presidente:

Vogais:

Porto, fevereiro 2022

À raposa de milhões...

Resumo

Este documento descreve, detalhadamente, o processo de pesquisa, desenho e implementação de um sistema de *software* de gestão de ferramentas e serviços disponibilizados pela Faculdade de Letras da Universidade do Porto.

O núcleo do projeto está diretamente relacionado com a necessidade de facultar funcionalidades de acessibilidade a utilizadores com baixa acuidade visual e invisuais, facilitando-lhes o acesso aos serviços da instituição, mais concretamente os de *vending* e carregamento de cartões.

O sistema é constituído por um *backend* e uma aplicação móvel multiplataforma, uma interface para a utilização dos serviços supramencionados, com vista a reduzir a necessidade de utilização dos métodos convencionais de interação com as máquinas de *vending*, moedeiros ou caixas multibanco, que obrigam ao contacto físico com os equipamentos presentes nas instalações da faculdade, que, dado o contexto pandémico vivido, podem ainda tornar-se fontes de contágio.

Desta forma, os meios acima descritos encontram-se concentrados numa só plataforma, desenhada com o objetivo de tornar o seu manuseamento mais intuitivo e abrangente para a comunidade académica.

Palavras-chave: aplicação, *vending*, invisuais, .NET, C#, Flutter

Abstract

This document describes, in detail, the process of research, design and implementation of a management software system of services provided by the Faculdade de Letras da Universidade do Porto.

The core of the project is directly related to the development of accessibility features for blind users and with low visual acuity, to allow access to the institution's services, specifically vending and card charging machines.

The system consists of a backend and a mobile application, an interface for the use of the aforementioned services, in order to reduce the need to use conventional methods of interaction with the vending machines, coin machines or ATMs, which require physical contact with the equipments present in the facilities, which, given the pandemic context, can still be considered sources of contamination.

The tools described are concentrated in a single platform, designed to make its handling more intuitive and comprehensive for the academic community.

Keywords: application, *vending*, blind, .NET, C#, Flutter

Agradecimentos

Um agradecimento especial ao responsável pelo Serviço de Informática da FLUP, Vítor Pereira e ao Eng.º João Paulo Pereira por sempre me terem apoiado, com o maior profissionalismo, de forma a atingir os meus objetivos e corrigir os meus erros.

Aos meus pais e avós, os meus segundos pais, que me apoiaram incondicionalmente, incentivando-me durante todo o meu percurso de vida e académico: um grande obrigado!

Por fim***, Fico-te*** eternamente agradecido, por ***Tudo***.

Índice

Lista de Figuras

Lista de Tabelas

Acrónimos e Símbolos

Lista de Acrónimos

AD	*Active Directory*
AHP	Analytic Hierarchy Process
API	*Application Programming Interface*
CNP	*Card-Not-Present*
CSS	*Cascading Style Sheets*
DDD	*Domain-Driven Design*
FLUP	Faculdade de Letras da Universidade do Porto
HOQ	*House of Quality*
HTML	*Hyper Text Markup Language*
HTTPS	*Hyper Text Transfer Protocol Secure*
IC	Índice de Consistência
IR	Índice Aleatório
JSON	*JavaScript Object Notation*
LDAP	*Lightweight Directory Access Protocol*
LINQ	*Language-Integrated Query*
MVC	*Model-View-Controller*
OEM	*Original Equipment Manufacturer*
QFD	*Quality Function Deployment*
QEF	*Quantitative Evaluation Framework*
QR	*Quick Response*
RC	Razão de Consistência
SI	Serviço de Informática
SQL	*Structured Query Language*

SWOT	*Strengths, Weaknesses, Opportunities, Threats*
UI	*User Interface*
UX	*User Experience*
VB	Visual Basic

Glossário

Backend	Componente associado à lógica de um software, impercetível para o utilizador
Back Office	Componente do *Frontend*, relacionado com as funcionalidades administrativas e de gestão de um *software*
Container	Unidade de software de código isolado, executável
Datacenter	Infraestrutura de armazenamento e processamento de dados
Debug	Depuração de erros de linguagens de código de *software*
Development Kit	Conjunto de ferramentas de desenvolvimento de *software*
Disaster Recovery	Recuperação de infraestruturas tecnológicas
Framework	Conjunto de ferramentas e bibliotecas que unificam tecnologias
Frontend	Componente associado à interface com o utilizador de um *software*
Front Office	Componente do *Frontend*, relacionado com as funcionalidades disponibilizadas aos consumidores do *software*
Host	Anfitrião de acesso a serviços
Hot Reload	Mecanismo que permite edição de código fonte em execução
Open-Source	Referente a software, cujo código está disponível publicamente
SARS-CoV-2	O vírus causador da doença coronavírus 19 (COVID-19)
Thread	Tarefa de execução de um determinado *software*
Vending	Atividade relacionada com a venda de produtos
WebHooks	Mecanismo de envio de mensagens automáticas de aplicações
Widget	Atalhos de acesso a funcionalidades do sistema
Wrapper	Código ou aplicação capaz de ser executado sem alterações da aplicação base

1 Introdução

1.1 Enquadramento

A revolução tecnológica provou (e prova) o quão benéfica a ciência da computação é para a humanidade.

O Serviço de Informática (SI) gere um conjunto de serviços integrantes da Faculdade de Letras da Universidade do Porto (FLUP), nomeadamente, os de controlo do parque de estacionamento, acesso a espaços, inscrições em eventos, os moedeiros, impressão de fotocópias, entre outros. Os utilizadores de rede têm contas com um saldo associado. O saldo pode ser creditado utilizando os moedeiros das instalações ou através dos métodos de pagamento disponíveis na plataforma *online* do SI, nomeadamente MBWay e geração de referências multibanco.

Os serviços descritos são suportados por uma infraestrutura de virtualização constituída por dois *datacenters*, com três *hosts* cada. Um é responsável pelos servidores de produção e outro consitui a infraestrutura de *disaster recovery*. Os três *hosts* alojam os controladores de domínio, responsável pela autenticação dos utilizadores da FLUP, o serviço de correio eletrónico em sistema de alta disponibilidade, caracterizado por múltiplos servidores que atendem a diferentes clientes de forma distribuída, de bases de dados, entre outros.

Em conjunto com o Serviço de Informática da FLUP, foi planeado o desenvolvimento de uma aplicação dotada de um conjunto de ferramentas úteis para a utilização de serviços geridos pelo SI. Os principais utilizadores beneficiários desta aplicação são a comunidade académica da instituição, inclusivamente, utilizadores com dificuldades visuais e invisuais, que evidenciaram dificuldades na utilização dos equipamentos com ecrãs táteis. Isto deve-se, principalmente, à falta de suporte de integração com ferramentas de acessibilidade como sintetizadores de voz, capazes de reproduzir os elementos de interface do utilizador (UI).

Aquando da escrita deste documento, a pandemia causada pelo vírus SARS-CoV-2 marca ainda presença mundial e diária. Notadas as evidências do impacto de uma pandemia no século XXI, a aplicação promoveria transações sem contacto e redução da circulação do dinheiro, dado que

o setor da educação foi um dos muitos afetados pelo vírus, dada a grande afluência de alunos nas escolas, faculdades e politécnicos.

1.2 Descrição do Problema

Estima-se que 80% da população portuguesa consome café (Almeida, 2015). Os equipamentos de *vending* são, atualmente, uma regular conveniência para muitos. Este tipo de equipamentos não oferece, por norma, ferramentas para auxiliar pessoas invisuais e com deficiências visuais na seleção e compra de produtos. Da mesma forma, estes indivíduos manifestam dificuldades durante a interação com outro tipo de máquinas que exigem contacto físico, como é o caso dos moedeiros utilizados para carregar os cartões de estudante da FLUP. Por estas razões, é urgente disponibilizar mecanismos de acessibilidade de modo que estes utilizadores usufruam das funcionalidades dos serviços, com menos inconvenientes.

A maioria dos equipamentos disponibilizados pela instituição não fornece a possibilidade de efetuar transações sem o uso de dinheiro, o que leva, muitas vezes, à desistência da compra por falta de opções mais convenientes.

A monitorização das transações feitas nas máquinas é feita através de plataformas *online*, o que faz com que o consumidor não tenha uma boa perceção dos gastos efetuados, visto não ser uma forma rápida e conveniente de acesso a estas informações.

Este conjunto de problemas levou à necessidade de repensar qual a melhor estratégia para embutir os serviços atualmente disponíveis, num *software* que vá ao encontro das necessidades dos utilizadores.

1.3 Objetivos

O projeto engloba um conjunto de objetivos que tem em vista resolver os problemas destacados no subcapítulo anterior. No entanto, o desenvolvimento do mesmo constitui uma mais-valia para o tratamento de futuras necessidades do sistema informático, não diretamente relacionadas com o projeto.

O primeiro objetivo consiste em permitir à comunidade académica e/ou utilizadores externos usufruírem de uma aplicação móvel como meio de comunicação com as máquinas, reduzindo o contacto físico com as mesmas. Do mesmo modo, pretende-se auxiliar os invisuais, através da utilização de ferramentas de acessibilidade na aplicação. Procura-se que a plataforma seja acessível a todos os utilizadores dos serviços da FLUP, independentemente das suas limitações, focando a inclusão social, no sentido que todos deverão ser igualmente beneficiados.

O desenvolvimento do *backend* pretende facultar novas formas de pagamento que não serão apenas utilizadas no contexto de *vending,* mas também em outras transações no âmbito da faculdade, nomeadamente no carregamento de cartões, inscrições em eventos, propinas,

candidaturas, entre outros serviços. Estes mecanismos promoverão a utilização de plataformas digitais, reduzindo a necessidade de utilização de moedeiros ou caixas multibanco.

Os utilizadores deverão poder consultar os movimentos resultantes das compras realizadas nas máquinas e outras transações ligadas ao cartão e à conta da faculdade.

A aplicação e o respetivo *backend* deverão ter em conta futuros desenvolvimentos, nomeadamente a integração de um sistema de agendamento de refeições dos bares da faculdade e controlo de acessos aos espaços da FLUP.

1.4 Metodologia

Um dos aspetos mais importantes da metodologia de trabalho deste projeto é o contacto com o *product owner* de forma a entender o contexto tecnológico do SI da instituição, atendendo aos atuais problemas que se pretendem solucionar. Opta-se, portanto, por um contacto frequente, de forma que haja um *feedback* iterativo do desenvolvimento do projeto, uma vez que os requisitos impostos poderão necessitar de adaptações ou atualizações, consoante as necessidades.

É crucial entender o modo de funcionamento e comunicação das máquinas de *vending* no espaço, a infraestrutura de *datacenters*, bem como as necessidades e problemas com que se deparam os utilizadores dos serviços. Após esta recolha de dados, é esquematizado todo o sistema tecnológico que o projeto requer.

Inclui-se um processo de pesquisa de aplicações/projetos semelhantes que permite avaliar os pontos fortes e fracos de cada um, de modo a selecionar os métodos mais eficazes e eficientes a serem aplicados na estruturação do projeto. Da mesma forma, será efetuada uma análise e interpretação do *feedback* dos utilizadores dessas aplicações.

Por fim, é feita uma seleção de possíveis tecnologias a serem aplicadas no desenvolvimento da solução.

1.5 Contributos

Após um estudo abrangente ao sistema informático gerido pelo SI, faz sentido optar por desenvolver um ambiente de servidor que seja compatível com a infraestrutura tecnológica atual da instituição (que faz uso de um ecossistema Microsoft), de forma a preservar o fluxo de trabalho dos membros do SI, que estão familiarizados com *software* do mesmo tipo.

As plataformas desenvolvidas deverão servir de base para integração de futuros desenvolvimentos ou serviços já existentes, tal como a integração de um módulo de gestão de refeições dos bares da faculdade ou de controlo ou gestão de acessos a espaços da FLUP.

Dentro do ambiente da faculdade, o *software* descrito neste documento terá vantagens para todos os utilizadores com contas registadas no sistema informático da instituição, nomeadamente alunos, docentes, pessoal administrativo, gestores, convidados, assim como indivíduos externos que necessitem de utilizar os serviços disponibilizados via *app*.

Uma das ideologias deste desenvolvimento é alargar a área de utilização da aplicação, visto que, ao serem expandidas as suas funcionalidades para outras instituições e, futuramente, para outras zonas do país, o *software* pode alcançar um público mais alargado e diversificado, fora do contexto da FLUP. Se tal se concretizar, constituirá também uma ferramenta mais capaz no controlo de futuros surtos de infeções transmissíveis por contacto com objetos. Ao atingir um público mais abrangente, será possível influenciar outros setores, visto que as máquinas de *vending* desta geração (com acesso à rede), estão distribuídas não só em escolas e faculdades, mas também em hospitais, escritórios, instituições públicas e privadas, entre outros locais. Paralelamente, os benefícios de acessibilidade para utilizadores invisuais e com dificuldades visuais seriam transportados para um público mais abrangente.

1.6 Planeamento de trabalho

O planeamento das diferentes tarefas foi descrito através do diagrama de Gantt, representado na Figura 1, onde são explicitados os períodos, ao longo dos meses, dedicados a cada fase de desenvolvimento do projeto.

Figura 1 - Diagrama de Gantt do planeamento do projeto

1.7 Estrutura do Documento

Este documento é constituído pelos seguintes capítulos:

- **Introdução** – descrição de informações de relevo para o entendimento do projeto, o problema em questão, objetivos e metodologias gerais para a conceção do mesmo
- **Contexto e Estado da Arte** – exposição e análise de projetos e/ou produtos com características semelhantes ao descrito neste documento
- **Análise de Requisitos** – identificação e detalhe dos requisitos do cliente
- **Análise de Valor** – análise dos fatores que definem o valor do projeto
- **Desenho da Solução** – delineação da arquitetura do sistema
- **Implementação da Solução** – descrição dos passos de desenvolvimento do projeto
- **Avaliação da Solução** – avaliação do sucesso do produto resultante do projeto
- **Conclusões** – descrição de resultados obtidos

2 Estado da Arte

No presente capítulo é descrito o contexto do problema do projeto a ser desenvolvido, e é feito um estudo de produtos semelhantes ao descrito neste documento, que existem atualmente no mercado, bem como técnicas e metodologias que poderão ser aplicadas no desenvolvimento do mesmo.

2.1 Contexto do Problema

O problema é composto por uma série de adversidades que a comunidade académica e membros externos à faculdade enfrentam, decorrentes da utilização dos serviços descritos. Destaca-se a falta de ferramentas de acessibilidade na utilização dos equipamentos das instalações e a descentralização da informação gerida pelos diferentes serviços disponíveis na FLUP. Por outro lado, existe ainda uma forte preocupação com as possíveis fontes de contágio devido à pandemia, especialmente no contacto com as máquinas, devido à necessidade de manuseamento do dinheiro.

O produto pode também beneficiar as empresas de *vending* que, para além de conseguirem gerir o dinheiro de forma direta, podem comercializar máquinas sem mecanismos de receção e troco de notas e moedas, reduzindo a probabilidade de assaltos e custo de equipamentos.

Algumas destas adversidades foram diretamente comunicadas ao Serviço de Informática da FLUP, por pessoas com deficiências visuais.

2.2 Produtos/Projetos relacionados

Neste subcapítulo, será efetuado o estudo de aplicações relacionadas com o mercado de *vending*, visto que as restantes funcionalidades requeridas pelo cliente são específicas da instituição, não existindo produtos que possam ser alvo de uma comparação direta.

Adicionalmente, serão analisadas e comparadas as tecnologias que atualmente são utilizadas no mercado, que poderão ser integradas nos sistemas geridos pelo SI.

2.2.1 BuyOn

A empresa BuyOn oferece uma solução em forma de aplicação móvel, com funcionalidades de comunicação com máquinas de *vending*, integração de pagamentos e funções de dispensa de produtos.

Para utilização da app, o utilizador deve registar-se, criando uma conta associada ao serviço. O processo de venda baseia-se no carregamento da conta de utilizador com saldo, através de pagamento por cartão de crédito ou MBWay. Seguidamente, o utilizador consegue visualizar, num mapa, as máquinas de *vending* compatíveis com o sistema. Uma vez em frente à máquina desejada, o utilizador deve fazer *scan* do código QR apresentado em forma de autocolante na máquina. Pode, por fim, selecionar um dos produtos disponíveis, sendo o valor do mesmo descontado do saldo do utilizador (BuyOn, s.d.).

Na Figura 2, é possível visualizar a representação de produtos e máquinas, bem como a seleção do método de pagamento na interface gráfica da aplicação.

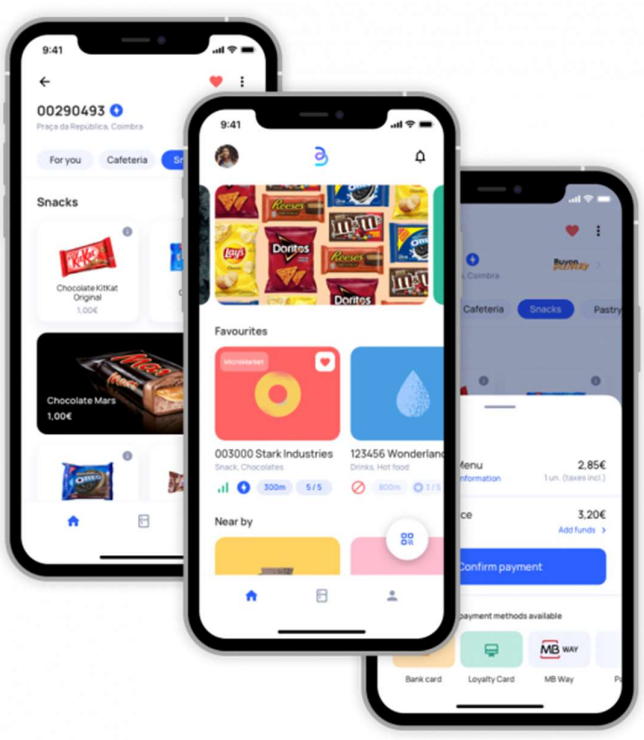

Figura 2 - Diferentes ecrãs da interface gráfica da aplicação BuyOn (BuyOn, s.d.)

Após uma análise do funcionamento desta solução, destacaram-se as seguintes vantagens e desvantagens, sendo alguns pontos baseados no *feedback* dos utilizadores:

Vantagens:

- Facilidade de identificação de uma máquina, via código QR
- Diferentes métodos de pagamento
- Possibilidade de visualizar, no mapa, máquinas disponíveis
- Registo através de associação de contas de redes sociais
- Listas de favoritos e recomendações

Desvantagens:

- Falta de detalhe no histórico de transações, nomeadamente data e hora
- Instabilidades no processo de pagamento e dispensa de produtos
- Obrigatoriedade de acesso à localização (GPS) para o seu funcionamento
- Desfasamento entre as máquinas disponíveis na app e as máquinas identificadas com código QR
- Impossibilidade de submeter *feedback*, via *app*

É de salientar que, nos testes executados, não foi possível obter uma listagem de máquinas no mapa. Este problema poderá estar relacionado com as limitações da tecnologia GPS em locais com pouca cobertura ou com outro tipo de problema derivado da natureza da aplicação.

2.2.2 Coffee APPeal

A Coffee APPeal é uma aplicação que permite interagir com máquinas de café de diferentes marcas. Esta aplicação não permite a dedução do valor de produtos, mas é dotada de funcionalidades relevantes no que toca à seleção e personalização de bebidas.

É adotado o método de *scan* de um código QR apresentado no ecrã do equipamento e não é necessário introduzir dados pessoais, ou seja, não integra contas de utilizador. A comunicação com a máquina dispensa o acesso à Internet, uma vez que utiliza o protocolo *Bluetooth*. A aplicação permite a personalização do grau de intensidade de certas bebidas (Newis, s.d.). O processo descrito é ilustrado através da Figura 3.

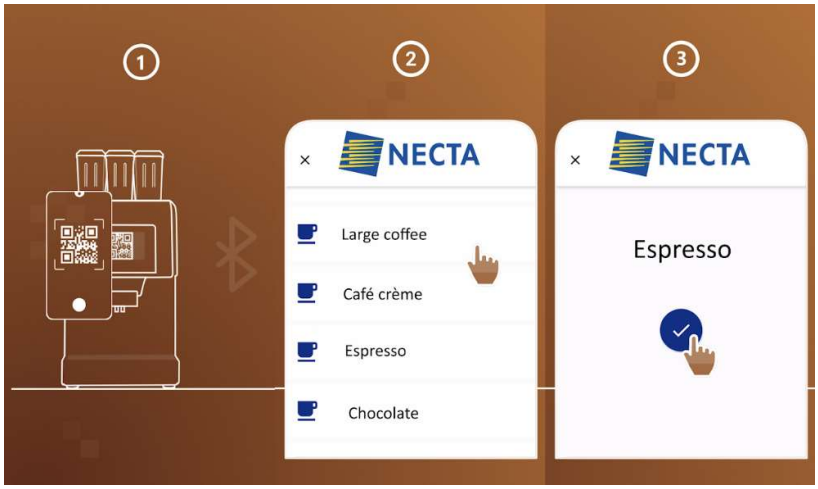

Figura 3 - Processo de dispensa de bebidas da aplicação Coffee APPeal (EVOCA S.p.A., 2021)

Tendo em conta a análise efetuada à aplicação, foram destacadas as seguintes vantagens e desvantagens:

Vantagens

- Facilidade de utilização
- Identificação do código QR através do ecrã da máquina
- Personalização de bebidas

Desvantagens

- Inexistência de histórico de bebidas ou lista de favoritos
- Não integra pagamentos
- Limitações de velocidade e segurança do protocolo *Bluetooth*

A maioria das funcionalidades mencionadas não foram testadas, dado que não foi possível obter máquinas para teste. No entanto, as informações obtidas foram baseadas em descritivos de funcionamento da aplicação, vídeos explicativos e demonstrativos do processo de seleção e dispensa de bebidas (Lavazza, 2021).

2.2.3 Button Barista

A aplicação Button Barista suporta uma série de máquinas de *vending* e oferece aos utilizadores uma vasta lista de personalizações de bebidas, nomeadamente a intensidade, quantidade, dosagem de açúcar, opção de ser ou não descafeinada, incluir leite, entre outras opções.

A aplicação guarda as opções favoritas dos utilizadores para futuros pedidos e permite visualizar um conjunto de estatísticas. Consoante o comerciante dos produtos, a aplicação pode oferecer

promoções baseadas num sistema de pontos que são acumulados de acordo com o número de pedidos (Button Barista, 2017). É possível visualizar os ecrãs e as funcionalidades descritas através da Figura 4.

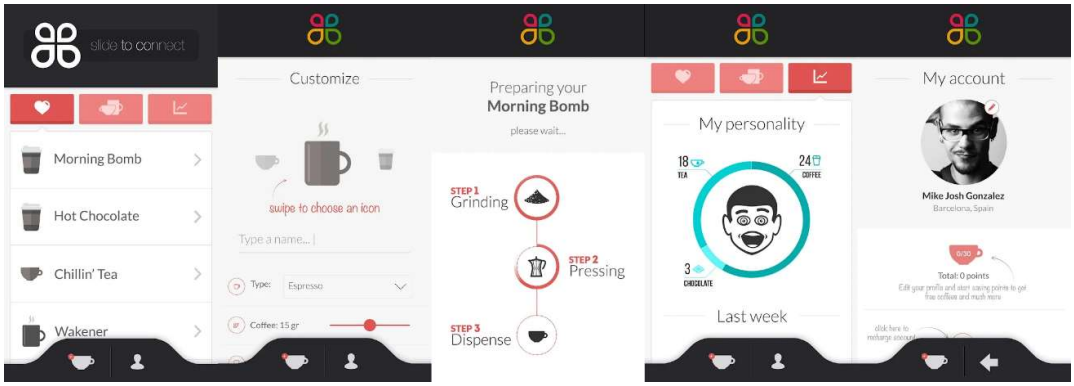

Figura 4 - Diferentes ecrãs da interface gráfica da aplicação Button Barista (Button Barista, 2017)

Após a análise de funcionamento da aplicação através de informações acerca do funcionamento da mesma, bem como de vídeos explicativos e demonstrativos do processo (Design, 2020), destacam-se as seguintes vantagens e desvantagens:

Vantagens

- Identificação da máquina intuitiva
- Elevado nível de personalização
- Listas de favoritos
- Interface clara e simples de navegar
- Sistema de promoções à base de pontos
- Acesso a estatísticas
- Ferramentas de suporte ao cliente

Desvantagens

- Limitações de velocidade e segurança do protocolo *Bluetooth*
- Pagamentos não integrados diretamente na *app*
- Possibilidade de conflito de ligação com outras máquinas nas proximidades

2.2.4 Pay4Vend

A aplicação Pay4Vend funciona como uma ponte para o pagamento entre o utilizador e a máquina de *vending*, sendo que a seleção do artigo exige contacto físico com a mesma. O processo exige um registo que será utilizado para associar e carregar saldo, e cujo pagamento pode ser efetuado através de dinheiro, nas máquinas de *vending*, via PayPal ou cartão de

crédito (Coges, s.d.). Não é especificado o protocolo utilizado para comunicação com as máquinas. É possível visualizar os ecrãs e as funcionalidades descritas através da Figura 5.

Figura 5 - Diferentes ecrãs da interface da aplicação Pay4Vend (Coges Mobile Solutions srl, 25)

Após a análise de funcionamento da aplicação através de informações explicativas do sistema, destacam-se as seguintes vantagens e desvantagens:

Vantagens

- Variedade de métodos de pagamento
- Conexão intuitiva com as máquinas
- Ferramentas de suporte ao cliente

Desvantagens

- Instabilidade do sistema
- Falta de personalização
- Inexistência de ferramentas de visualização e seleção artigos

Foi analisado o funcionamento da aplicação, assim como o *feedback* dos utilizadores que, embora seja baseado na cotação (visto não existirem evidências de crítica em forma de comentário) indica ser maioritariamente negativo, pressupondo-se estar relacionado com a instabilidade do sistema descrito.

2.2.5 Comparação de Soluções

Uma vez identificadas as soluções, seus detalhes, vantagens e desvantagens, é feita a comparação entre as mesmas, de acordo com a utilidade que estas oferecem aos utilizadores.

De forma a estabelecer comparações entre soluções, foram concebidos os seguintes critérios, relevantes ao contexto de *vending*:

- **Diversidade de métodos de pagamento** – a solução oferece a possibilidade de efetuar pagamentos através de cartão, dinheiro e outras formas de pagamento
- **Personalização de bebidas** – a solução permite personalizar a quantidade e intensidade de bebidas, a quantidade de açúcar, e guardar as preferências do utilizador
- **Pesquisa de máquinas de *vending*** – a solução permite visualizar as máquinas de *vending* disponíveis, em forma de lista e de localização no mapa
- **Seleção e dispensa de produtos** – a solução disponibiliza ao utilizador uma lista de produtos organizados por famílias e tipos, que podem ser dispensados através da aplicação
- **Estatísticas do utilizador** – a solução faculta múltiplas estatísticas acerca das compras que o utilizador efetua na aplicação e suas preferências, assim como o seu histórico
- **Promoções e/ou ofertas** – a solução permite aos utilizadores usufruírem de cupões, sistemas de pontos ou outro tipo de promoções
- **Segurança** – a solução utiliza mecanismos seguros de comunicação com as máquinas de *vending*, assim como de pagamentos e gestão de utilizadores
- **Aspetos visuais** – a solução apresenta elementos UI compreensíveis, simples e estéticos
- **Diversidade de produtos** - a solução consegue comunicar com máquinas de *vending* de bebidas quentes, frias, produtos de pastelaria, confeitaria, entre outros

Na Tabela 1, são apresentadas as avaliações de critérios das soluções descritas anteriormente, com pesos de 0 a 5, em que 0 significa que a solução não dispõe de qualquer tipo funcionalidade descrita e 5 que a solução oferece todas as funcionalidades descritas.

Tabela 1 - Comparação de funcionalidades de aplicações de *vending*

	BuyOn	Coffee APPeal	Button Barista	Pay4Vend
Diversidade de métodos de pagamento	4	0	1	5
Personalização de bebidas	0	0	5	0
Pesquisa de máquinas de *vending*	4	1	2	4
Seleção e dispensa de produtos	5	3	3	0
Estatísticas do utilizador	0	0	5	1
Promoções e/ou ofertas	0	0	5	0
Segurança	4	1	4	4
Aspetos visuais	4	3	4	2
Diversidade de produtos	5	2	2	5

2.3 Acessibilidade

Em 2019, foi realizado um estudo avaliativo da acessibilidade das máquinas de *vending* automáticas nos Estados Unidos da América, onde se estimou existirem cerca de 4,6 milhões de unidades (Caporusso, et al., 2019).

O protocolo de teste da acessibilidade das máquinas consistiu na ação de dispensa de um artigo, através de dois cenários: o primeiro, em que o utilizador estava vendado, de forma a simular a experiência de um invisual; o segundo, em que o utilizador não tinha qualquer bloqueio sensorial. Os resultados da experiência (Figura 6) mostram que na aplicação desta metodologia de testes a dois tipos de máquinas (A e B), os utilizadores vendados demoraram consideravelmente mais tempo a concluir a ação de dispensa do produto, que incluiu as etapas de seleção, pagamento (por cartão e em dinheiro) e recolha do artigo (Caporusso, et al., 2019).

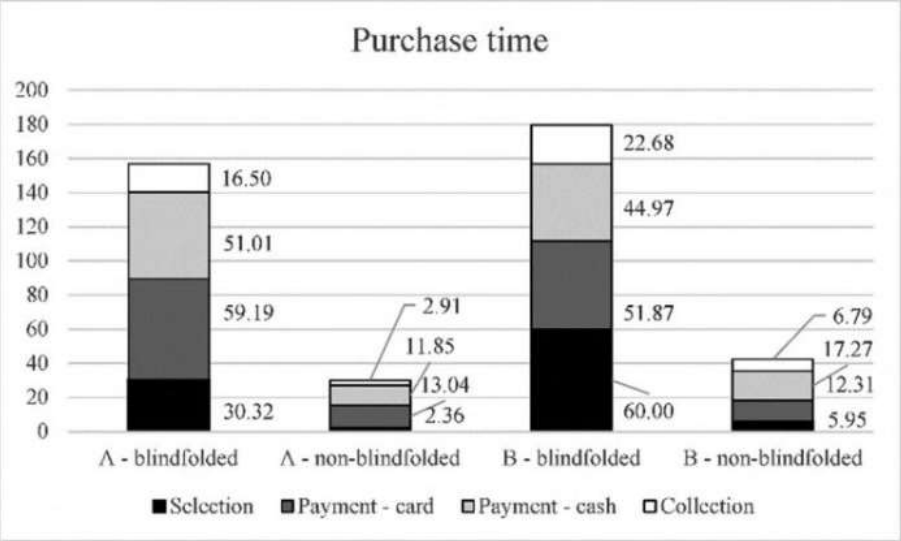

Figura 6 – Resultados da experiência de teste de acessibilidade a máquinas de *vending* (Caporusso, et al., 2019)

A pesquisa permitiu concluir que, apesar da *Americans with Disabilities Act* (ADA) - uma lei que proíbe a discriminação de pessoas com deficiência - definir critérios que permitem o acesso das máquinas a utilizadores com mobilidade reduzida, não existem tais que se apliquem a invisuais (Caporusso, et al., 2019).

Adicionalmente, são relatados aspetos negativos que criam obstáculos à acessibilidade dos equipamentos, como más práticas de *design* e falta de *feedback* não visual (Caporusso, et al., 2019).

2.3.1 Acessibilidade em aplicações móveis

Foram estudados os métodos utilizados em aplicações móveis para colmatar as dificuldades referidas anteriormente.

As aplicações VoiceOver e TalkBack, integradas no sistema iOS e Android, respetivamente, são ferramentas que permitem aos utilizadores invisuais, através de um sintetizador de voz, reproduzir o conteúdo que está a ser apresentado no ecrã do dispositivo (Zafar, 2017).

De modo que os utilizadores consigam utilizar convenientemente estas ferramentas, as aplicações devem estar devidamente preparadas. Neste sentido, devem-se seguir um conjunto de boas práticas, durante o desenvolvimento, nomeadamente a identificação clara dos elementos da UI, como imagens e botões, através de *labels* que os descrevem, com exatidão, e utilização de linguagem simples e compreensível para descrever o que está no ecrã (Zafar, 2017).

Na Figura 7, destacam-se dois exemplos de interfaces gráficas: a da esquerda apresenta elementos visuais sem guias de orientação que descrevam o que a aplicação espera do utilizador. Intuitivamente, há possibilidade de o teclado numérico ser associado à introdução de um código, embora não haja qualquer menção objetiva do mesmo, pelo que não é compatível com as boas práticas de *design*; no exemplo da direita, os botões e ações esperadas pelo utilizador estão devidamente identificados e podem ser convenientemente lidos por um sintetizador de voz (Zafar, 2017).

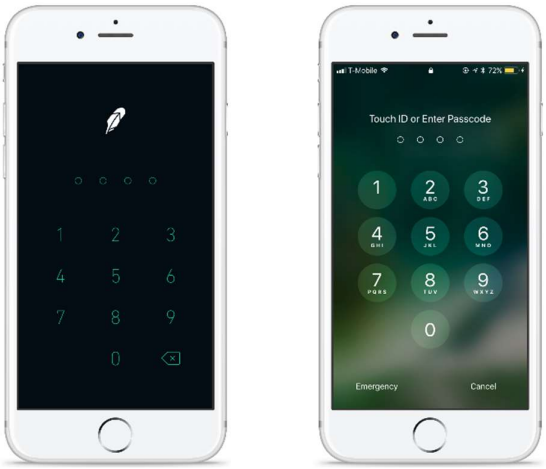

Figura 7 – Exemplo de linguagem descritiva em dispositivos móveis (Zafar, 2017)

Ainda dentro do contexto da acessibilidade, o programador deve ter em conta não só os utilizadores invisuais, mas também utilizadores com dificuldades visuais. Para tal, considera-se boa prática de *design* selecionar pares de cores contrastantes para a representação de texto bem visível em diferentes fundos, assim como o uso de símbolos e não apenas cores para representar diferentes tipos de elementos visuais, de modo a facilitar a leitura do ecrã a utilizadores daltónicos (Gupta, 2022).

Por fim, é aconselhada a inclusão de um “Modo Noturno” ou “Modo Noite”, dominado por textos claros e fundos escuros, que permite reduzir a fadiga ocular dos utilizadores em ambientes com pouca luz (Gupta, 2022), assim como melhorar a autonomia de alguns dispositivos.

2.4 Tecnologias existentes

No presente subcapítulo, são analisadas tecnologias atualmente disponíveis, que podem facilitar o desenvolvimento do projeto, minimizando o custo e tempo de conclusão do mesmo.

Paralelamente, permite ir ao encontro das necessidades do cliente, tendo em conta o ambiente tecnológico do SI, de modo que o projeto seja facilmente adaptado ao sistema.

2.4.1 Frontend

O desenvolvimento do *frontend* terá por base uma aplicação móvel. Atualmente, existem no mercado uma variedade de opções de ferramentas de desenvolvimento multiplataforma que permitem a compilação de código para diferentes dispositivos. Este *software* pode ser um tipo de *framework* ou *Development Kit*.

React Native

O React Native é uma *framework* baseada em JavaScript que tem como função principal o desenvolvimento de aplicações Android e iOS. Esta *framework* é baseada na biblioteca React, que permite o desenvolvimento de interfaces gráficas. O código utilizado é também composto por JSX, JavaScript XML. A *framework* utiliza API de renderização nas linguagens Objective-C e Java, para iOS e Android, respetivamente (Eisenman, 2015). A arquitetura da *framework* é representada pela Figura 8.

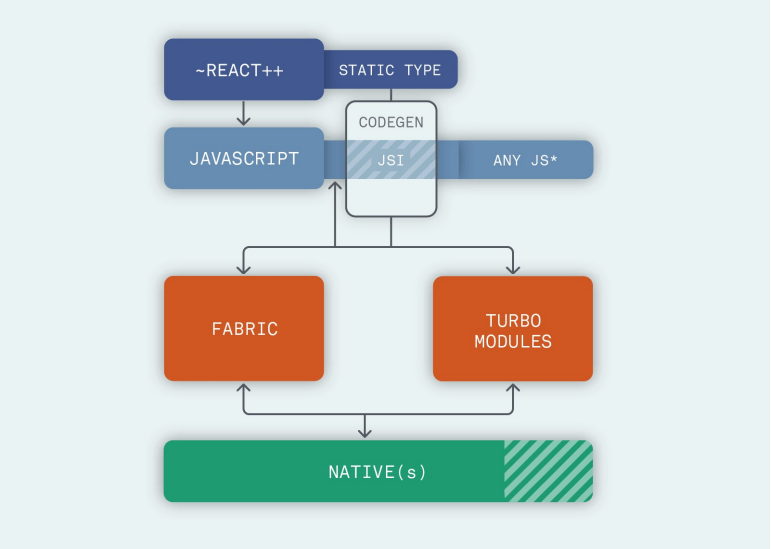

Figura 8 - Arquitetura da tecnologia React Native (Sciandra, 2019)

As principais vantagens do React Native são a sua performance, devido ao uso de API específicas de cada dispositivo, acesso a elementos de UI nativos e independência da *thread* principal da UI (Eisenman, 2015).

O seu desenvolvimento torna-se cómodo para o programador, uma vez que oferece ferramentas de *hot reload* e de *debug* inteligentes, e clareza no detalhe de erros, como exemplificado na Figura 9 (Eisenman, 2015).

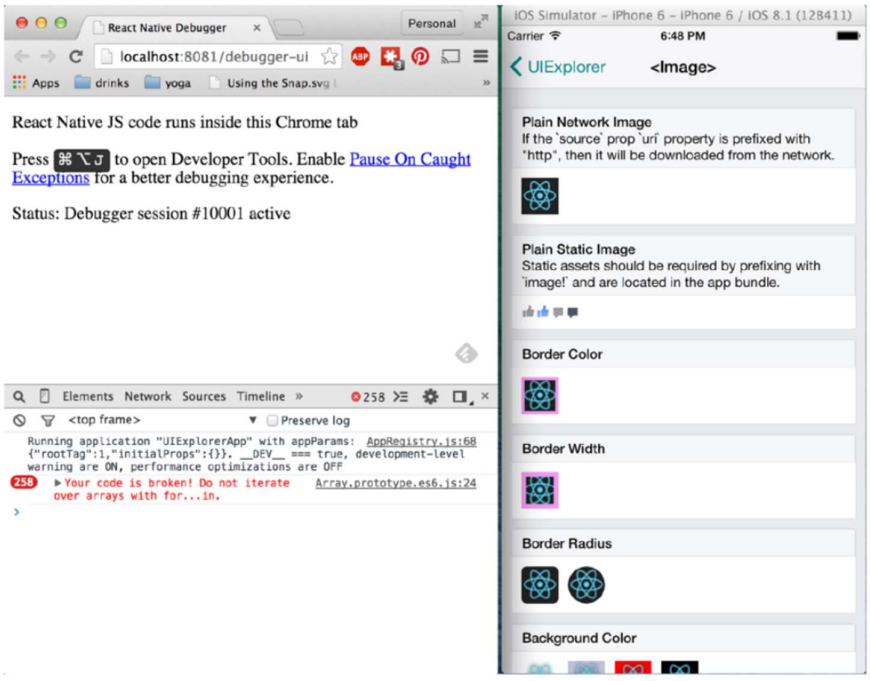

Figura 9 - Exemplo de *debug* de código React Native (Eisenman, 2015)

Flutter

O Flutter é uma *framework* multiplataforma da Google que tem como foco o desenvolvimento de aplicações *mobile*. Esta permite renderizar todos os componentes de visualização através dos seus próprios mecanismos de alto desempenho, em vez de recorrer a componentes *Web* ou *widgets* OEM do dispositivo. Desta forma, a performance do Flutter pode ser equiparada à de uma aplicação nativa (Wu, 2018).

Em termos de arquitetura, o código baseado em C/C++ é compilado através do NDK (Android *Native Development Kit*) e LLVM (*Low Level Virtual Machine*) do Android e iOS, respetivamente. A arquitetura da *framework* encontra-se na Figura 10.

O código utilizado baseia-se na linguagem de programação Dart. São utilizados compiladores AOT (*Ahead-of-Time*), em que o código é compilado antes de ser enviado para ambiente execução e JIT (*Just-In-Time*), ou seja, é convertido em nativo antes da execução do programa (Wu, 2018). Estes compiladores influenciam a velocidade do *software*, tanto em ambiente de desenvolvimento como em produção (Obinna, 2020).

O Flutter suporta a funcionalidade de *hot reload*, que permite a injeção de código-fonte na Dart VM - ambiente virtual de execução - sendo possível preservar a estrutura interna da aplicação, assim como todas as suas transições e ações, durante o edição de código (Wu, 2018).

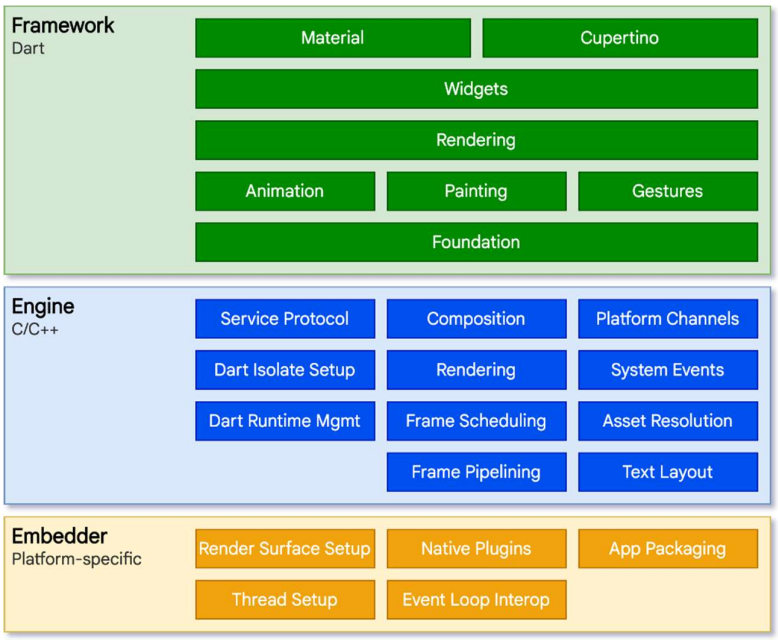

Figura 10 - Camadas da arquitetura do Flutter (Google, s.d.)

Ionic

O Ionic é uma *Framework open-source* que permite o desenvolvimento de aplicações para dispositivos móveis. Esta disponibiliza várias ferramentas e bibliotecas para desenvolvimento

de interfaces gráficas. *A framework* é baseada numa estrutura chamada *wrapper*, que consiste numa espécie de *container* que permite a execução de código em diferentes arquiteturas móveis (Chaudhary, 2018).

Esta tecnologia tem como base ferramentas como o HTML, o CSS e a linguagem de programação JavaScript. O Ionic é capaz de gerar aplicações híbridas, ou seja, que partilham componentes *Web* e nativos de cada dispositivo (Chaudhary, 2018), como descrito na Figura 11.

O Ionic utiliza o Apache Cordova como *wrapper* principal, para converter o código *Web* numa aplicação nativa Android ou iOS.

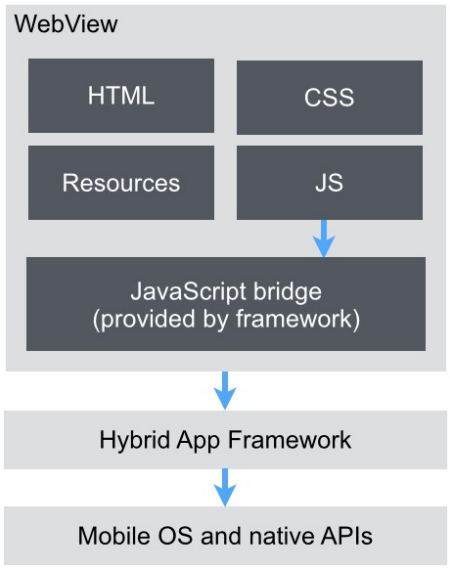

Figura 11 – Representação da arquitetura híbrida do Ionic (Ripkens, 2014)

2.4.2 Backend

Dada a adoção de um ecossistema Microsoft da parte do SI da FLUP e sendo a grande maioria dos sistemas existentes baseados nas linguagens de programação C# ou Visual Basic, faz sentido explorar os diferentes tipos de tecnologias que a empresa disponibiliza, que ofereçam vantagens no desenvolvimento do projeto.

Visto que a Microsoft não tenciona evoluir a linguagem VB (Ramel, 2020), o processo mais lógico será adotar um desenvolvimento *backend* baseado em C#, aliado à *Framework .NET* que oferece uma vasta lista de ferramentas e bibliotecas que agilizam o processo de conceção do *software*, bem como a comunicação entre camadas e a performance.

A disponibilização da linguagem LINQ (*Language-Integrated Query*) facilita a consulta de informações persistidas em forma de modelo de dados, uma vez que se encontra diretamente integrada na linguagem C# (Microsoft, s.d.).

2.4.3 Persistência de Dados

Mais uma vez, salienta-se a utilização de tecnologias Microsoft no contexto de persistência de dados, nomeadamente bases de dados SQL Server e SQL Server Express, atendendo ao contexto tecnológico da instituição.

2.4.4 Comparação de Tecnologias

2.4.4.1 Frontend

A comparação e avaliação de tecnologias de *frontend* é explicada, detalhadamente, no subcapítulo 3.2 do capítulo Análise de Valor, onde são considerados fatores de relevo que afetam a escolha da tecnologia mais adequada para o seu desenvolvimento e onde são explicadas as vantagens e desvantagens de cada opção, comparativamente às restantes.

Recorre-se também a cálculos estatísticos de modo a determinar, de forma mais objetiva, qual a tecnologia de *frontend* que será efetivamente utilizada na criação da aplicação multiplataforma.

2.4.4.2 Backend

Dadas as opções de mercado e as restrições de desenvolvimento dependentes dos requisitos do cliente, assim como a natureza da infraestrutura informática da instituição, é cada vez mais lógico optar-se pelo desenvolvimento em linguagem C#, uma vez que o VB está cada vez mais em desuso e que a Microsoft deixou de desenvolver novas funcionalidades e revisões da linguagem.

A nível de *framework*, uma das opções é o desenvolvimento de uma API Entity Framework .NET, que oferece uma variedade de ferramentas. Sendo esta baseada em modelos de dados, facilita a organização das entidades de negócio, consulta das respetivas informações e promove a modularidade do sistema, em oposição a um tipo de projeto sem um contexto de modelo de dados, que obrigaria a um trabalho acrescido na comunicação com a camada de persistência.

2.5 Conclusões

Uma vez analisados os dados relativos às necessidades e restrições impostas pelo cliente, foi definido que o projeto será desenvolvido recorrendo à tecnologia Flutter para o desenvolvimento da aplicação multiplataforma, que será o *frontend* do sistema.

O *backend* basear-se-á na *Framework .NET*, de modo a contruir uma API de comunicação com o *frontend* e outros serviços existentes, em linguagem C#, em combinação com o formato JSON (*JavaScript Object Notation*) na troca de dados.

Por fim, devido às restrições impostas pelo SI e pelos aspetos referidos anteriormente no que concerne às opções de tecnologias de armazenamento, a persistência de dados deverá ser

assegurada por uma base de dados SQL, responsável por gerir as estruturas de modelo de dados e respetivas informações criadas a partir da API .NET.

3 Análise de Valor

A análise de valor da solução tem como principal objetivo avaliar e identificar os fatores que favorecem o valor do produto desenvolvido, preservando uma boa relação de custo/qualidade. Neste capítulo será analisado o processo da inovação do produto e os fatores que acrescentam valor ao mesmo, na medida em que responde às necessidades do cliente. Desta forma, serão utilizados métodos matemáticos de análise de dados, que permitem obter conclusões deste estudo (Crawford & Benedetto).

3.1 Processo de Inovação

O processo de inovação é definido por um conjunto de passos intermédios que irão levar à comercialização de um produto. Peter Koen, professor da *Wesley J. Howe School of Technology Management* (Peter Koen, s.d.), estudou e desenvolveu métodos de otimização deste processo.

Segundo a documentação de Koen, o processo de inovação pode ser dividido em três etapas (Koen, 2002), ilustradas na Figura 12.

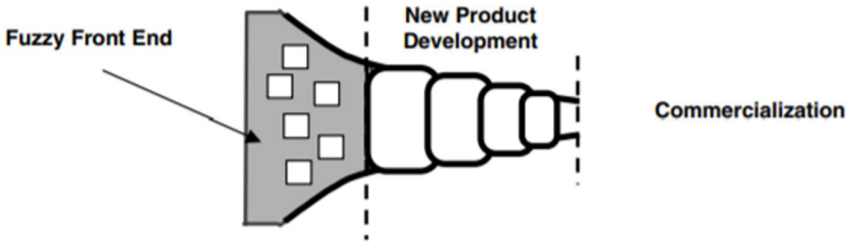

Figura 12 - Etapas do Processo de Inovação (Koen, 2002)

- *Fuzzy Front End* – primeira parte do processo de inovação do produto, em que são analisadas e avaliadas as ideias, antes do processo de desenvolvimento do produto. Desta forma, definem-se as oportunidades de melhoria do projeto em função das necessidades do cliente, aumentando o seu benefício (Schreiner, Mello, & Nascimento,

2015). É nesta fase que se decide se é pertinente ou não gastar recursos para desenvolver a ideia.

- *New Product Development* – consiste na etapa de conceção do produto previamente avaliado, em que podem ocorrer alterações ao planeamento consoante as necessidades.
- Comercialização do Produto – inserção do produto no mercado, de forma a ser vendido ao cliente final

3.2 New Concept Development

O *New Concept Development* é um modelo evoluído, desenvolvido a partir do conceito de *Fuzzy Front End*, que tem como objetivo decompor e normalizar as diferentes fases do processo de inovação, sendo normalmente constituído por quatro fases:

- Identificação de Oportunidade
- Análise de Oportunidade
- Formação de Ideias
- Avaliação e Seleção de Ideias

3.2.1 Identificação de Oportunidade

Relativamente ao projeto em causa, as oportunidades de mercado identificadas sobre o produto desenvolvido são:

- Existência de poucas ou nenhumas aplicações *mobile* que implementem a comunicação com o tipo de máquinas de *vending* disponibilizadas na FLUP, através de mecanismos fiáveis
- O conjunto de benefícios que o projeto é capaz de gerar para a instituição como forma de combate à pandemia, bem como no âmbito de gestão de serviços
- Existência de utilizadores com deficiência visual

3.2.2 Análise de Oportunidade

Após identificada a oportunidade de mercado do negócio, são explorados os diferentes fatores que afetam o projeto, através da ferramenta denominada análise SWOT. É efetuado um estudo dos pontos fortes e fracos (análise interna), e fatores de oportunidade e ameaças para o negócio (análise externa).

• **Forças** (*Strengths*) - Atributos positivos do negócio que estão diretamente relacionados com o sucesso no alcance dos objetivos definidos para o projeto

• **Fraquezas** (*Weaknesses*) – Fatores potencialmente prejudiciais ao sucesso e alcance dos objetivos do projeto, dentro do âmbito do negócio

• **Oportunidades** (*Opportunities*) – Fatores externos que afetam positivamente o projeto, normalmente associados ao estado do mercado onde o produto se enquadra

• **Ameaças** (*Threats*) - Fatores externos que afetam negativamente o projeto, frequentemente relacionados com o ambiente mercantil

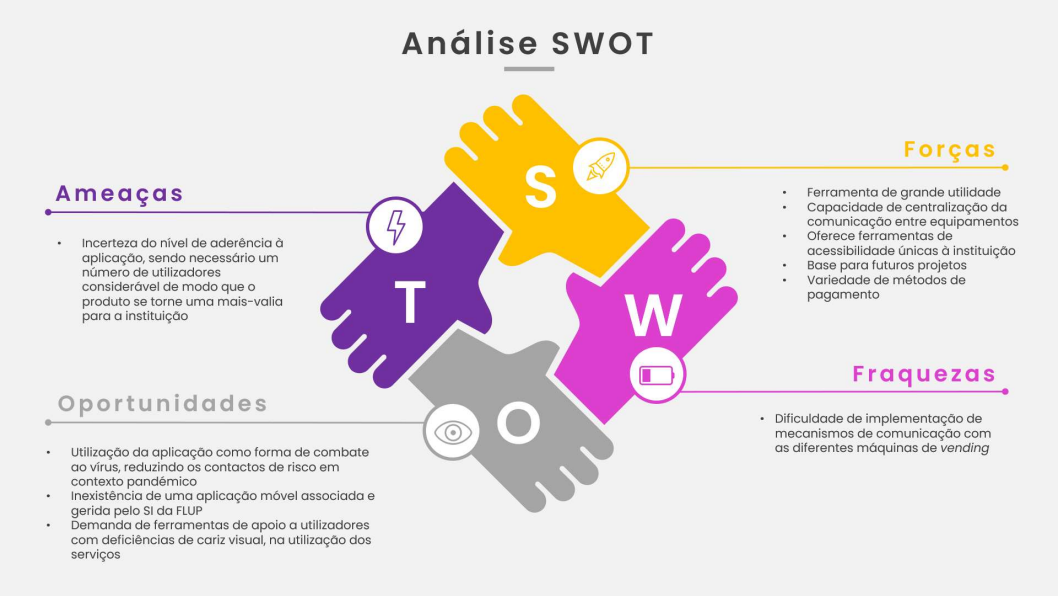

Figura 13 - Gráfico ilustrativo da análise SWOT

Forças

- O produto resultante do projeto em questão constitui uma ferramenta essencial, de grande utilidade para a FLUP, que visa centralizar os serviços atualmente existentes, numa aplicação *mobile*
- A comunicação com as máquinas de *vending* da FLUP não é universal, ou seja, não é utilizado sempre o mesmo protocolo ou API, em diferentes equipamentos. O *software* desenvolvido é feito mediante as necessidades da instituição, convergindo várias interfaces numa só plataforma
- É a única ou das poucas ferramentas que oferece funcionalidades de apoio a indivíduos com deficiência visual na instituição
- Criação de uma plataforma base para o possível incremento de novas funções
- Disponibilização de um portal de opções de pagamentos móvel, inexistente até à data

Fraquezas

- Atualização e manutenção do *software* no que diz respeito à comunicação com as máquinas de *vending* torna-se mais árduo, visto que as interfaces disponibilizadas para o acesso às mesmas não são públicas e estão pobremente documentadas, podendo gerar futuros constrangimentos

Oportunidades

- Utilização da aplicação como forma de reduzir a disseminação do vírus, reduzindo os contactos de risco em contexto pandémico e de outras eventuais doenças transmissíveis do mesmo modo
- Inexistência de uma aplicação móvel associada e gerida pelo SI da FLUP
- Demanda de ferramentas de apoio a utilizadores com baixa visão ou invisuais

Ameaças

- Incerteza do nível de aderência à aplicação, sendo necessário um número de utilizadores considerável, de modo que o produto se torne uma mais-valia para a instituição

3.2.3 Formação de Ideias

Uma vez identificados os fatores internos e externos que influenciam o sucesso do negócio, são investigados os métodos candidatos para a concretização do projeto. Uma vez identificados, são registados e descritos para que, posteriormente, seja analisada a melhor opção. No contexto do problema, é relevante identificar as principais tecnologias disponíveis para concretizar o processo de desenvolvimento, sendo estas baseadas na análise de soluções atualmente existentes. Identificaram-se as seguintes hipóteses:

- **React Native** – *framework open-source*, baseada na biblioteca React JS e na linguagem JavaScript. Utiliza componentes nativos das plataformas em vez de componentes Web. (React Native, 2022)
- **Flutter** – *framework open-source* criada pela empresa Google, que oferece ferramentas de compilação para código nativo do dispositivo (iOS ou Android), juntamente com bibliotecas de UI, e utiliza a linguagem de programação Dart. (Thomas, 2019)
- **Ionic** – *framework open-source* baseada em Angular JS, Apache Cordova e na linguagem JavaScript. Integra uma combinação de componentes *Web* e nativos dos dispositivos. (Haire, s.d.)
- **Nativo** – Desenvolvimento orientado especificamente a cada dispositivo, neste caso, Android (em linguagem Java) e iOS (em Swift ou Objective-C)

3.2.4 Avaliação e Seleção de Ideias

Nesta secção, são avaliadas as ideias selecionadas anteriormente através de métodos analíticos que combinam uma série de critérios relevantes na conceção do produto e as opções de tecnologias de desenvolvimento mencionadas anteriormente, com o objetivo de escolher a mais indicada a ser aplicada. Foi selecionado o método *Analytic Hierarchy Process* (AHP), de

apoio à decisão, de modo a comparar as opções seus valores para o negócio (Saaty, 1987). O método consiste em:

- Dividir hierarquicamente os critérios de avaliação pelas opções formalizadas no subcapítulo anterior
- Estabelecer a importância de cada critério face a cada opção
- Confirmar e validar a consistência dos dados tendo em conta as relações de importância entre critérios e opções

3.2.4.1 Divisão hierárquica

De forma concluir qual a melhor tecnologia a aplicar ao desenvolvimento deste projeto, foram selecionados os seguintes critérios de avaliação:

- **Tempo** – refere-se ao tempo de desenvolvimento do projeto, associado às necessidades de cada tecnologia
- **Performance** – refere-se, principalmente, ao tempo de resposta do *software* e à fluidez na navegação de UI
- **Curva de Aprendizagem** – diz respeito ao conhecimento adicional necessário para utilizar corretamente as ferramentas disponibilizadas por cada tecnologia
- **Custo** – valor do projeto consoante as horas de trabalho despendidas a desenvolver o produto

Após a identificação dos critérios e das ideias das tecnologias a adotar, é construída uma árvore hierárquica, representativa das comparações que serão convertidas em forma de cálculo.

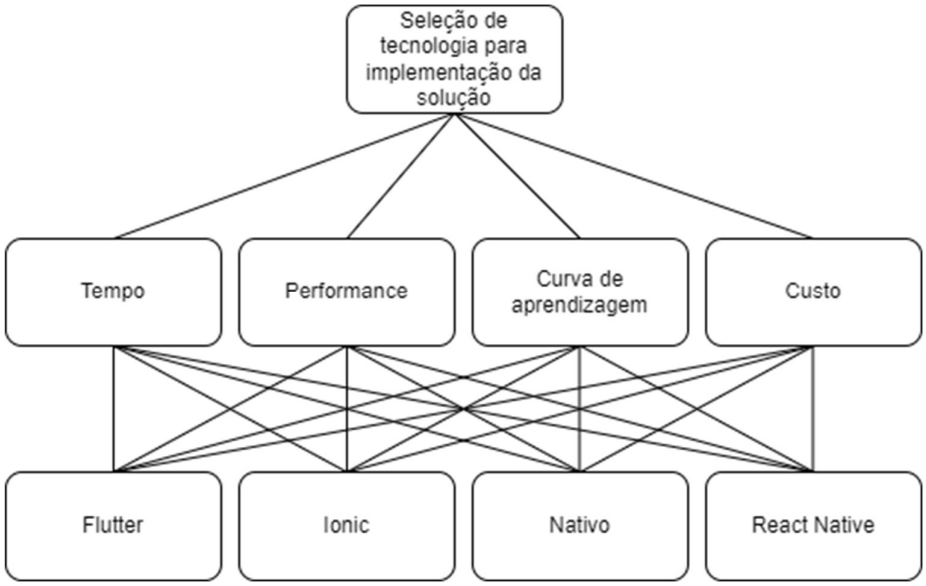

Figura 14 - Árvore hierárquica do QFD

3.2.4.2 Peso de Critérios e Opções

Segundo a análise AHP, as prioridades dos critérios identificados são classificadas de acordo com a seguinte escala de valores (Saaty, 1987), representada pela Tabela 2:

Tabela 2 - Graus de importância e respetivo significado (Saaty, 1987)

Grau de importância	**Descrição**
1	Igual importância
3	Fraca Importância
5	Forte Importância
7	Muito Forte Importância
9	Importância Absoluta
2,4,6,8	Valores Intermédios

A Tabela 3 representa as relações de prioridade entre os diferentes critérios. A última coluna diz respeito ao resultado do cálculo das respetivas prioridades relativas.

Tabela 3 - Tabela de prioridades de critérios

	Tempo	**Performance**	**C. Aprendizagem**	**Custo**	**Prioridades**
Tempo	1	3	1/2	6	0,293
Performance	1/3	1	1/6	5	0,137
C. Aprendizagem	2	6	1	7	0,520
Custo	1/6	1/5	1/7	1	0,050

Podemos concluir, através dos valores de prioridade da Tabela 3, que:

- O Tempo de desenvolvimento é ligeiramente mais importante que a Performance do *software*; o resultado do projeto é uma necessidade urgente para a instituição, logo, priorizou-se o encurtamento do prazo de lançamento da aplicação face à Performance na escolha da tecnologia, mas não de forma a descurar a última.
- A Curva de Aprendizagem é um fator ligeiramente mais importante que o Tempo; embora estejam relacionados, o conhecimento sobre a tecnologia tem uma importância ligeiramente maior, visto ser necessário construir um software compreensível, de fácil manutenção e gestão.
- O Tempo é um fator com um forte grau de importância comparativamente ao Custo; tal como mencionado anteriormente, o lançamento do produto é urgente, fazendo com que este fator se sobreponha à eventualidade do valor do projeto encarecer.
- A Performance é fortemente valorizada, comparativamente ao Custo; a responsividade do sistema é um fator importante, principalmente no que toca ao fluxo de utilizadores internos e externos, bem como ao nível de UX, não sendo um *trade-off* viável em relação ao Custo.
- A Curva de Aprendizagem é muito mais valorizada que a Performance; como referido anteriormente, existe uma relação com o tempo de realização do projeto. Por outro lado, o SI será, futuramente, responsável por manter o *software* que se pretende que seja de fácil compreensão e aberto a atualizações, fator que se sobrepõe à responsividade do sistema.
- A Curva de Aprendizagem é um fator com um forte grau de importância comparativamente ao Custo; mais uma vez, é destacada a importância da facilidade de manutenção (a par do Tempo) na realização do projeto face ao custo total do mesmo.

No que toca às tecnologias em função de cada critério, foram calculadas as prioridades relativas que são apresentadas nas Tabelas Tabela 4, Tabela 5 ,Tabela 6 ,Tabela 7 e Tabela 8:

Tabela 4 - Tabela de prioridades do critério Tempo

Tempo	Flutter	React Native	Ionic	Nativo	Prioridades Relativas
Flutter	1	4	2	9	0,489
React Native	1/4	1	1/4	5	0,133
Ionic	1/2	4	1	8	0,337
Nativo	1/9	1/5	1/8	1	0,041

- **Tempo** – o Flutter é a tecnologia destacada por exigir menos tempo de desenvolvimento, logo a seguir ao Ionic com características semelhantes. O React Native utiliza elementos Web e nativos que podem exigir uma posterior otimização conforme a plataforma, o que pode tornar o seu desenvolvimento mais demorado (Skuza, Mroczkowska, & Włodarczyk, 2021). A opção de desenvolver uma aplicação nativa para cada dispositivo é de longe a que mais tempo requer, visto que não é utilizada nenhuma *framework* capaz de compilar código para diferentes sistemas.

Tabela 5 - Tabela de prioridades do critério Performance

Performance	Flutter	React Native	Ionic	Nativo	Prioridades Relativas
Flutter	1	3	5	1/3	0,264
React Native	1/3	1	5	1/4	0,155
Ionic	1/5	1/5	1	1/7	0,052
Nativo	3	4	7	1	0,530

- **Performance** – o desempenho de uma aplicação nativa é, por norma, superior, visto que quaisquer otimizações poderão ser feitas recorrendo a código nativo, no caso do Android em Java e, no caso do iOS, Swift ou Objective-C, embora tenha sido reportado, pela própria empresa, a Apple, que o Swift é 2.6 vezes mais rápido que o Objective-C (Apple, s.d.). A utilização de componentes nativos da parte do Flutter e do React Native faz com que estas tecnologias não estejam muito distantes da performance nativa, embora utilizem ferramentas e linguagens de programação distintas. Por fim, a performance do Ionic é impactada pela utilização mista de componentes Web e nativos.

Tabela 6 - Tabela de prioridades do critério Curva de Aprendizagem

C. Aprendizagem	Flutter	React Native	Ionic	Nativo	Prioridades Relativas
Flutter	1	4	1	9	0,414
React Native	1/4	1	1/4	6	0,143
Ionic	1	4	1	8	0,404
Nativo	1/9	1/6	1/8	1	0,040

- **Curva de Aprendizagem** – a nível de conhecimento necessário para desenvolvimento do *software* com as tecnologias mencionadas, o Flutter e o Ionic apresentam aspetos e ambientes muito semelhantes, de fácil compreensão. A possível necessidade de otimizações por dispositivo da parte do React Native requer um conhecimento adicional sobre cada plataforma. O desenvolvimento nativo força a uma aprendizagem do código nativo de cada sistema, que o torna muito mais árduo.

Tabela 7 - Tabela de prioridades do critério Custo

Custo	Flutter	React Native	Ionic	Nativo	Prioridades Relativas
Flutter	1	5	2	9	0,507
React Native	1/5	1	1/4	5	0,125
Ionic	1/2	4	1	8	0,328
Nativo	1/9	1/5	1/8	1	0,040

- **Custo** – este fator está intrinsecamente associado ao Tempo. O Flutter e o Ionic estão sensivelmente no mesmo patamar no que se refere ao número de horas de trabalho requeridas, que se traduzem em custo acrescido para o cliente. Como referido anteriormente, o React Native, devido à sua natureza, poderá necessitar de otimizações específicas, o que incrementará o tempo necessário para a realização do projeto e, consequentemente, o seu custo. Por fim, o desenvolvimento nativo é o que mais impacta o custo do projeto, dado os requisitos já mencionados.

Através do cálculo das prioridades relativas das soluções apresentadas na Tabela 8, conseguimos verificar que, numa primeira análise, a solução mais apelativa de modo a rentabilizar os recursos disponíveis consoante os critérios é o **Flutter**. No próximo subcapítulo, será analisada a consistência dos dados aqui apresentados.

Tabela 8 - Prioridades relativas entre soluções e critérios

Solução	Tempo	Performance	C. Aprendizagem	Nativo	PR (Critérios)	PR Finais
Flutter	0,049	0,264	0,414	0,507	0,293	**0,420**
React Native	0,133	0,155	0,143	0,125	0,137	0,140
Ionic	0,337	0,052	0,404	0,328	0,520	0,332
Nativo	0,041	0,530	0,040	0,040	0,050	0,107

3.2.4.3 Validação de Consistência

A consistência dos dados é avaliada através do cálculo da Razão de Consistência (RC), de forma a confirmar se os juízos feitos relativamente aos pesos de critérios apresentam consistência comparativamente a amostras maiores. O método AHP baseia-se em avaliações racionais, ou seja, se um critério X tiver um grau de importância maior do que o de Y e Y maior do que Z então X é mais importante que Z. Só é possível afirmar a consistência dos dados se o RC calculado for inferior a 0.1 (10%), que se traduz em julgamentos confiáveis (Saaty, 1987). Para obter o RC, é necessário o valor do Índice de Consistência (IC), calculado através da fórmula 1.

$$IC = \frac{\lambda_{max} - n}{n - 1} \qquad (1)$$

O valor λ_{max} corresponde ao maior valor próprio da matriz de comparação das prioridades relativas e n à ordem da matriz.

O RC é calculado através da seguinte fórmula, em que o IR corresponde ao Índice Aleatório, um valor obtido de uma tabela de constantes resultante de valores aleatórios, formados através da escala AHP (Saaty, 1987) e o IC ao Índice de Consistência previamente calculado:

$$RC = \frac{IC}{IR} \qquad (2)$$

A Tabela 9 define o Índice Aleatório para matrizes quadradas de ordem n:

Tabela 9 - Tabela de Índices Aleatórios para matrizes quadradas (Saaty, 1987)

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0.00	0.00	0.58	0.90	1.12	1.24	1.32	1.41	1.45	1.49	1.51	1.48	1.56	1.57	1.59

Validação de Consistência de Critérios

Primeiramente, é obtido o valor do IC através do cálculo do λ_{max}, correspondente à matriz de ordem n de comparação de prioridades:

$$\begin{bmatrix} 1 & 3 & \frac{1}{2} & 6 \\ \frac{1}{3} & 1 & \frac{1}{6} & 5 \\ 2 & 6 & 1 & 7 \\ \frac{1}{6} & \frac{1}{5} & \frac{1}{7} & 1 \end{bmatrix} * \begin{bmatrix} 0{,}293 \\ 0{,}137 \\ 0{,}520 \\ 0{,}050 \end{bmatrix} = \begin{bmatrix} 1{,}263 \\ 0{,}570 \\ 2{,}277 \\ 0{,}200 \end{bmatrix}$$

$$IC = \frac{\left(\frac{1{,}263}{0{,}293} + \frac{0{,}570}{0{,}137} + \frac{2{,}277}{0{,}520} + \frac{0{,}200}{0{,}050}\right) - 4}{3} = 0{,}073$$

Uma vez obtido o IC, é consultado o valor de IR tabelado, correspondente à ordem da matriz em questão e é aplicada a fórmula de cálculo do RC:

$$RC = \frac{0{,}073}{0{,}90} = 0{,}081 < 0{,}1$$

O valor do RC obtido foi de 0,081, ou seja, inferior à margem de inconsistência de 10%. Conclui-se que os dados resultantes da utilização do método AHP são válidos.

Seguiu-se a mesma lógica no cálculo das razões de consistência para cada critério em função de cada opção, sendo que foram obtidos os seguintes valores (omitiram-se os passos intermédios para simplificação da leitura dos resultados, uma vez que a lógica dos cálculos se repetiria):

$$RC_{tempo} = 0{,}055 < 0{,}1$$

$$RC_{performance} = 0{,}086 < 0{,}1$$

$$RC_{c.aprendizagem} = 0{,}053 < 0{,}1$$

$$RC_{custo} = 0{,}059 < 0{,}1$$

Conclui-se que os valores calculados estão todos dentro da margem de inconsistência, sendo possível afirmar que os dados são válidos e que a utilização do Flutter como tecnologia é a opção mais apropriada para o desenvolvimento do projeto.

3.3 Valor da Solução

No presente capítulo, é avaliado o valor que o produto resultante do projeto oferece ao cliente, consoante os seus requisitos. O método *Quality Function Deployment* (QFD) é usado para identificar a relação entre as necessidades do cliente e as funcionalidades do *software*, sendo, desta forma, definido o valor da solução (Crawford & Benedetto).

3.3.1 Definição de Valor

O conceito de valor está diretamente ligado a elementos de *marketing* e é um pré-requisito fulcral para o sucesso de um produto. Este é normalmente associado à relação entre os benefícios que um projeto oferece ao cliente e os custos que o mesmo acarreta (Graf & Maas, 2008).

No domínio de um produto, o conceito de valor está associado à sua relação qualidade/preço. Muitos autores marcam a importância da análise e diferenciação de indicadores de valor, podendo estes ser intrínsecos e extrínsecos: Intrínsecos, como é exemplo da qualidade do produto: fazem parte dele mesmo e só podem sofrer alterações dependendo da modificação do produto; indicadores extrínsecos como preço, marca, nível de publicidade ou país de origem estão relacionados com o produto, mas não são inerentes ao produto em si, ou seja, podem mudar ao longo do tempo. Nesse contexto, a qualidade é considerada mediadora na relação entre todos os indicadores extrínsecos. O custo funciona como mediador na relação entre o preço e o valor. Desta forma, o preço serve como um indicador extrínseco tanto para o custo como para a qualidade (Graf & Maas, 2008).

3.3.2 Proposta de Valor

A apresentação da proposta de valor ao cliente deve ser feita de uma forma compreensível e apelativa. Para tal, é utilizado o modelo *Business Model Canvas*, em forma de gráfico representado na Figura 15, com o objetivo de explicitar os benefícios e as diferentes estratégias do negócio, justificando a compra do produto.

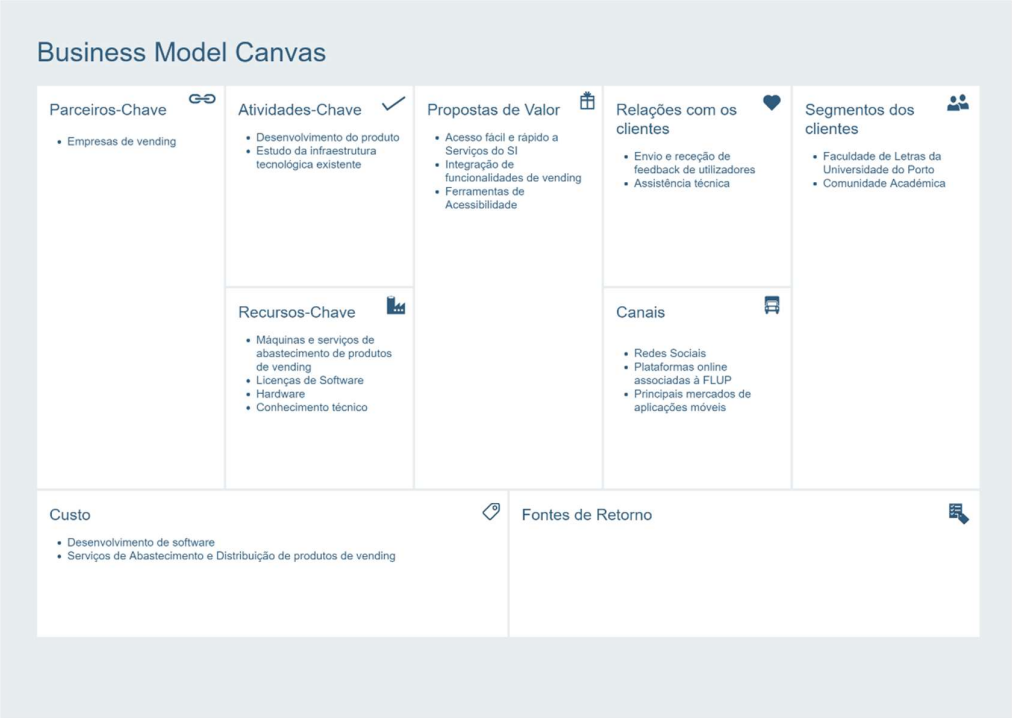

Figura 15 - Ilustração do *Business Model Canvas* do produto

- **Parceiros-Chave** - parceiros de negócio envolvidos no desenvolvimento do produto, sendo o caso das empresas de *vending*, que disponibilizam serviços de abastecimento de máquinas e API de comunicação com as mesmas.
- **Atividades Chave** – ações ligadas à concretização do produto. Destacam-se o desenvolvimento do *software* e o estudo da infraestrutura tecnológica da instituição
- **Recursos-Chave** - recursos indispensáveis para a concretização do projeto. Neste contexto, é importante referir todo o *hardware* de computação e de *vending*, as licenças de *software* e o conhecimento técnico
- **Proposta de Valor** – razões válidas que justificam a aquisição do produto em questão. Neste caso, identificou-se a simplicidade de acesso aos serviços do SI, a integração das funcionalidades de *vending* e ferramentas de acessibilidade associadas
- **Relações com os clientes** – modos de comunicação com os clientes, nomeadamente através de suporte técnico e de interação com o utilizador final através de *feedback*
- **Canais** – meios que permitem que o produto seja distribuído ao cliente: redes sociais, plataformas *online* associadas à FLUP e principais lojas *online* de aplicações
- **Fontes de Retorno** – no projeto em causa, não são consideradas fontes de retorno do produto
- **Segmentos de Clientes** – no atual contexto, os principais clientes são a Faculdade de Letras da Universidade do Porto e a sua respetiva comunidade académica
- **Custo** – o custo do projeto depende do desenvolvimento do *software* e dos serviços de abastecimento e distribuição de produtos de *vending*

3.3.3 Quality Function Deployment

O *Quality Function Deployment* (QFD) é um método que permite assegurar que as necessidades do cliente são respeitadas no desenvolvimento de um produto. É também utilizado como forma de comparação entre os atributos do produto e dos seus concorrentes. Para tal, é utilizada uma ferramenta denominada de *House of Quality* (HOQ) que evidencia as características que interessam aos clientes e estabelece uma relação com os aspetos técnicos do produto. Estas interligações incentivam a comunicação entre os intervenientes do projeto, de diferentes áreas, de forma a haver uma reflexão conjunta e consensual sobre a conceção do mesmo (Crawford & Benedetto).

3.3.3.1 Necessidades do Cliente

De forma a definir o sentido que o projeto tomará, é necessário compreender as necessidades do cliente ou clientes, e identificar os benefícios que estes pretendem extrair das qualidades do produto. Para o projeto em questão, recorreu-se a uma análise feita em conjunto com o Serviço de Informática da FLUP e a comunidade académica. Identificaram-se as seguintes características de valor:

- **Integração de Serviços do SI** – as funcionalidades associadas a contas dos utilizadores de rede da FLUP necessitam de estar interligadas, visto que existe o conceito de saldo em cartão associado a estudantes, docentes, *staff*, entre outros utilizadores. Existe necessidade de o sistema consumir outros serviços existentes.
- **Diversidade de Opções de Pagamento** – suporte dado à variedade de métodos de pagamento que o utilizador deverá conseguir optar, podendo estes ser internos ou externos.
- **Funcionalidades de *vending*** – ferramentas de comunicação e gestão do processo de distribuição de artigos das máquinas de *vending*
- **Custo** – valor de aquisição do produto

3.3.3.2 Atributos Técnicos

Este subcapítulo foca os aspetos de engenharia informática que vão também ao encontro das necessidades do(s) cliente(s). Na análise anteriormente efetuada, foram discutidas e destacadas as seguintes características de interesse:

- **Experiência de utilização da UI (UX)** – qualidade da interação entre o utilizador e a interface gráfica da aplicação.
- **Responsividade do Sistema** – fator associado ao tempo de resposta do *software.*
- **Aspetos Visuais da Aplicação** – características apelativas da aparência da interface gráfica da aplicação que podem incluir animações, temas, menus de navegação, entre outros elementos gráficos.
- **Modularidade do Sistema** – capacidade de o software ser subdividido em módulos com diferentes funções, facilitando a manutenção e a compreensão de um código mais "limpo".

3.3.3.3 Concorrência

Antes do preenchimento do gráfico, são ainda identificados os concorrentes de mercado capazes de oferecer uma solução para o problema. Visto que não existe um produto universal para a concretização do projeto, e dadas as características e o ambiente específico do projeto, serão mencionados e analisados os tipos de soluções que podem ser aplicadas:

- **Empresas de *vending*** – oferecem serviços de abastecimento e gestão de máquinas de *vending*, normalmente de modelos variados. No entanto, esta gestão pode depender de *software* de terceiros.
- **Fabricantes de máquinas de *vending*** – comercializam o seu *hardware* e, quando disponível, *software* de comunicação com os seus equipamentos
- **Empresas de Serviços** – entidades ligadas, especificamente, ao desenvolvimento de *software*
- **Comunidades inclusivas** – conjunto de indivíduos ou organização que valoriza a acessibilidade na utilização dos serviços disponibilizados à comunidade académica, principalmente a utilizadores com necessidades especiais.

3.3.3.4 House of Quality

Após a identificação das necessidades e qualidades que o cliente procura, bem como os concorrentes no mercado, os dados são descritos na matriz de planeamento (Crawford & Benedetto), representada na Figura 16.

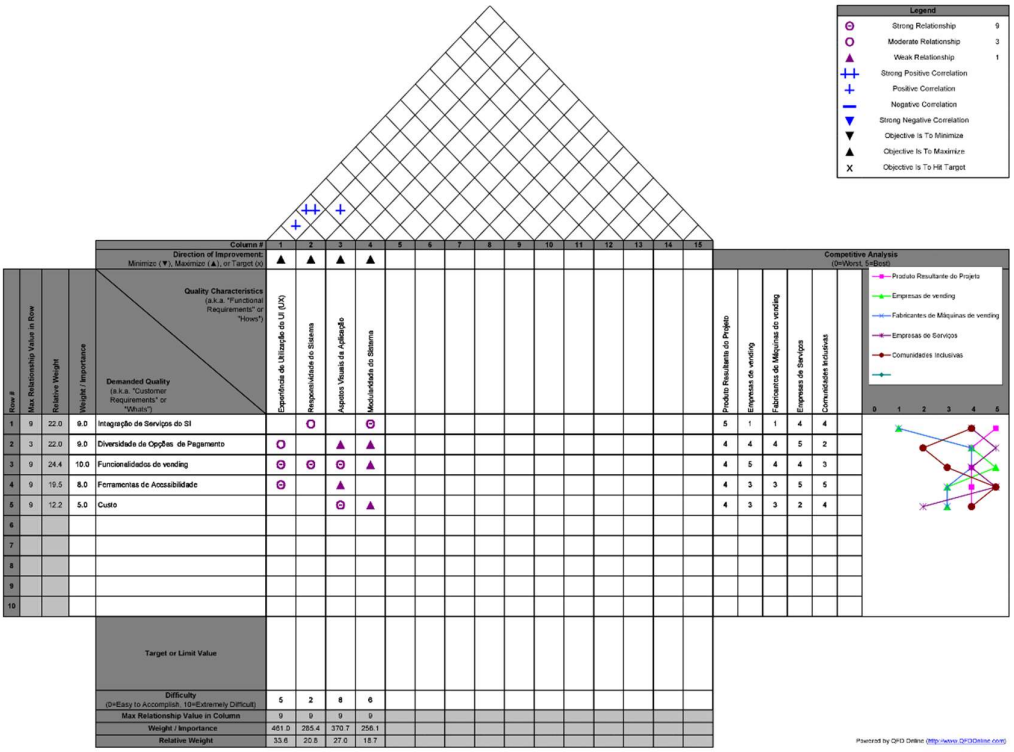

Figura 16 - Matriz de planeamento HOQ (QFD Online, 2007)

3.4 Conclusões

Após o esclarecimento do valor da proposta de solução apresentada, foram expostas as principais ideias que fortalecem o seu valor consoante as necessidades e fatores de relevo para o cliente. Seguidamente, recorrendo ao método de análise estatística AHP, foram relacionados os critérios de prioridade e as hipóteses de tecnologias de desenvolvimento que poderiam ser aplicadas ao projeto. Uma vez selecionada a opção mais viável, procedeu-se à validação da consistência dos pesos das prioridades relativas entre critérios e soluções.

Finalmente, após obtidos resultados concretos e válidos, pode concluir-se que a utilização do Flutter como ferramenta de desenvolvimento da aplicação multiplataforma do sistema descrito é a opção que mais se adequa, e responde às necessidades do cliente.

4 Análise de Requisitos

Este capítulo descreve os requisitos do cliente de forma categorizada, clara e detalhada, assim como o modelo de domínio respeitante ao negócio, e noções do mesmo.

4.1 Modelo de Domínio

Os conceitos do domínio do problema são representados através do diagrama da Figura 17, que segue a abordagem *Domain-Driven Design* (DDD) (Nilsson, 2006), um conjunto de objetos abstratos que expõe a ideia do negócio apresentado.

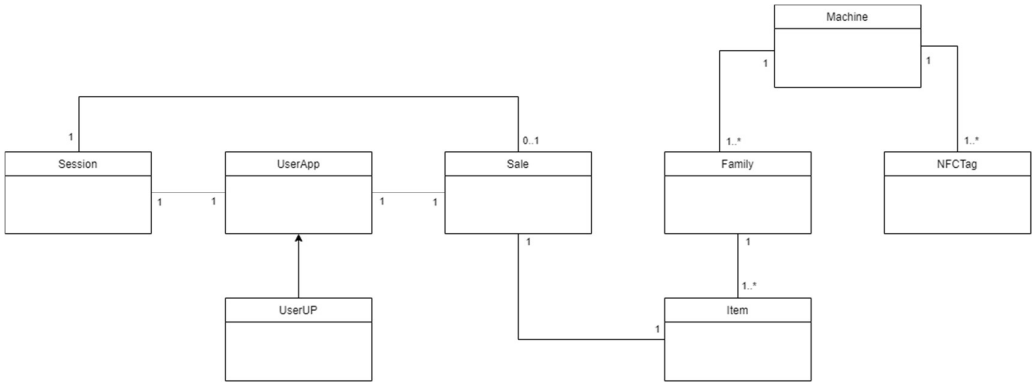

Figura 17 – Diagrama de Modelo de Domínio

- **UserApp** – Utilizador global da aplicação, que contém as informações genéricas partilhadas por todos os tipos de utilizadores
- **UserUP** – Utilizador interno da FLUP, registado no controlador de domínio
- **Sale** – Registo de uma compra efetuada por um utilizador
- **Session** – Registo de utilização da aplicação por um utilizador
- **Machine** – Representação de uma máquina de *vending* fisicamente instalada
- **Item** – Representação de um artigo vendável ao consumidor final
- **Family** – Categoria de artigos
- **NFCTag** – Identificação de uma etiqueta NFC, correspondente a uma máquina física

4.2 Requisitos Funcionais

Os requisitos concebidos são os seguintes:

- **RF01** – Os utilizadores internos podem iniciar sessão na aplicação através das funcionalidades de controlo de domínio existentes

- **RF02** – Os utilizadores externos podem registar-se, e iniciar sessão na aplicação com os seus respetivos dados
- **RF03** – Os utilizadores têm acesso a um perfil onde podem consultar as suas informações
- **RF04** – Apenas os utilizadores externos conseguem alterar/recuperar a respetiva palavra-passe
- **RF05** – Os utilizadores podem ativar/desativar o seu perfil de *vending*, que permite o acesso às respetivas funcionalidades de interação com as máquinas
- **RF06** – Os utilizadores conseguem terminar sessão, em todos os dispositivos anteriormente ligados
- **RF07** – Os utilizadores conseguem consultar o saldo da conta
- **RF08** – Os utilizadores conseguem carregar as respetivas contas com saldo, através de métodos de pagamento *online*
- **RF09** – Os utilizadores conseguem identificar uma máquina, através da leitura de um código QR
- **RF10** – Os utilizadores conseguem identificar uma máquina, através da leitura de uma ou múltiplas etiquetas NFC
- **RF11** – Os utilizadores podem consultar uma lista de categorias de produtos de uma máquina de *vending*
- **RF12** – Os utilizadores podem optar por consultar os produtos de uma categoria (e respetivas informações), disponíveis para compra
- **RF13** – Os utilizadores podem optar por ajustar o nível de açúcar de bebidas que o permitam
- **RF14** – Os utilizadores podem optar por incluir ou não um copo, na compra de bebidas que o permitam
- **RF15** – Os utilizadores devem conseguir, através da aplicação, solicitar um produto que se encontre disponível numa máquina de *vending*, integrada no sistema

4.3 Requisitos Não Funcionais

- **RNF01** – A aplicação deve estar disponível em mais do que um idioma (Português (Portugal) e Inglês)
- **RNF02** – O desenvolvimento do sistema deve reger-se pelas boas práticas de engenharia de *software*
- **RNF03** – A aplicação deverá ter um *layout* visual compreensível, simples de navegar e apelativo para o utilizador
- **RNF04** – A aplicação deverá suportar ferramentas de acessibilidade para utilizadores invisuais e com dificuldades de cariz visual
- **RNF05** – A aplicação deverá ser multiplataforma, mais concretamente, possuir versões para o sistema Android e iOS

4.4 FURPS+

O FURPS+ é um modelo de qualidade que permite classificar de forma detalhada os requisitos não funcionais do projeto a ser desenvolvido. O acrónimo representa: Funcionalidade, Usabilidade, Fiabilidade, Performance e Suportabilidade. O "+" inclui aspetos relacionados com o design, implementação, interface, físicos, entre outros que sejam relevantes ao projeto em questão (Eeles, 2001).

4.4.1 Usabilidade

A interface da aplicação deve ter um *layout* compreensível e fácil navegação, assim como incluir a tradução de termos nos idiomas Português (Portugal) e Inglês.

O *software* deverá suportar ferramentas de acessibilidade para auxiliar utilizadores com dificuldades visuais a identificar as máquinas, navegar nos menus da aplicação, efetuar carregamentos de saldo e comprar produtos.

Durante o desenvolvimento do projeto, deverá ser utilizada terminologia da área de negócio de *vending.*

4.4.2 Fiabilidade

O sistema deverá assegurar a segurança na transmissão e armazenamento de dados entre cliente e servidor, no momento da autenticação e durante as sessões de utilização da aplicação.

Em caso de erros ou indisponibilidade de quaisquer sistemas envolvidos no sistema, deverão ser apresentadas as devidas mensagens, de forma colocar o utilizador ao corrente de qualquer situação anómala.

4.4.3 Performance

Não existe um padrão de indústria que dite qual o tempo de resposta aceitável, sendo este valor variável, consoante o cenário.

Preferencialmente, o tempo de resposta do sistema não deverá superar os 200 milissegundos, de forma a manter a fluidez de operações, consultas de informação e navegação de menus.

4.4.4 Suportabilidade

O sistema deverá ser desenvolvido de modo a ser expandido ou adaptado. Adotar-se-á uma metodologia modular no seu desenvolvimento, de modo a reduzir a dependência entre os diferentes componentes.

O *software* deverá suportar o sistema operativo Android e iOS.

4.4.5 Restrições e Requisitos Adicionais

A nível de interface, o sistema deverá seguir os elementos característicos que identificam as máquinas de *vending* como os ícones, logótipos e terminologias, em conjunção com os requisitos impostos.

No que toca ao *hardware*, o *software* será suportado por duas máquinas virtuais, de um servidor de produção e um de testes, integrantes da infraestrutura do SI da FLUP.

4.5 Casos de Uso

Uma vez expostos os requisitos, são formalizados os casos de uso e atribuídas as funcionalidades da plataforma aos atores do sistema.

4.5.1 Atores

- **Utilizador FLUP** – utilizador registado na *Active Directory* existente. Não necessita de registo na plataforma, uma vez que a autenticação é gerida pelo controlador de domínio
- **Utilizador Externo** – este tipo de utilizador deve registar-se na plataforma, inserindo os seus dados de forma a criar uma conta única, de modo a utilizar as funcionalidades de *vending*

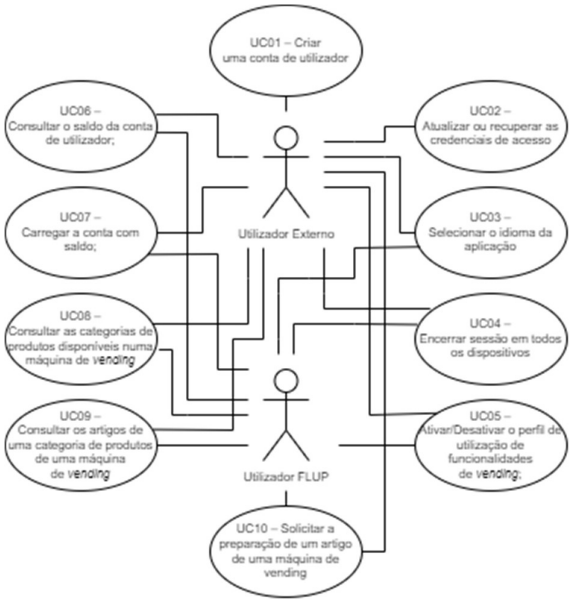

Figura 18 – Diagrama de Casos de Uso

Os casos de uso respetivos a cada ator do sistema estão representados na Figura 18.

4.5.2 UC01 – Criar uma conta de utilizador

Este caso de uso estará acessível apenas a utilizadores externos à FLUP, que não estão registados no controlador de domínio do SI. Esta ação envolve a introdução de informações pessoais para a criação de uma conta de utilizador na plataforma, independente da AD. Os detalhes do caso de uso são descritos na Tabela 10.

Tabela 10 – Detalhes do UC01

Ator(es)	Utilizador Externo
Descrição	Um utilizador externo regista-se na plataforma, introduzindo informações solicitadas pelo sistema
Pré-Condições	O utilizador deverá ser externo à FLUP (não deve possuir e-mail institucional) e não possuir uma conta na plataforma
Pós-Condições	O utilizador deverá ficar registado na plataforma
Cenário Principal	1 – O utilizador seleciona a opção de criação de conta 2 – O sistema apresenta um formulário de inscrição na plataforma 3 – O utilizador submete as informações de inscrição 4 – O sistema envia um código de confirmação para o endereço de correio eletrónico 5 – O utilizador insere o código de confirmação 6 – O sistema valida o código, cria a conta e informa o utilizador do sucesso da operação
Cenário Alternativo	3. a) As informações inseridas pelo utilizador não são válidas: o sistema informa o utilizador dos campos que necessitam de correção 5. a) O utilizador insere um código inválido: o sistema informa o utilizador e solicita o código correto

4.5.3 UC02 – Recuperar as credenciais de acesso

Apenas um utilizador externo poderá concluir a ação de alteração/recuperação das credenciais de acesso à plataforma, uma vez que este tipo de conta é gerido por este sistema, e não pela AD. Os detalhes do caso de uso são descritos na Tabela 11.

Tabela 11 - Detalhes do UC02

Ator(es)	Utilizador Externo
Descrição	Um utilizador solicita a recuperação das credenciais de acesso
Pré-Condições	O utilizador deverá ter uma conta na plataforma, com um e-mail não institucional
Pós-Condições	O utilizador deverá ter acesso à plataforma, com novas credenciais de acesso
Cenário Principal	1 – O utilizador seleciona a opção de recuperação de credenciais de acesso

	2 – O sistema envia um código de confirmação para o endereço de correio eletrónico 3 – O utilizador insere o código de confirmação 4 – O sistema solicita a introdução de uma nova palavra-passe 5 – O utilizador insere a nova palavra-chave 6 – O sistema informa o utilizador do sucesso da operação
Cenário Alternativo	3. a) O utilizador insere um código inválido: o sistema informa o utilizador e solicita o código correto 5. a) O utilizador insere uma palavra-passe inválida: o sistema informa o utilizador e solicita uma nova

4.5.4 UC03 – Selecionar o idioma da aplicação

Um utilizador poderá selecionar o idioma de todos os termos da aplicação móvel. Este idioma terá influência na tradução dos termos dos produtos que estão presentes nas máquinas de *vending.* Os detalhes do caso de uso são descritos na Tabela 12.

Tabela 12 - Detalhes do UC03

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador seleciona o idioma da aplicação
Pré-Condições	O utilizador deverá ter sessão iniciada. Devem existir pelo menos dois idiomas
Pós-Condições	Todos os termos da aplicação deverão estar traduzidos no idioma selecionado pelo utilizador
Cenário Principal	1 – O utilizador seleciona a opção de seleção do idioma 2 – O sistema apresenta uma lista de idiomas disponíveis 3 – O utilizador seleciona o idioma preferido 4 – O sistema apresenta a aplicação traduzida no idioma selecionado
Cenário Alternativo	Sem cenário alternativo

4.5.5 UC04 – Encerrar sessão em todos os dispositivos

Um utilizador deverá ter, como função de segurança, a possibilidade de encerrar sessão em todos os dispositivos anteriormente ligados à sua conta. Esta ação reduz a possibilidade furto de informações, em caso extravio de equipamentos. Os detalhes do caso de uso são descritos na Tabela 13.

Tabela 13 - Detalhes do UC04

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador encerra sessão em todos os dispositivos anteriormente ligados à sua conta
Pré-Condições	O utilizador deverá ter sessão iniciada
Pós-Condições	As sessões ativas em todos os dispositivos ligados à conta de utilizador deverão ter sido terminadas

Cenário Principal	1 – O utilizador seleciona a opção de término de sessão em todos os dispositivos 2 – O sistema pede confirmação 3 – O utilizador confirma 4 – O sistema termina sessão em todos os dispositivos
Cenário Alternativo	Sem cenário alternativo

4.5.6 UC05 – Ativar/Desativar o perfil de utilização de funcionalidades de *vending*

O utilizador deverá aceitar os termos e condições de utilização do serviço para usufruir de todas as funções de *vending* disponibilizadas pela aplicação. Para tal, deverá conseguir ativar ou desativar o seu perfil de *vending*. Os detalhes do caso de uso são descritos na Tabela 14.

Tabela 14 - Detalhes do UC05

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador altera o estado do seu perfil de *vending*
Pré-Condições	O utilizador deverá ter sessão iniciada
Pós-Condições	O estado do perfil de utilização deverá ter sido alterado
Cenário Principal	1 – O utilizador seleciona a opção de alteração do estado do perfil de *vending* 2 – O sistema altera o estado do perfil e mostra ao utilizador o estado atual
Cenário Alternativo	Sem cenário alternativo

4.5.7 UC06 – Consultar o saldo da conta de utilizador

O saldo dos utilizadores é gerido por um *Web Service* que se encontra ativo na infraestrutura na faculdade, e é utilizado em diferentes contextos de pagamento (posteriormente referidos). Os detalhes do caso de uso são descritos na Tabela 15.

Tabela 15 - Detalhes do UC06

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador deverá conseguir visualizar o saldo associado à sua conta
Pré-Condições	O utilizador deverá ter sessão iniciada na plataforma e o perfil de *vending* ativo
Pós-Condições	O utilizador deverá ter acesso ao seu saldo em conta
Cenário Principal	1 – O utilizador seleciona a opção de consulta de saldo 2 – O sistema mostra o saldo disponível
Cenário Alternativo	2. a) O utilizador não tem o perfil de *vending* ativo: o sistema informa o utilizador que deve ativar o mesmo para fazer a consulta de saldo

4.5.8 UC07 – Carregar a conta com saldo

Um utilizador deverá ter um saldo associado à sua conta de utilizador, que será utilizado nos serviços de *vending* disponibilizados. Os detalhes do caso de uso são descritos na Tabela 16.

Tabela 16 - Detalhes do UC07

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador deverá conseguir carregar a sua conta com saldo através dos métodos de pagamentos disponibilizados na plataforma
Pré-Condições	O utilizador deverá ter sessão iniciada na plataforma e o perfil de *vending* ativo
Pós-Condições	O saldo do utilizador deverá ter sido atualizado, de acordo com o montante selecionado
Cenário Principal	1 - O utilizador seleciona a opção de carregamento de conta 2 – O sistema mostra os métodos de pagamento disponíveis 3 – O utilizador seleciona o método de pagamento preferido 4 – O sistema solicita as informações de pagamento 5 – O utilizador submete as informações de pagamento 6 – O sistema processa o pagamento e informa o utilizador do sucesso da operação
Cenário Alternativo	6. a) O método de pagamento exige uma aceitação externa da parte do utilizador: o sistema indica as instruções para continuar o processo e aguarda a ação do utilizador b) O processo de pagamento falha: o sistema informa o utilizador da ocorrência

4.5.9 UC08 – Consultar as categorias de artigos disponíveis numa máquina de *vending*

Um utilizador deverá conseguir visualizar a lista de categorias de produtos disponíveis numa máquina de *vending*, devidamente identificada, acompanhadas de um título e ilustração. Os detalhes do caso de uso são descritos na Tabela 17.

Tabela 17 - Detalhes do UC08

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador deverá conseguir visualizar as categorias de produtos de uma máquina de *vending*
Pré-Condições	O utilizador deverá ter sessão iniciada na plataforma, perfil de *vending* ativo e acesso a uma etiqueta NFC ou código QR de uma máquina de *vending*
Pós-Condições	O utilizador deverá ter conhecimento das categorias de produtos existentes
Cenário Principal	1 – O utilizador faz *scan* de uma etiqueta NFC ou de um código QR, afixado numa máquina de *vending* integrada no sistema 2 – O sistema mostra os nomes das categorias de produtos, acompanhados de uma imagem ilustrativa

Cenário Alternativo	Sem cenário alternativo

4.5.10 UC09 – Consultar os artigos de uma categoria de produtos de uma máquina de *vending*

Uma vez apresentadas as categorias de produtos, o utilizador deverá conseguir selecionar uma delas e visualizar os artigos correspondentes à sua pesquisa, acompanhados de um título, ilustração e preço de venda. Os detalhes do caso de uso são descritos na Tabela 18.

Tabela 18 - Detalhes do UC09

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador deverá conseguir visualizar os produtos presentes numa máquina de *vending*
Pré-Condições	O utilizador deverá ter sessão iniciada na plataforma, perfil de *vending* ativo e completado o **UC08**
Pós-Condições	O utilizador deverá ter conhecimento dos produtos disponíveis para compra
Cenário Principal	1 – O utilizador seleciona uma categoria de produtos 2 – O sistema mostra os nomes dos produtos disponíveis da mesma categoria, acompanhados de uma imagem ilustrativa e o respetivo preço
Cenário Alternativo	Sem cenário alternativo

4.5.11 UC10 – Solicitar a preparação de um artigo de uma máquina de *vending*

Após a seleção de um artigo, o utilizador deverá conseguir solicitar a preparação do mesmo. Em determinados artigos (como algumas bebidas), deverá ser possível optar por incluir ou não um copo e controlar a quantidade de açúcar adicionado.

A plataforma deverá comunicar com a máquina de *vending* correspondente para efetuar a ação e deduzir o valor do artigo do saldo de conta. As máquinas compatíveis com o sistema disponibilizam uma API que permite a comunicação com as mesmas, através de pedidos HTTPS. Os detalhes do caso de uso são descritos na Tabela 19.

Tabela 19 - Detalhes do UC10

Ator(es)	Utilizador FLUP, Utilizador Externo
Descrição	Um utilizador solicita a preparação de um artigo presente numa máquina de *vending*, através da aplicação
Pré-Condições	O utilizador deverá ter sessão iniciada na plataforma, perfil de *vending* ativo, completado o **UC09** e ter saldo disponível na sua conta para cobrir o valor do artigo selecionado
Pós-Condições	A máquina deverá ter dispensado o produto e atualizado o saldo de conta de utilizador
Cenário Principal	1 – O utilizador seleciona o produto que pretende

	2 – O sistema comunica com a máquina de *vending*, dispensa o artigo, atualiza o saldo do utilizador, e informa-o do sucesso da operação
Cenário Alternativo	1. a) O utilizador personaliza o artigo (quando disponível) 2. a) O sistema deteta que o utilizador não possui saldo suficiente para prosseguir com a compra: o sistema informa o utilizador da ocorrência e termina o processo b) O sistema não consegue contactar com a API da máquina de *vending* ou é devolvido um erro desconhecido: o sistema informa o utilizador da ocorrência e termina o processo c) A API de *vending* informa o sistema que o produto selecionado não se encontra disponível: o sistema informa o utilizador da ocorrência

4.6 Conclusão

Este capítulo permite ao leitor compreender os requisitos funcionais e não funcionais impostos pelo cliente e os casos de uso associados ao projeto, assim como os conceitos ligados ao negócio e interligações entre as suas partes. São também abordadas, com detalhe, as funcionalidades que serão implementadas, as suas condições e interações utilizador-máquina.

5 Desenho da Solução

No atual capítulo, é abordada a arquitetura global do sistema, a organização dos vários componentes, assim como métodos e tecnologias utilizados na concretização da estrutura do projeto.

5.1 Arquitetura do Sistema

O sistema é baseado num modelo cliente-servidor: o servidor será suportado por uma ou mais máquinas consoante as necessidades de processamento, memória, segurança ou manutenção; os clientes *mobile* comunicam com o(s) servidor(es) através do protocolo HTTPS, via uma API.

5.1.1 Padrão *Model-View-Controller* (*MVC*)

Durante o desenvolvimento, adota-se o padrão *Model-View-Controller*, que consiste na organização do sistema em módulos independentes, repartindo responsabilidades: as regras do negócio, representadas através de conjunto de classes numa arquitetura orientada a objetos (*Model*); a interface gráfica com que o utilizador irá interagir (*View*); o tratamento dos *inputs* que são enviados a partir da interface gráfica, com base na lógica de negócio (*Controller*). O padrão é descrito através da Figura 19 (Deacon, 2013)

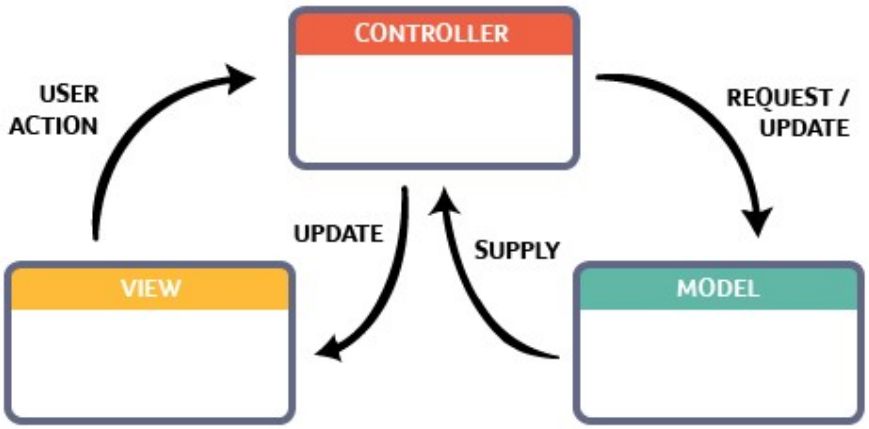

Figura 19 – Funcionamento do Padrão Model-View-Controller (Madooei, s.d.)

A aplicação deste padrão ao projeto permite distinguir a parte lógica da de apresentação, ao dispor diferentes tarefas por módulos. Desta forma, cada um é responsável por solucionar uma parte do problema. Esta distribuição de tarefas torna a arquitetura do sistema mais legível e menos propensa a falhas.

5.1.2 Serviço de *Active Directory*

Atualmente, a gestão de autenticação de utilizadores da FLUP é gerida por um serviço de controlo de domínio *Active Directory*, baseado no protocolo LDAP (*Lightweight Directory Access Protocol*). Este é suportado por servidores Microsoft dedicados, utilizados por várias aplicações pertencentes à infraestrutura, nomeadamente o portal do SI, o serviço de reprografia, de correio eletrónico, no acesso aos computadores das salas de aula e às redes sem fios.

5.1.3 Gestão de conta e saldo dos utilizadores

A FLUP dispõe de um serviço de gestão de saldo, associado ao cartão institucional dos utilizadores de rede. Este "porta-moedas de rede" permitiu a integração de funcionalidades adicionais, facilitando o acesso a um conjunto diversificado de serviços pagos, evitando a circulação de dinheiro, nomeadamente na utilização dos moedeiros da instituição como forma de carregamento, e minimizando falhas humanas nos trocos e fechos de caixa. São exemplo destas funcionalidades o serviço *self-service* de impressão, o pagamento do parque de estacionamento e de multas da biblioteca.

O serviço é disponibilizado em forma de *Web Service*, integrável com o sistema a ser desenvolvido, que oferece funções essenciais para a consulta do saldo, transações dos utilizadores, bem como atualização de saldo, resultante de pagamentos de serviços de *vending*.

5.1.4 API de interação com as máquinas de *vending*

A comunicação com as máquinas é feita através de uma API, disponibilizada pela empresa de gestão e abastecimento dos equipamentos de *vending* das instalações. Esta aplicação permite consultar informações relativamente ao estado das máquinas e seus conteúdos, assim como executar pedidos de preparação/dispensa de artigos. O fluxo de processamento do pedido de dispensa é gerido pela aplicação consumidora da API, assim como o tratamento de cenários alternativos, como erros, indisponibilidade de *hardware* ou produtos.

5.1.5 Protocolo de pagamento 3-D Secure

O 3-D Secure foi desenvolvido pela empresa Visa e MasterCard com vista a aumentar o nível de segurança nos pagamentos com cartão, através da autenticação e verificação do titular, ao reduzir as tentativas de fraude do tipo *Card-Not-Present* (CNP), que consistem na tentativa de transação fraudulenta sem que haja posse do cartão físico (Kagan, 2022).

O protocolo tenta combater este tipo de fraude através do pedido de credenciais que serão introduzidas pelo dono do cartão, que o identificam. O processo de autenticação é independente do ambiente disponibilizado pelo comerciante dos bens ou serviços (Bouch, 2011).

O esquema do processo e seus componentes é descrita através da ilustração da Figura 20.

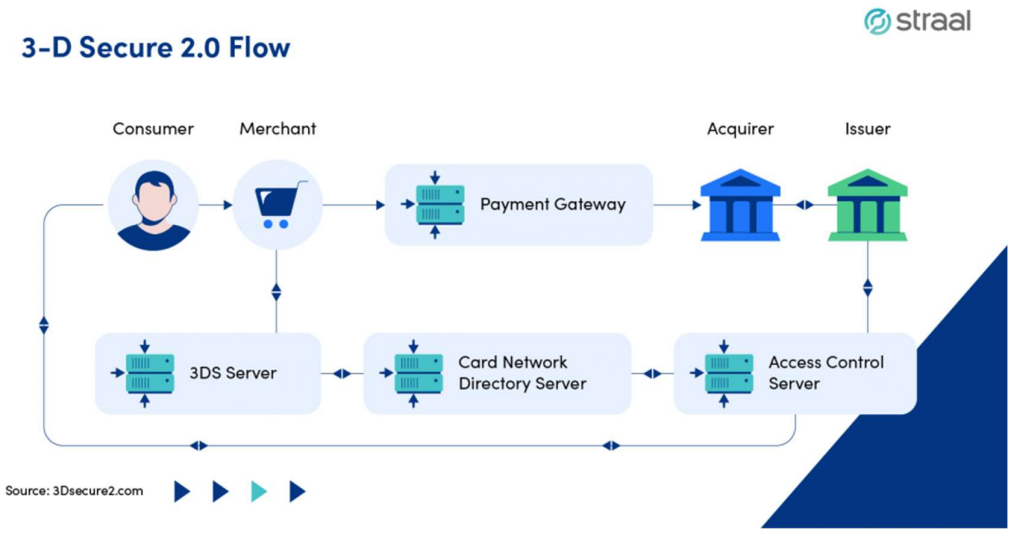

Figura 20 – Funcionamento do 3-D Secure (Mizinska, 2021)

O nome do protocolo deriva dos três domínios envolvidos no processo (Bouch, 2011):

- O **emissor** - o banco que emite o cartão
- O **adquirente** - o comerciante recetor do dinheiro e o respetivo banco

- O **intermediário** - a plataforma online que disponibiliza as funcionalidades 3DS

Este será o método utilizado nas transações de carregamento de cartões FLUP.

5.2 Diagrama de Componentes do Sistema

Os componentes do sistema são descritos no diagrama da Figura 21, assim como as respetivas relações, de modo a permitir discernir a interação entre os diferentes módulos, de acordo com os devidos padrões e natureza do projeto.

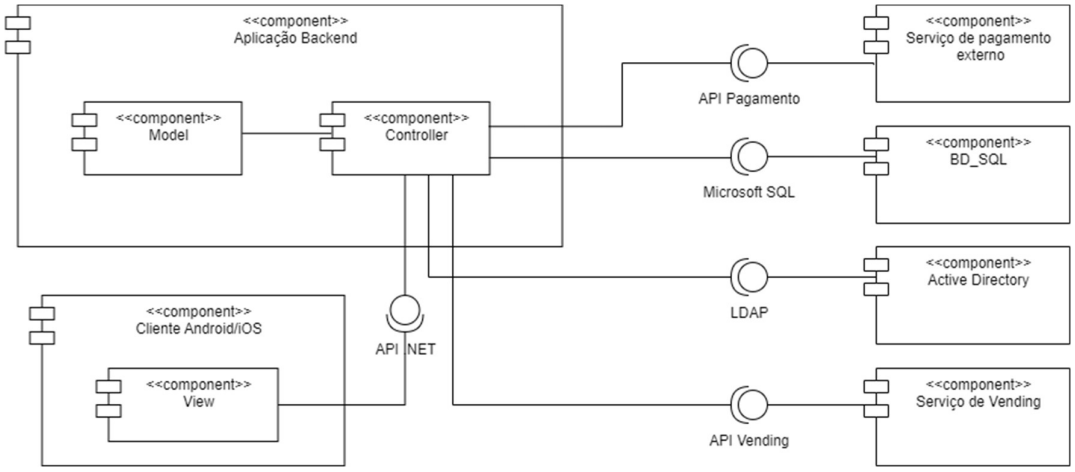

Figura 21 - Diagrama de Componentes do Sistema

5.2.1 Frontend

O *frontend* deverá funcionar como um módulo independente do *backend*, capaz de fazer pedidos HTTPS e tratar todos os inputs e apresentação de informação associada aos casos de uso do projeto. Neste contexto, a *View* estará integrada na aplicação cliente.

5.2.2 Backend

O *backend* deverá ser composto por uma API, responsável pelo tratamento dos pedidos originários do *frontend*, independente dos outros componentes. Esta aplicação tem como grande vantagem eliminar a necessidade do *frontend* se ligar diretamente a um conjunto de serviços essenciais ao sistema. Desta forma, a API será o único componente responsável por contruir uma ponte de ligação entre outras API, *Web Services* ou aplicações externas, através do componente *Controller*. O *backend* deve consumir e tratar dados provenientes do(s) serviço(s) de pagamento, da base de dados alocada ao sistema e do servidor de *Active Directory*.

5.3 Diagrama de Implantação do Sistema

A Figura 22 ilustra o diagrama de implantação do sistema, que permite visualizar a disposição dos vários módulos do sistema numa infraestrutura física, e respetivas interligações.

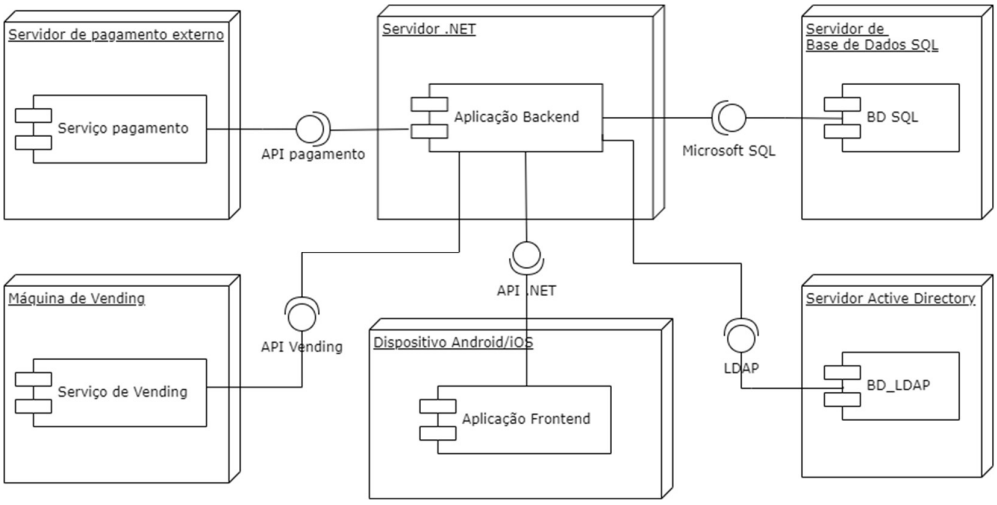

Figura 22 – Diagrama de Implantação do Sistema

5.4 Diagramas de Sequência do Sistema

Os diagramas de sequência do sistema permitem a compreensão das interações entre os diferentes componentes de forma mais granular. No sistema descrito, são explicitados os pedidos cliente-servidor de cada caso de uso.

5.4.1 UC01 - Criar uma conta de utilizador

Como anteriormente referido, esta ação está reservada para utilizadores externos. O sistema necessita de registá-los no *Web Service* de pagamento de modo a, posteriormente, gerir o seu saldo em conta. Adicionalmente, o sistema cria um utilizador na sua base de dados, com informações relacionadas com o contexto de *vending*. O respetivo diagrama de sequência está representado na Figura 23.

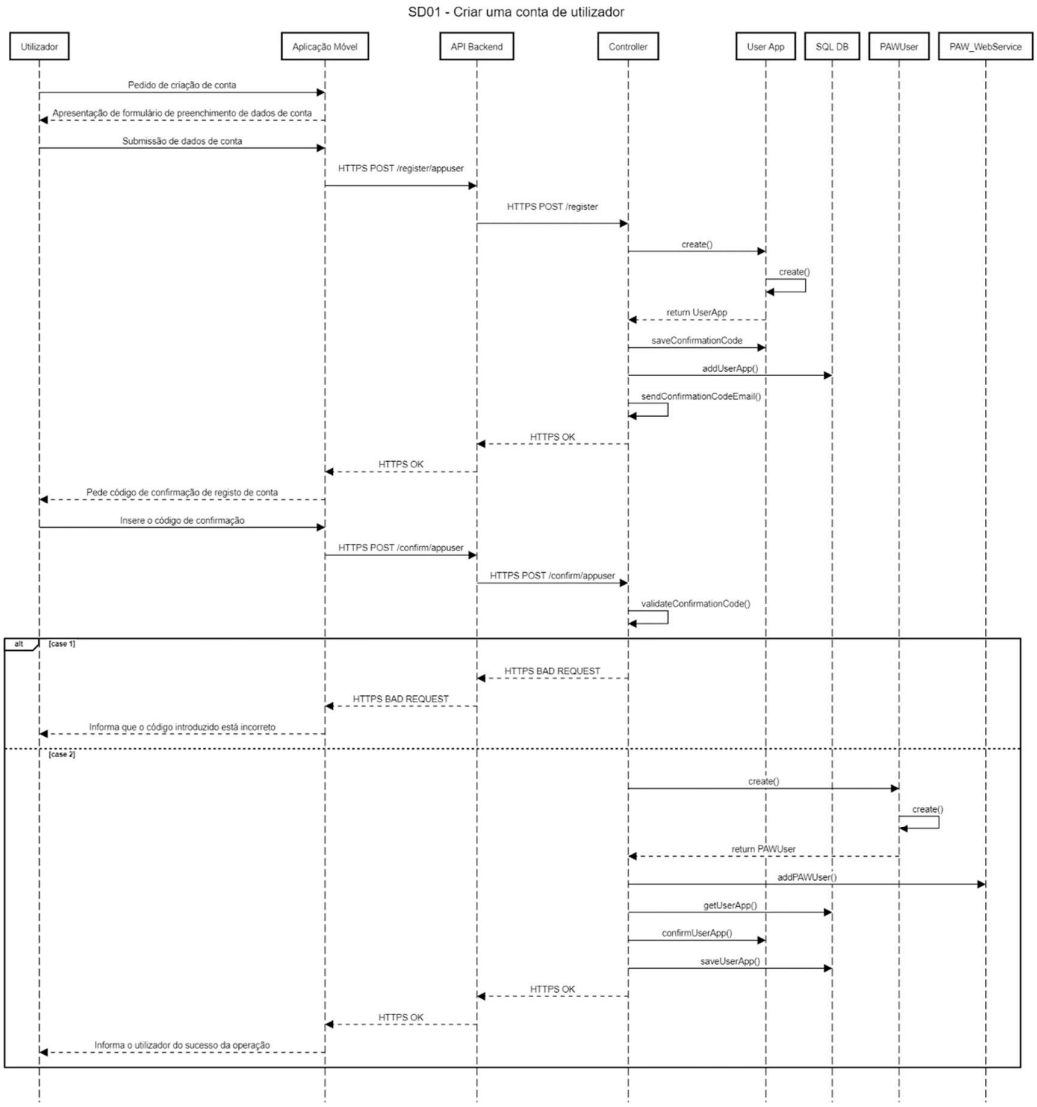

Figura 23 – Diagrama de Sequência do UC01

5.4.2 UC02 – Atualizar ou recuperar as credenciais de acesso

O sistema executa esta ação, independentemente da AD, uma vez que os utilizadores externos estão registados na base de dados da aplicação de *vending*. O respetivo diagrama de sequência está representado na Figura 24.

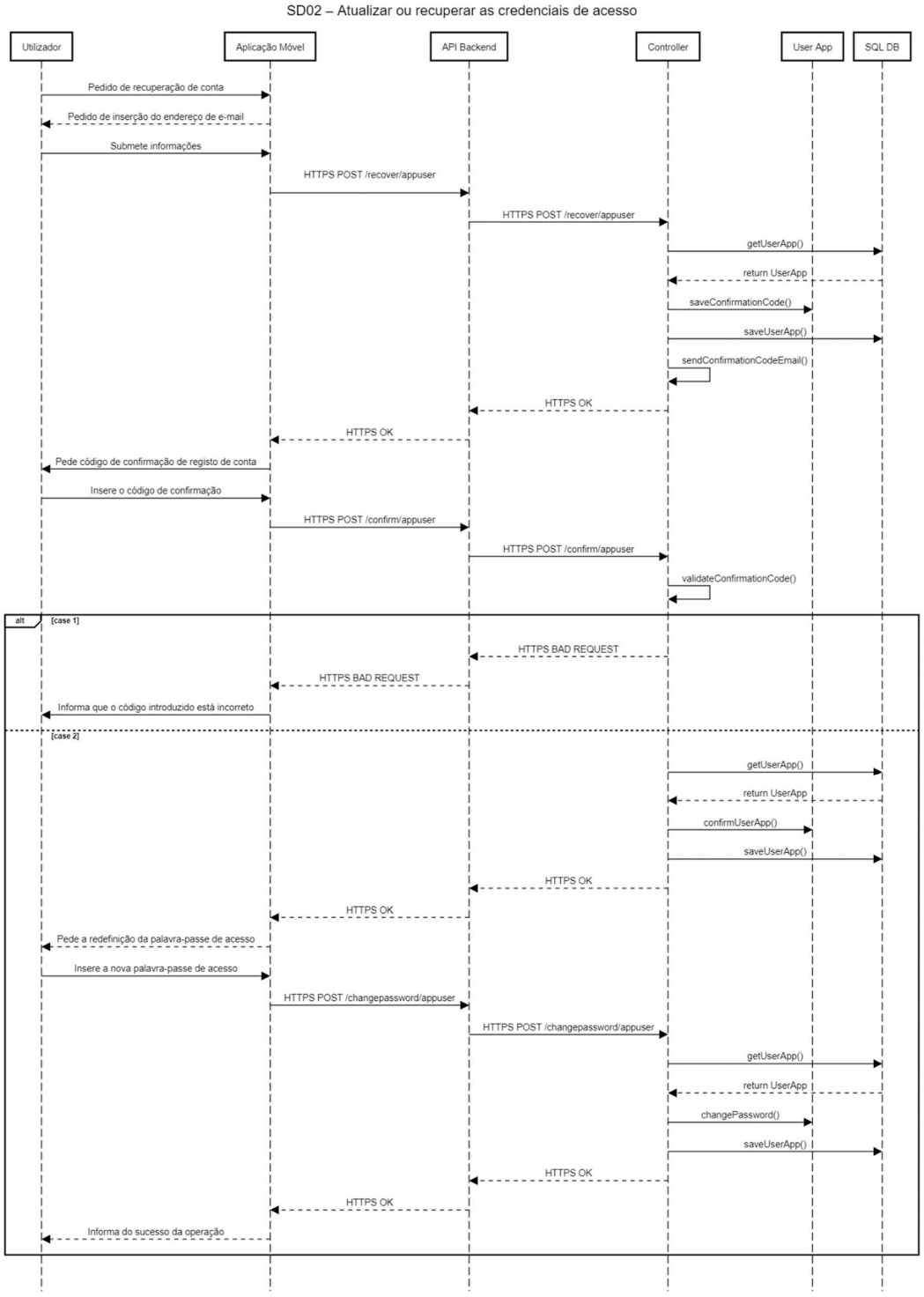

Figura 24 - Diagrama de Sequência do UC02

5.4.3 UC03 – Selecionar o idioma da aplicação

Uma vez que o idioma dos termos de *frontend* é gerido pela aplicação móvel, não é relevante representar este caso de uso em formato de diagrama de sequência. No entanto, o *frontend* depende do *backend* para mostrar os termos relativos às categorias e seus produtos, traduzidos convenientemente. Estes não são geridos pela aplicação, mas sim, em *backoffice*, de modo a coincidirem com os artigos presentes nas máquinas, que podem sofrer alterações a qualquer momento. Por esta razão, a ação está integrada no diagrama de sequência dos casos de uso **UC08** e **UC09**.

5.4.4 UC04 – Encerrar sessão em todos os dispositivos

Esta funcionalidade consiste em invalidar o(s) *token(s)* de acesso armazenados no sistema, registando-os numa "lista negra". Quando um pedido é enviado para o sistema, essa lista é verificada e, se o *token* utilizado for detetado, a sessão é automaticamente cancelada. O respetivo diagrama de sequência está representado na Figura 25.

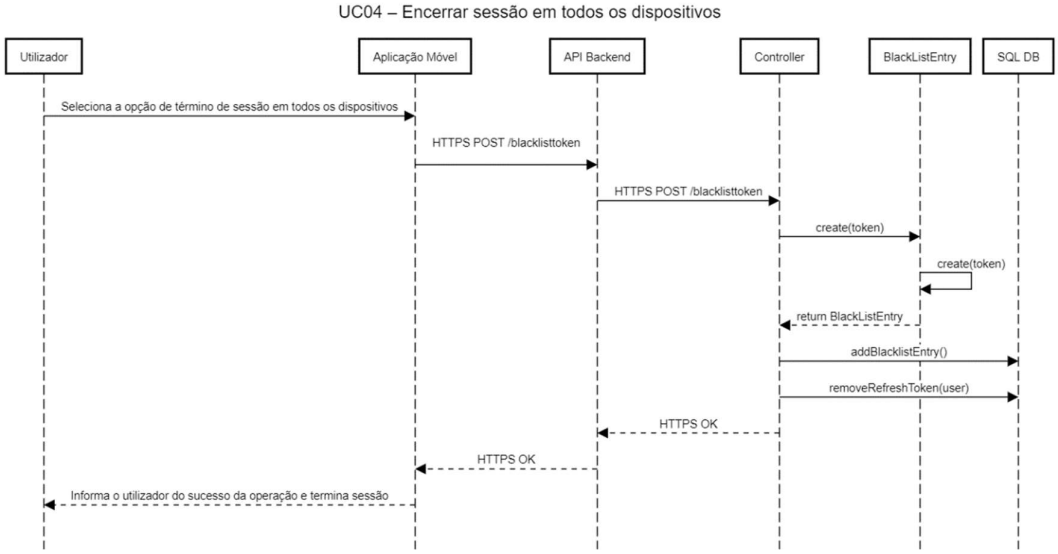

Figura 25 - Diagrama de Sequência do UC04

5.4.5 UC05 – Ativar/Desativar o perfil de utilização de funcionalidades de *vending*

Esta funcionalidade tem por base a aceitação dos termos e condições de utilização da plataforma, e, consequentemente, a ativação de um perfil de *vending*, gerido pelo *Web Service* de pagamentos. O respetivo diagrama de sequência está representado na Figura 26.

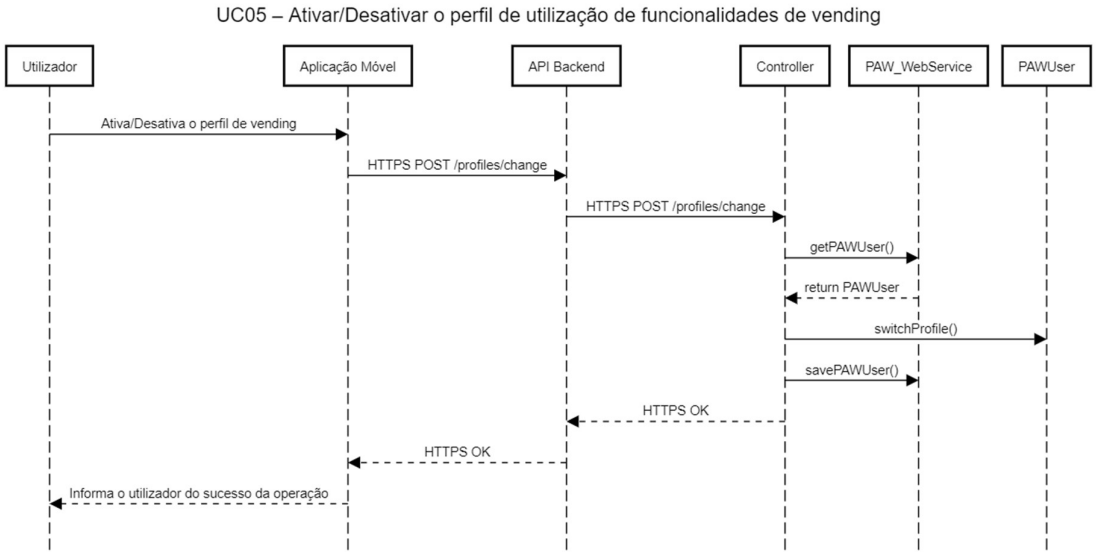

Figura 26 - Diagrama de Sequência do UC05

5.4.6 UC06 – Consultar o saldo da conta de utilizador

Esta ação tem por base um pedido de consulta de saldo do utilizador anteriormente registado no *Web Service* de pagamento. O respetivo diagrama de sequência está representado na Figura 27.

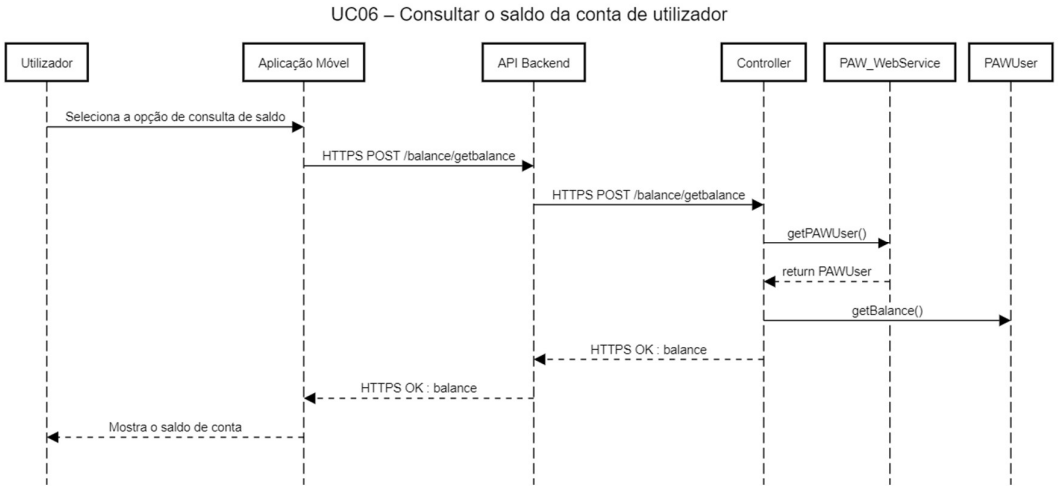

Figura 27 - Diagrama de Sequência do UC06

5.4.7 UC07 – Carregar a conta com saldo

O carregamento de saldo inicia-se com um pedido de criação de pagamento ao respetivo serviço do método selecionado. Após a receção do pedido, é enviada uma resposta através de

um mecanismo de *WebHooks* para um serviço da FLUP, que se encontra à escuta e armazena as informações recebidas.

O sistema aguarda até que a informação do pedido esteja disponível. Se o processo de pagamento tiver tido sucesso, o sistema atualiza o saldo de utilizador através do *Web Service* de pagamentos da FLUP.

O diagrama da Figura 28 - Diagrama de Sequência do UC07 apresentado descreve o processo utilizado para o método de pagamento MBWay.

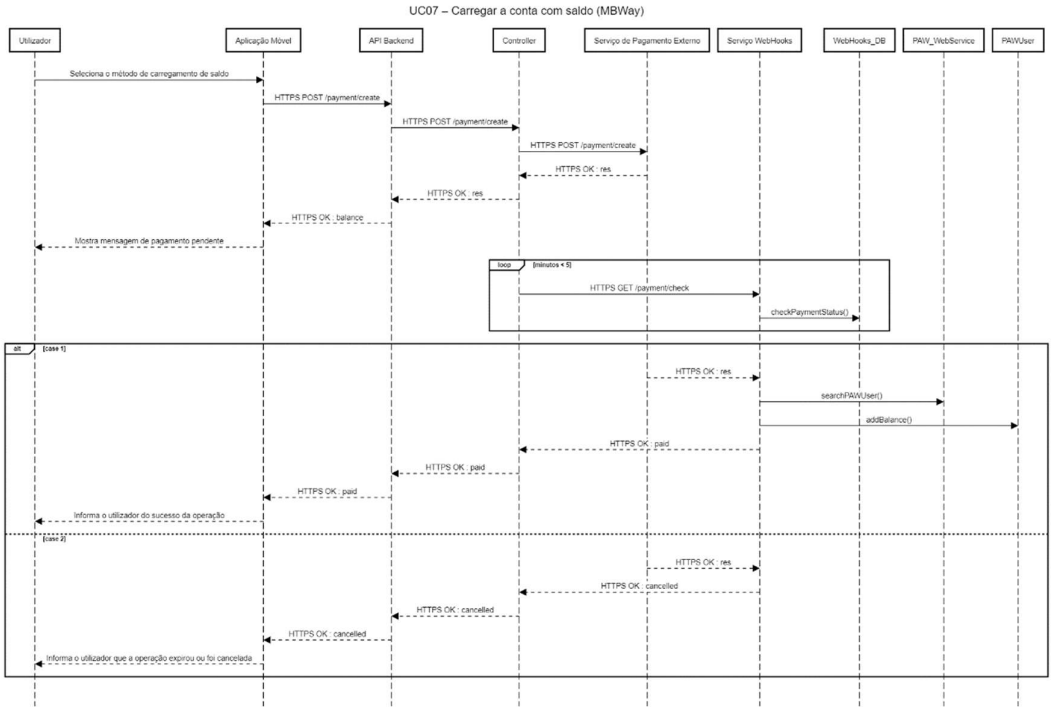

Figura 28 - Diagrama de Sequência do UC07

5.4.8 UC08 – Consultar as categorias de artigos disponíveis numa máquina de *vending*

A aplicação *frontend* solicita ao *backend* as informações de categorias de artigos correspondentes à máquina previamente identificada. O respetivo diagrama de sequência está representado na Figura 29.

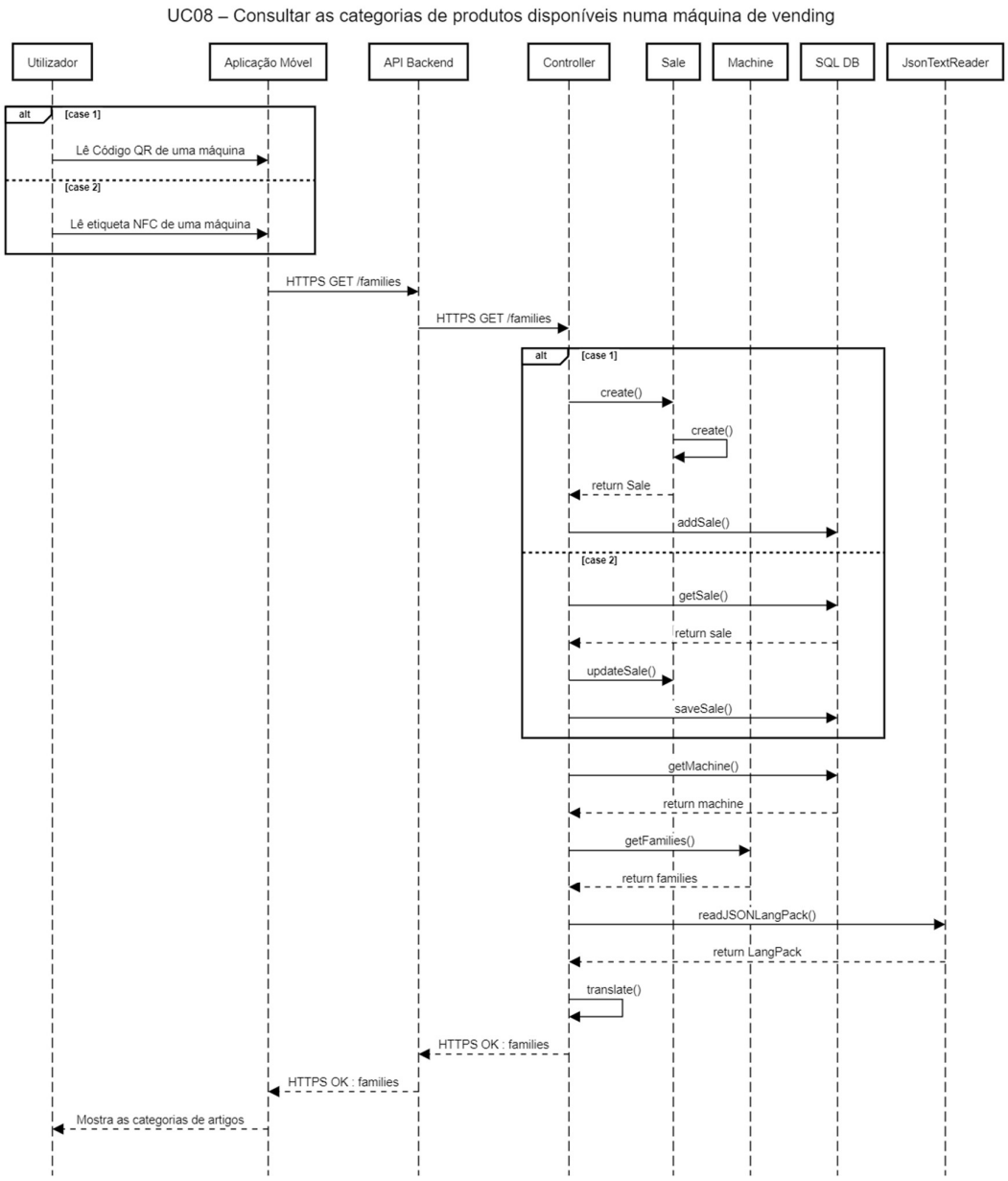

Figura 29 - Diagrama de Sequência do UC08

5.4.9 UC09 – Consultar os artigos de uma categoria de produtos de uma máquina de *vending*

A aplicação *frontend* solicita ao *backend* as informações dos artigos correspondentes à categoria previamente selecionada. O respetivo diagrama de sequência está representado na Figura 30.

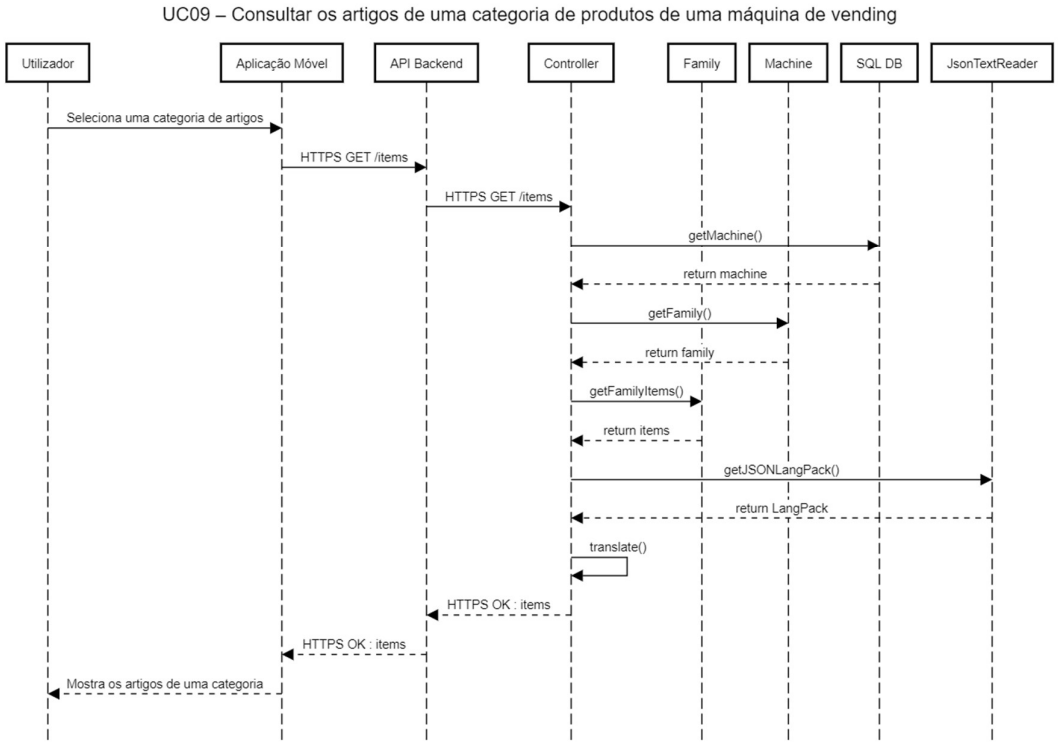

Figura 30 - Diagrama de Sequência do UC09

5.4.10 UC10 – Solicitar a preparação de um artigo de uma máquina de *vending*

Este processo consiste em localizar a produto selecionado pelo utilizador e enviar um pedido de dispensa à API de *vending*, controlada pela máquina anteriormente identificada. O sistema aguarda pela resposta da API e, de acordo com o resultado, atualiza as informações do utilizador e do registo de venda. O respetivo diagrama de sequência está representado na Figura 31.

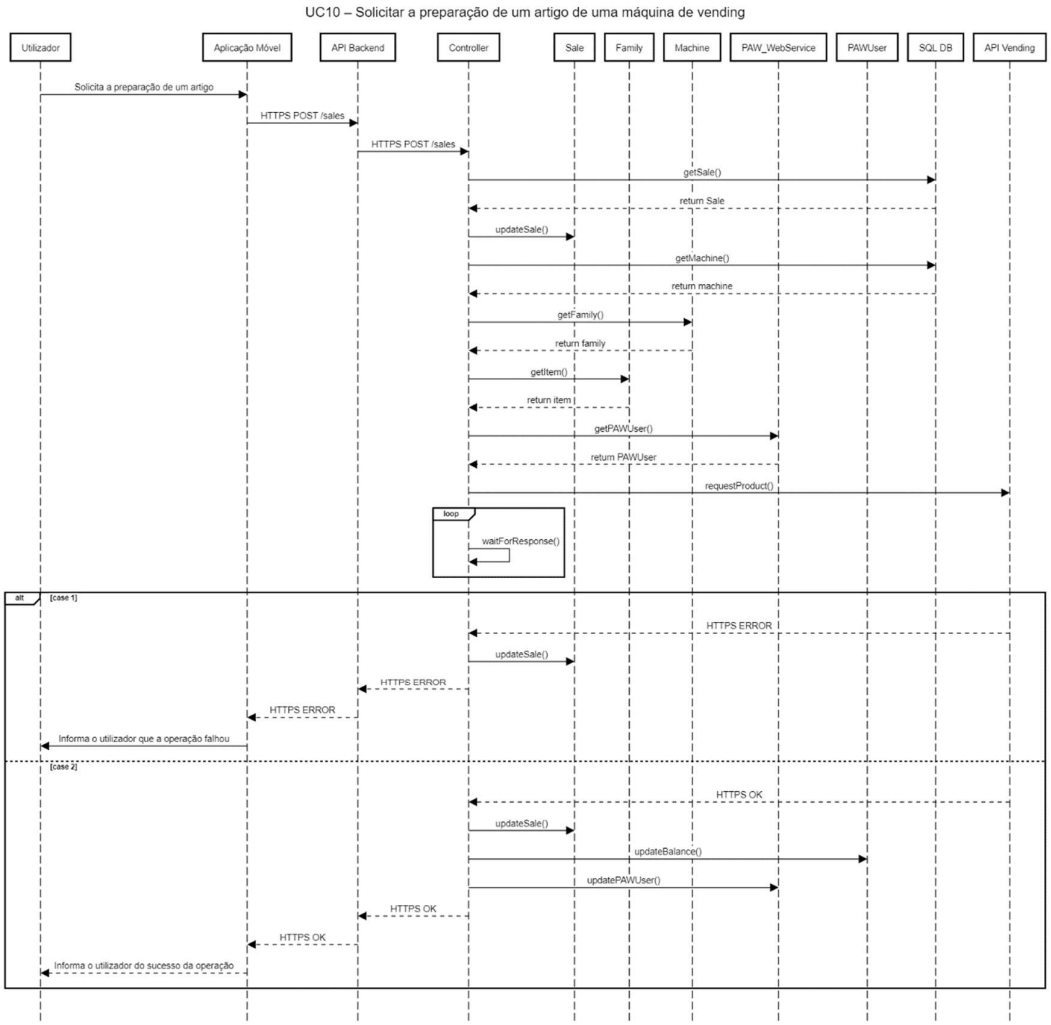

Figura 31 - Diagrama de Sequência do UC10

5.5 Conclusões

Este capítulo permite ao leitor compreender, de forma mais granular, a arquitetura do sistema descrito no documento.

A arquitetura baseia-se no modelo *Model-View-Controller* e cliente-servidor, onde o *frontend* é derivado de uma aplicação móvel e o *backend* de uma API responsável por comunicar com um conjunto de serviços internos e externos.

É relatado o funcionamento dos métodos de autenticação dos diferentes utilizadores da plataforma, bem como o tratamento do saldo do utilizador e forma de comunicação com as máquinas de *vending*.

Finalmente, são descritos os casos de uso em formato de Diagrama de Sequência, de modo a que o leitor entenda, detalhadamente, a interação lógica entre os diferentes objetos e entidades do sistema, de forma sequencial.

6 Implementação da Solução

O presente capítulo aborda a implementação do sistema descrito ao longo dos capítulos anteriores.

É feita uma exposição dos métodos utilizados na conceção do software, acompanhados de extratos de código em formato de linguagem de programação e detalhes explicativos das técnicas aplicadas.

6.1 Descrição da implementação

O sistema consiste no desenvolvimento de uma aplicação móvel, com recurso à *framework* Flutter, com o intuito principal de unificar o processo de desenvolvimento multiplataforma, e posteriormente, facilitar o *deployment* do *software*, em conjunto com a linguagem de programação Dart.

O *backend* deverá ser desenvolvido recorrendo à .NET Entity Framework, em linguagem de programação C#, com base num modelo de dados orientado a objetos. A *framework* disponibiliza a implementação do modelo MVC, que será utilizada como base do projeto.

A comunicação entre o cliente (aplicação móvel) e servidor (aplicação *backend*) é feita através de pedidos que seguem o protocolo HTTPS, sendo utilizado o JSON como formato de troca de dados.

O armazenamento é suportado por um servidor de base de dados SQL e gerido através da *framework* .NET.

6.2 Metodologia de desenvolvimento

O processo de desenvolvimento do software inicia-se pela criação do Modelo de Dados com base nos conceitos de negócio do contexto do projeto. As entidades abstratas do modelo serão

representadas através classes de objetos, a que têm associados um conjunto de funções e responsabilidades.

Seguidamente, procede-se à implementação das funcionalidades descritas em cada caso de uso, sendo que, será desenvolvida a API (*backend*), seguida da aplicação móvel (*frontend*).

Finalmente, serão executados testes à solução desenvolvida, de modo a garantir a qualidade do produto final e o cumprimento dos requisitos impostos pelo cliente.

6.2.1 Modelo de Dados

A criação do modelo de dados consiste na tradução dos conceitos de negócio descritos no modelo de domínio em classes de objetos. A *framework* .NET executa esta tarefa, ao criar um contexto de base de dados que acomode todas as informações de cada entidade.

Os próximos pontos relatam a implementação, em formato de código C#, das classes do modelo de domínio.

Primeiramente, são definidas as estruturas de dados para os dois tipos de utilizadores do sistema. Ambos herdam as propriedades de utilizador da classe **IdentityUser**, membro da *framework*. A classe **UserApp** deriva diretamente da mesma, contendo o primeiro e último nome do utilizador, bem como o código de confirmação utilizado no registo e na recuperação de credenciais de acesso, como é possível verificar no excerto 1.

```
public class UserApp : IdentityUser
   {
       public string Name { get; set; }
       public string LastName { get; set; }

       public string ConfirmationCode { get; set; }

    (…)
```

Excerto 1 – Classe e atributos UserApp

A classe **UserUP** deriva da **UserApp**, ou seja, contém os atributos anteriormente descritos e, adicionalmente, o número mecanográfico FLUP do utilizador, como é possível verificar no excerto 2.

```
  public class UserUP : UserApp
  {
      public string UPNumber { get; set; }

       (…)
```

Excerto 2 - Classe e atributos UserUP

O registo de uma sessão (utilização da aplicação por um utilizador) é representado através de um objeto da classe **Session** (excerto 3).

```
[Serializable]
    public class Session
    {
        [JsonIgnore]
        public int SessionId {get; set;}
        public string UserName { get; set; }
        public string Signature { get; set; }
        public string ClientIP { get; set; }
        public string AppVersion { get; set; }
        public DateTime StartDate { get; set; }
        public DateTime? EndDate { get; set; }

     (…)
```

Excerto 3 - Classe e atributos Session

Esta estrutura de dados permite armazenar a identificação do utilizador, o seu endereço IP (para eventual necessidade de tratamento de erros), a data do *login* e *logout* (se existente) e a versão da aplicação. Estas informações são também relevantes para a criação e compreensão de estatísticas de utilização do serviço.

As máquinas de *vending* são representadas em forma de objeto da classe **Machine**, que contém o nome da máquina, a identificação do local em que está instalada, um código QR e uma lista de etiquetas NFC que identificam a máquina, as famílias de artigos que a máquina disponibiliza e um identificador para a comunicação com a API de vending, como é possível verificar no excerto 4.

```
    [Serializable]
    public class Machine
    {
        [JsonIgnore]
        public int MachineId { get; set; }
        public string MachineName { get; set; }
        public string MachineLocal { get; set; }
        public string MachineLocalId { get; set; }
        public string QRCode { get; set; }
        public List<NFCTag> NFCTags { get; set; }
        public List<Family> Families { get; set; }
        public string DeviceId { get; set; }
     (…)
```

Excerto 4 - Classe e atributos Machine

Uma família de artigos é representada por um objeto da classe **Family** (excerto 5), que contém identificadores de nome e de uma imagem ilustrativa, um período com início e fim (correspondente à sua disponibilidade), um número inteiro que define a sua ordem na lista (prioridade), assim como a lista de artigos da família.

Em contexto de base de dados, existe um relacionamento entre a tabela **Family** e **Machine**, sendo utilizada uma chave estrangeira para a identificação do objeto correspondente:

```
[Serializable]
    public class Family
    {
        [JsonIgnore]
        public int FamilyId { get; set; }
        public List<Item> FamilyItems { get; set; }
        public string ImageId { get; set; }
        public string FamilyNameId { get; set; }
        public DateTime StartDate { get; set; }
        public DateTime? EndDate { get; set; }
        public int Priority { get; set; }

        [JsonIgnore]
        [ForeignKey("Machine")]
        public int MachineId { get; set; }
        public virtual Machine Machine { get; set; }
 (…)
```

Excerto 5 - Classe e atributos Family

Um artigo físico de uma máquina de *vending* é representado por um objeto da classe **Item** (excerto 6), que contém, tal como as famílias de artigos, os identificadores de nome e de uma imagem ilustrativa, o valor comercial, um período com início e fim (correspondente à sua disponibilidade), a prioridade na lista de artigos e o código do produto para interação com a API da máquina.

Existe também uma chave estrangeira para a identificação do objeto da classe **Family** (família de produtos), a que o artigo pertence, na tabela de base de dados.

```
[Serializable]
    public class Item
    {
        [JsonIgnore]
        public int ItemId { get; set; }
        public float Price { get; set; }
        public string ImageId { get; set; }
        public string ItemNameId { get; set; }
        public DateTime StartDate { get; set; }
        public DateTime? EndDate { get; set; }
        public int Priority { get; set; }
        public string ProductCode { get; set; }

        [JsonIgnore]
        [ForeignKey("Family")]
        public int FamilyId { get; set; }
        public virtual Family Family { get; set; }
(…)
```

Excerto 6 - Classe e atributos Item

O registo de uma venda é representado por um objeto da classe **Sale** (excerto 7), que armazena identificadores do utilizador, da máquina, da compra, da sessão e registos do ato de pagamento.

```
[Serializable]
public class Sale
    {
        [JsonIgnore]
        public int SaleId { get; set; }
 //Identificação da máquina
        public string DeviceId { get; set; }
 //Nome de Utilizador
        public string UserName { get; set; }
 //Produto
        public string ItemChoice { get; set; }
 //Id de transação, fornecido pela máquina de vending
        public string TransationID { get; set; }
 //Assinatura do pedido
        public string Signature { get; set; }
 //Assinatura da sessão
        public string SessionID { get; set; }
 //Valor a pagar pelo produto
        public float Price { get; set; }
 //ID do movimento de pagamento
        public string PaymentID { get; set; }
 //Forma de pagamento
        public string PaymentSource { get; set; }
 //Data do pedido
        public DateTime RequestDate { get; set; }
 //Data de pagamento
        public DateTime? PaymentDate { get; set; }
 //Saldo final
        public float? NewBalance { get; set; }
     (…)
```

Excerto 7 - Classe e atributos Sale

Dada a natureza dos mecanismos de autenticação da *framework* .NET, a invalidação de *tokens* não se encontra disponível. Uma vez que o **UC04** depende desta ação, o processo adotado consiste no armazenamento dos *tokens* inválidos em base de dados para, posteriormente, serem utilizados no processo de validação. Para tal, os *tokens* inválidos são representados sob a forma de classe do tipo **BlacklistEntry** (excerto 8).

```
public class BlacklistEntry
    {
        public int BlacklistEntryId { get; set; }
        public string Token { get; set; }

        public string Username { get; set; }
        public DateTime CreationDate { get; set; }

        //Campos a ser preenchidos se for detetada uma tentativa de login
ilegal
        public DateTime? InvasionAttemptDate { get; set; }
        public string InvaderIP { get; set; }
     (…)
```

Excerto 8 - Classe e atributos BlacklistEntry

Adicionalmente, em caso de tentativa de acesso à plataforma com um *token* inválido, é registada a data do acontecimento e o IP do respetivo cliente.

Uma vez concebido o modelo de dados, é possível criar um conjunto de funções que seguem os padrões de repositório de persistência de dados, nomeadamente o *Create*, *Read*, *Update* e *Delete* (CRUD), denominado contexto de base de dados.

6.2.2 Implementação do Backend

Primeiramente, é tratada a implementação do **UC01**. Antes do registo de um utilizador externo na plataforma, o sistema verifica se o endereço de e-mail está em utilização.

Seguidamente, o sistema envia um e-mail com o código de confirmação para o endereço do utilizador.

O utilizador é armazenado na base de dados com o seu estado de confirmação inativo. De modo a confirmar o registo, o utilizador envia um pedido de confirmação para o *backend* com o respetivo código.

Após a verificação do código, o sistema procede à criação de um utilizador no *Web Service* de gestão de pagamentos e à ativação do utilizador de base de dados do sistema, ação representada no excerto 9.

```
(…)

            PAW_WebService.PAWUser UPLogin_PAW = new PAW_WebService.PAWUser();
            UPLogin_PAW.UserName = newUser.Email;
            UPLogin_PAW.Email = newUser.Email;
            UPLogin_PAW.UserID = newUser.Email;
            UPLogin_PAW.Name = newUser.Name + " " + newUser.LastName;
            UPLogin_PAW.BalanceActivateLimits = true;
            UPLogin_PAW.Enabled = true;
            UPLogin_PAW.IDUserProfile = Constants.IDUserProfile;

            //invoca o webservice do PAW para criar o novo utilizador
PAW_WebService.WSBalanceSoapClient client =
new    PAW_WebService.WSBalanceSoapClient(PAW_WebService.WSBalanceSoapClient.E
ndpointConfiguration.WSBalanceSoap12);

var result = await client.AddUserAsync(UPLogin_PAW, strToken, dateTime);
            int userID = result.Body.AddUserResult;

            // utilizador criado com sucesso
            if (userID > 0) {

                newUser.EmailConfirmed = true;
                newUser.ConfirmationCode = "";

                await _userManager.UpdateAsync(newUser);
                _context.SaveChanges();

                return Ok();
            }
```

Excerto 9 – Registo de um utilizador no *Web Service* de gestão de pagamentos

Os passos para o desenvolvimento do **UC02** são semelhantes ao registo, uma vez que se recorre à validação da identidade do utilizador através de um código de confirmação para repor a palavra-passe de acesso.

Relativamente ao **UC04**, o pedido de encerramento de sessão em todos os dispositivos consiste no registo do *token* inválido e remoção do *refresh token* da base de dados (excerto 10).

```
[Authorize]
[TypeFilter(typeof(VerifyBlacklist))]
// POST api/BlacklistToken
[HttpPost("BlacklistToken")]
public IActionResult BlacklistToken(){

            (…)

  var principal = pc.GetPrincipalFromExpiredToken(token);

                if (principal == null){
                    return BadRequest();
                }

_context.Blacklist.Add(new Model.BlacklistEntry(token,
principal.Identity.Name));

var invalidRefreshToken = _context.RefreshTokens.FirstOrDefault(rt =>
rt.Username == principal.Identity.Name);

                _context.RefreshTokens.Remove(invalidRefreshToken);
                _context.SaveChanges();

                return Ok();
        (…)
```

Excerto 10 – Pedido de encerramento de sessão em todos os dispositivos

Em todos os pedidos protegidos por autenticação, é utilizado um filtro, uma implementação da interface **IAuthorizationFilter,** representada no excerto 11, responsável por validar (antes do pedido ser executado) se o *token* existe na tabela **Blacklist** e, se a condição se verificar, devolver o respetivo código de erro HTTPS (403).

```
[AttributeUsage(AttributeTargets.All)]
public class VerifyBlacklist : Attribute, IAuthorizationFilter
{
(…)
private void unauthorizeRequest(AuthorizationFilterContext filterContext){

filterContext.HttpContext.Response.StatusCode = (int)HttpStatusCode.Forbidden;
            filterContext.HttpContext.Response.HttpContext.Features.Get<IHttpR
esponseFeature>().ReasonPhrase = "Not Authorized";

filterContext.Result = new JsonResult("NotAuthorized"){
                Value = new
                {
                    Status = "Error",
                    Message = "Blacklisted Token"
                },
        };
 }
(…)
```

Excerto 11 – Filtro de *tokens* inválidos

Se o sistema detetar que o *token* foi guardado na base de dados como inválido, é registada a data e hora da tentativa de acesso, assim como o IP do cliente (excerto 12). A resposta do pedido é enviada para o *frontend*, onde a sessão é terminada de imediato.

```
private bool verifyBlacklist(string token, AuthorizationFilterContext
filterContext){
            var entry = _context.Blacklist.FirstOrDefault(e => e.Token ==
token);
            if (entry == null){
                return true;
            }

             entry.InvasionAttemptDate = DateTime.Now;
entry.InvaderIP
=  filterContext.HttpContext.Connection.RemoteIpAddress.MapToIPv4().ToString()
;

            _context.Blacklist.Update(entry);
            _context.SaveChanges();

            return false;
}
```

Excerto 12 – Verificação de *token* e registo de informações de tentative de acesso

O **UC05 e UC06** baseiam-se na comunicação com o *Web Service* de pagamentos da FLUP, que disponibiliza as funcionalidades necessárias para gerir o perfil de *vending* do utilizador e consultar o saldo corrente.

Durante o processo de desenvolvimento, foi acordado, com o SI, que a implementação de um sistema de pagamento por cartão de crédito seria descartada, uma vez que o custo das operações atingiriam valores superiores aos dos produtos, tornando este método incomportável.

O desenvolvimento do **UC07** inicia-se com um pedido de pagamento ao serviço correspondente. É apresentado um exemplo do tipo de conteúdo enviado para o método de pagamento MBWay no excerto 13.

```
client = new HttpClient();

       client.DefaultRequestHeaders.Add("Authorization", "Bearer " +
Constants.PAYMENT_TOKEN);

string data = "entityId=xxxxxxxxxxxxxxxxxxxxxxxxxxxx" +
        "&amount=10.00" +
        "&currency=EUR" +
        "&paymentType=PA" +
        "&paymentBrand=MBWAY" +
        "&virtualAccount.accountId=351#911222111" +
        "&customParameters[SHOPPER_APP]=" + strApp +
        "&customParameters[SHOPPER_UserName]=" + username +
        "&customParameters[SHOPPER_UsereMail]=" + eMail;

(…)
```

Excerto 13 – Pedido de pagamento do serviço MBWay

O sistema fica à escuta da resposta que será enviada para um servidor *WebHooks* da FLUP. Se a resposta for positiva, a conta do utilizador será atualizada com o saldo do pedido de carregamento e o servidor responde ao *frontend* para ser mostrada a respetiva mensagem ao utilizador, como é possível verificar no excerto 14.

```
switch (result[0].resultcode)
{
                    case var isTrue when new
Regex(SIBS_ResultCodes.SIBS_TRANSACTION_SUCCEEDED).IsMatch(result[0].resultcod
e):
                    case var isTrueReview when new
Regex(SIBS_ResultCodes.SIBS_SUCCESSMANUALLYREVIEW).IsMatch(result[0].resultcod
e):
                        state = 1;//State.SUCCEEDED
                        break;
                    case var isTrue when new
Regex(SIBS_ResultCodes.SIBS_PENDINGTRANSACTION).IsMatch(result[0].resultcode):
                    case var isTrueDays when new
Regex(SIBS_ResultCodes.SIBS_PendingTransactionsDays).IsMatch(result[0].resultc
ode):
                        state = 2; //State.PENDING
                        break;
                    default:
                        state = 3; //State.CANCELLED
                        break;
                }
```

Excerto 14 – Tratamento de respostas do pedido de pagamento

Das possíveis respostas, o sistema pode detetar se o pagamento foi concluído com sucesso, se se encontra pendente ou se foi cancelado (pelo utilizador ou devido a uma quebra no fluxo). Adicionalmente, existe um código de resposta que indica que a operação foi concluída com

suspeita de fraude. Para o utilizador, esta ação é indetetável. Posteriormente, a transação deverá ser revista pelo gestor da plataforma, que tomará as devidas medidas.

O processo integra o protocolo *3-D Secure*, que, dependendo do método de pagamento, exige que o utilizador se autentique na plataforma 3DS do seu banco, ou aceite a transação através da aplicação MBWay.

Relativamente ao processo da interação com as máquinas de *vending*, começa-se pela implementação do **UC08**, respeitante à pesquisa e listagem das categorias de produtos de uma máquina de *vending*.

Do *frontend*, é enviada a identificação da máquina selecionada através de um código QR ou de uma etiqueta NFC, assim como o idioma dos termos que serão devolvidos. Para tal são criados dois tipos de pedidos, explicitados no excerto 15.

```
[Authorize]
        [TypeFilter(typeof(VerifyBlacklist))]
        // GET api/families/qrcode=xxxxxx&lang=pt-pt
        [HttpGet("qrcode={qrcode}&lang={langName}")]
        public ActionResult GetFamiliesByQRCode(string qrCode, string
langName)

 (…)

        [Authorize]
        [TypeFilter(typeof(VerifyBlacklist))]
        // GET api/families/nfcTag=xxxxxx&lang=pt-pt
 [HttpGet("nfctag={nfcTag}&lang={langName}")]
 public ActionResult GetFamiliesByNFCTag(string nfcTag, string langName)

 (…)
```

Excerto 15 – Pedidos de identificação da máquina de *vending*

A pesquisa é feita com base no identificador da máquina. Uma vez devolvida a lista, são reordenados os elementos por ordem do campo **Priority**, que dita a sua posição na UI (excerto 16).

```
           (…)
            var families = _context.Machines.Include(m =>
m.Families).FirstOrDefault(m => m.MachineId == machineId).Families;

            //ordenar por prioridades
            families = families.OrderBy(f => f.Priority).ToList();
         (…)
```

Excerto 16 – Ordenação de categorias, por prioridade

Na transmissão de dados para o cliente, servidor não expõe as informações suas entidades de base de dados diretamente para o cliente. Os dados são enviados em ***Data Tranfer Objects***

(DTO), com o objetivo de reduzir a partilha de informações não necessárias, assim como minimizar o *payload*. (Microsoft, 2022). É criado um **DTO** da classe **Family**, com a estrutura apresentada no excerto 17.

```
public class GetFamilyDTO
    {
        public GetFamilyDTO(string familyNameId, string imageId, string
translatedName)
        {
            FamilyNameId = familyNameId;
            ImageId = imageId;
            TranslatedName = translatedName;
        }
    (…)
```

Excerto 17 – Exemplo de modelo DTO, utilizado na transmissão de dados

O dicionário dos termos traduzidos no idioma pretendido encontra-se no formato de ficheiro JSON, que é lido através da classe de utilitária **LanguageParser**. É formada uma lista de **DTO** que será devolvida na resposta do pedido (excerto 18).

```
LanguageParser langPack = new LanguageParser(language);
(…)
foreach (Family fam in families){

listFamilies.Add(new GetFamilyDTO(fam.FamilyNameId, fam.ImageId,
langPack.getValue(fam.FamilyNameId)));
(…)
```

Excerto 18 – Construção do DTO, de acordo com o idioma recebido

Para a listagem dos artigos dos produtos, como descrito pelo **UC09**, é feita uma pesquisa dos objetos **Item** associados a uma **Family** e a ordenação por prioridades (excerto 19).

```
(…)
           var items = _context.Families
                .Include(f => f.FamilyItems)
                .FirstOrDefault(f => f.Machine.MachineId == machineId &&
f.FamilyNameId == familyNameId).FamilyItems;

            //ordenar por prioridades
            items = items.OrderBy(i => i.Priority).ToList();
(…)
```

Excerto 19 - Ordenação de produtos, por prioridade

O processo de conversão de objeto do modelo de dados em DTO é também utilizado no envio dos artigos para o *frontend*.

Finalmente, é implementado o **UC10**. Antes do processo de solicitação de um artigo a uma máquina de *vending*, verifica-se se o utilizador tem alguma uma transação pendente. Se sim, procede-se à respetiva cobrança. Este processo é executado através de um serviço assíncrono de *background* (implementação da interface **IHostedService,** representada no excerto 20):

```
internal sealed class WatchdogService : IHostedService, IDisposable
    {
    (…)
    private async Task SearchPendingSalesAsync()
        {
        (…)
        var pendingSales = _context.Sales.Where(s => s.PaymentSource == "Pen-
ding...");

        foreach (var ps in pendingSales){

        (…)

                        resultDebit = await Transactions.DebitaUtente(_con-
text, resultvenda, (decimal)ps.Price, ps.ItemChoice, ps.SaleId);

                        //se resultado ok
                        if (resultDebit.VendID > 0){

                            //atualiza venda
                            ps.PaymentDate = DateTime.Now;
                            ps.PaymentID = resultDebit.VendID.ToString();
                            ps.PaymentSource = resultDebit.Result;
                            ps.NewBalance = (float)resultDebit.Saldo;
                            _context.Sales.Update(ps);
                        }
                }
        (…)
```

Excerto 20 – Serviço de tratamento de pagamentos pendentes

Seguidamente, o sistema consulta a base de dados para obter o artigo escolhido pelo utilizador de forma a registá-lo no objeto **Sale** respetivo à venda: o pedido de solicitação do artigo foi iniciado, mas o pagamento ainda não foi concluído.

No próximo passo, trata-se o pedido de dispensa à máquina de *vending*. Verifica-se se a máquina se encontra disponível e operacional. Em seguimento, o sistema verifica se o artigo está disponível. O processo é descrito, em linguagem de programação, no excerto 21.

```
string UrlErogate = UrlDevice + "/sendtointerface+remoteselect+" + artigo;
                        client.DefaultRequestHeaders.Add("manager-code",
Constants.ManagerCodeEPay);
    client.DefaultRequestHeaders.Add("api-key", Constants.ApiKeyEPay);

httpResponse = await client.PostAsync(@UrlErogate, null);
```

Excerto 21 – Exemplo de verificação da disponibilidade de um artigo de uma máquina

O sistema envia o pedido de dispensa e aguarda o resultado da operação, através de um pedido de estado, de um em um segundo, de forma a obter uma das possíveis respostas:

- O produto foi dispensado com sucesso
- O produto não foi dispensado e o utilizador não deve ser cobrado
- Ocorreu um problema com a máquina no processo de dispensa

O processo é apresentado no excerto 22.

```
(…)

if (httpResponse.IsSuccessStatusCode) {
      (…)

      json = JObject.Parse(responseBody);
      messageID = (string)json["message"];
      UrlStatus = UrlDevice + "/" + messageID;

      tmpget = 0; //forçar a saída
      message = "";


      while ((message == "") && tmpget < 5) {

                httpResponse = await client.GetAsync(@UrlStatus);
                httpResponse.EnsureSuccessStatusCode();

            if (httpResponse.IsSuccessStatusCode) {

            (…)

            json = JObject.Parse(responseBody);

            message = (string)json["message"];

                  if (message.IndexOf("ack enderog") == -1 && message.In-
dexOf("ack refund") == -1 && message.IndexOf("notavailable") == -1 && mes-
sage.IndexOf("not available") == -1) {
                                    Thread.Sleep(1000);
                   }
             }

            tmpget++;
       }
        (…)
```

Excerto 22 – Tratamento do pedido de dispensa à máquina

6.2.3 Implementação do Frontend

Neste subcapítulo serão abordados os principais pontos do desenvolvimento da interface gráfica do utilizador, através da apresentação dos ecrãs da aplicação e excertos de código relevantes.

Modo de funcionamento do Flutter:

O *frontend* é desenvolvido em forma de aplicação móvel multiplataforma, recorrendo à *framework* Flutter e à linguagem de programação Dart.

Os componentes de UI são concebidos com base em ***Widgets***, que definem como os elementos gráficos devem ser representados na ***View***. Os *widgets* podem ser indexados e organizados como no exemplo da Figura 32.

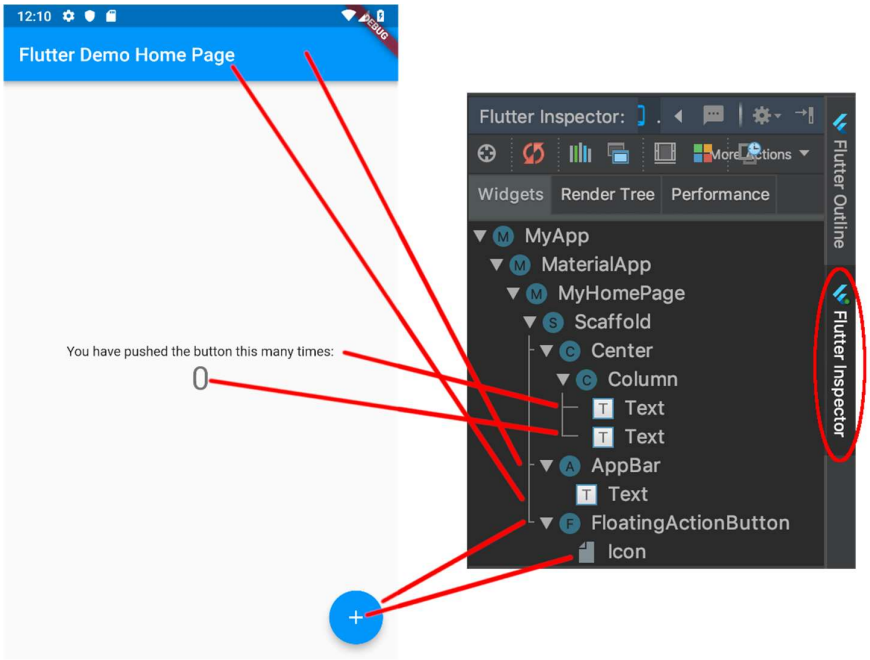

Figura 32 - Exemplo de Aplicação de Widgets (Suragch, 2018)

Aplicação da *framework* à implementação da UI:

Na Figura 33, é apresentado o ecrã de início de sessão, o primeiro a ser mostrado ao utilizador, após uma nova instalação da aplicação.

Figura 33 – Ecrã de Início de Sessão

A disposição dos elementos segue uma formatação em lista, derivada da aplicação do *widget* **ListView** (excerto 23), que integra as caixas de texto para recolher o e-mail e password do utilizador, bem como os botões de *login* e de inscrição na plataforma para utilizadores externos.

De forma a existir coerência entre os elementos visuais das máquinas e da aplicação, foi adicionado o logótipo da empresa de *vending* que disponibiliza e abastece as máquinas e da Faculdade de Letras da Universidade do Porto.

```
(…)

child: Container(
          child: Padding(
            padding: const EdgeInsets.all(36.0),
            child: ListView(
              children: <Widget>[
                SizedBox(
                  height: 100.0,
                  child: Image.asset(
                    "assets/caffe.png",
                    semanticLabel:
myEscolha.SelIdioma['autenticacao_login_image_semanticLabel'],
                    fit: BoxFit.contain,
                  ),
                ),
                emailField,
                passwordField,
                loginButton,
                registerButton,
(…)
```

Excerto 23 – Elementos do ecrã de *login*

No ecrã de registo de utilizadores externos (Figura 34), é pedida a inserção de um código CAPTCHA (*Completely Automated Public Turing test to tell Computers and Humans Apart*), um teste que prova que o registo está a ser executado por um humano e não por uma máquina, que tem como principal objetivo combater o *spam* e/ou *bots* (software desenvolvido com o objetivo de replicar ações humanas) (Cloudflare, s.d.).

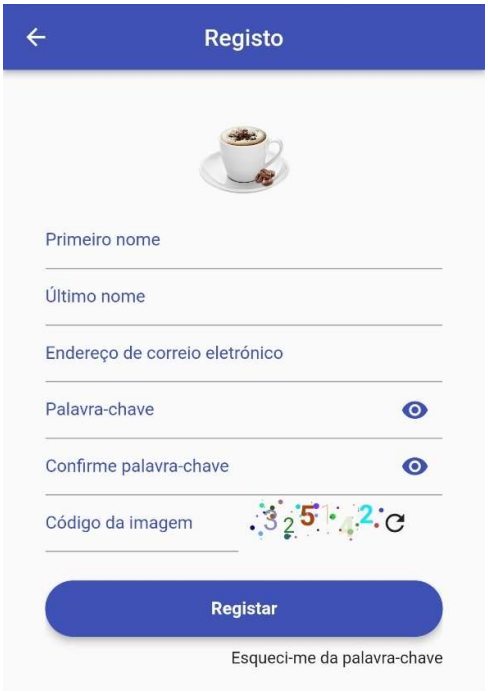

Figura 34 - Ecrã de Registo de Utilizador Externo

A gráfico do código CAPTCHA é gerado através do plugin *open-source* **hb_check_code**, que fornece um *widget*, que é integrado no *layout* de acordo com o excerto 24.

```
(…)

Expanded(
  flex: 1,
  child: Container(
      alignment: Alignment.bottomRight,
      child: HBCheckCode(
        code: code,
      ))
),

(…)
```

Excerto 24 – Integração do CAPTCHA

Uma vez autenticado, o utilizador é redirecionado para o ecrã principal (Figura 35) onde se encontram os botões de acesso às funcionalidades de compra de artigos, carregamento e consulta de saldo, bem como um menu lateral (Figura 36) onde é apresentado o nome e endereço de e-mail do utilizador, assim como o acesso às definições da aplicação.

Figura 35 - Ecrã Principal da Aplicação

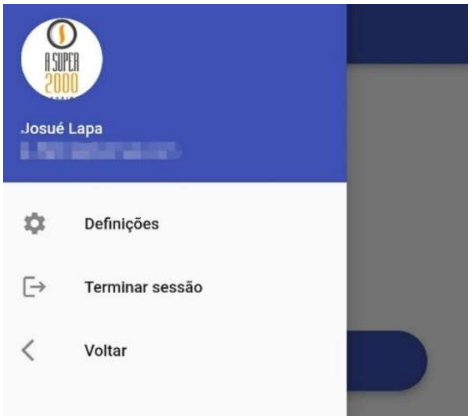

Figura 36 - Menu Lateral do Ecrã Principal

O menu de definições é composto pelas funcionalidades descritas na Figura 37.

Figura 37 - Menu de Definições

Os "*packs*" de idiomas da aplicação são armazenados em ficheiros JSON, que contêm, para além dos termos traduzidos, o campo "*priority*" que define a sua posição na lista de idiomas. É possível verificar a ordenação dos idiomas através da Figura 38, de acordo com o excerto 25, de um ficheiro JSON.

No primeiro arranque, o idioma da aplicação é definido de acordo com o do dispositivo. Se o idioma não estiver disponível na aplicação, é predefinido o Inglês.

```
{
  "language_name": "Português (Portugal)",
  "flag_code": "PT",
  "priority": "1",
  "main_title_app": "Login APP",
  "main_title_auth": "Autenticação",

(…)
```

Excerto 25 – *Pack* de idioma em formato JSON

Figura 38 – Apresentação de idiomas por ordem de prioridade

Uma vez importados para a aplicação, os termos são armazenados em formato de mapa, que permite a busca de um valor através de uma chave correspondente ao valor pretendido (excerto 26).

```
String data = await rootBundle.loadString("assets/languages/"+ language
+".json");

languageJSON = json.decode(data);

(…)

SizedBox(
      height: 100.0,
      child: Image.asset(
            "assets/caffe.png",
            semanticLabel: languageJSON['auth_login_semanticLabel'],
                  fit: BoxFit.contain,
                ),
              ),
(…)
```

Excerto 26 – Aplicação de termos de idiomas

No que toca ao carregamento de saldo, é exposto o ecrã do método de pagamento MBWay (Figura 39), onde o utilizador insere o seu número de telemóvel e o valor que pretende descontar para carregar a sua conta.

Figura 39 - Ecrãs do método de pagamento MBWay

Relativamente ao processo de seleção e dispensa do produto, começa-se por identificar a máquina de *vending*.

Como referido anteriormente, as máquinas integradas no sistema estão identificadas com um código QR e uma etiqueta NFC.

Figura 40 - Interface de uma Máquina de *Vending* instalada na FLUP

As máquinas têm afixadas duas etiquetas: uma posicionada a uma altura mais elevada para a comodidade dos indivíduos que se encontram de pé; outra a uma altura mais baixa, com vista a facilitar o acesso a portadores de deficiência física, que utilizam cadeira de rodas (Figura 40).

Ao selecionar a opção “Comprar”, na aplicação, o utilizador é redirecionado para um ecrã que fica à escuta da leitura de uma etiqueta NFC. Para tal, é utiliza a ferramenta **FlutterNfcReader**, como é possível verificar no excerto 27.

```
  (...)

FlutterNfcReader.onTagDiscovered().listen((onData) {

        widget.myEscolha.SelNFCTag = onData.id.toString();

        if (widget.myEscolha.SelNFCTag.toString() != null) {
          Navigator.push(
            context,
            MaterialPageRoute(builder: (context) => FamilyRoute(myAcucar:
widget.myAcucar, myEscolha: widget.myEscolha)),
          );
        } else {
          showAlertDialog(context,myEscolha.SelIdioma["machineroute_alert_titl
e1"],myEscolha.SelIdioma["machineroute_alert_message1"]);
        }

  (...)
```

Excerto 27 – Mecanismo de leitura de etiquetas NFC

Existe também a funcionalidade de leitura de um código QR, suportada pela ferramenta **BarCodeScanner**, apresentada no excerto 28.

```
String qrResult = (await BarcodeScanner.scan()) as String;
```

Excerto 28 - Mecanismo de leitura de códigos QR

Uma vez identificada a máquina, as categorias de produtos são apresentadas e, mediante a seleção, são apresentados os artigos correspondentes (Figura 41).

Figura 41 - Ecrã de Seleção da Categoria de Produtos, e de Produtos

No ecrã de seleção de artigos (Figura 41), é disponibilizada a opção de controlar o nível de açúcar através de um ***Slider*** (excerto 29), e de alternar entre incluir ou não um copo (em determinadas bebidas).

```
Slider(
  value: _sugarValue(),
  onChanged: (newSugar) {
    setState(() => _sugarValue = newSugar );
  },
    min: 0,
    max: 5,
    divisions: 5,
    label: "$_sugarValue"
)
```

Excerto 29 – *Slider* de personalização do nível de açúcar

Ferramentas de Acessibilidade:

A aplicação foi adaptada para integrar as ferramentas de acessibilidade do dispositivo, para a leitura de elementos visuais dos ecrãs e tradução para voz, sons e/ou vibrações, orientadas a utilizadores cegos ou com deficiências visuais.

No sistema Android, é utilizado o serviço Google TalkBack (Google) e, no iOS, ferramentas incorporadas no sistema operativo (Apple, 2022).

Para descrever o que está a ser apresentado no ecrã, os textos e imagens apresentados devem estar devidamente identificados através de descritivos que possam ser lidos pelos sintetizadores de voz. É o caso do botão que permite alternar entre incluir ou não um copo (Figura 42).

Quando a ferramenta de acessibilidade está ativa, esta lê o campo **semanticLabel** do recurso correspondente à imagem, a que é atribuído o valor do JSON (excerto 30 e 31) correspondente ao idioma da aplicação.

Figura 42 - Seleção de um elemento visual do serviço TalkBack

```
(…)
"itemroute_cup_image": "Imagem clicável de um copo",
"itemroute_nocup_image": "Imagem clicável de um copo com uma cruz vermelha
sobreposta",
(…)
```

Excerto 30 – Termos para leitura por sintetizadores de voz

```
GestureDetector(
  onTap: (() {
    setState(() {
      if (_myImage == "assets/semcopo.png"){
        _myImage = "assets/copo.png";
        _myImageLabel = myEscolha.SelIdioma["itemroute_cup_image"];
        myAcucar.myCopo = 1;
      } else {
        _myImage = "assets/semcopo.png";
        _myImageLabel = myEscolha.SelIdioma["itemroute_nocup_image"];
        myAcucar.myCopo = 0;
      }
    });
  }),
  child: Image.asset(_myImage,semanticLabel: _myImageLabel,height: 50.0,),
)
```

Excerto 31 – Aplicação dos termos de acessibilidade à UI

De forma a reduzir a fadiga ocular dos utilizadores em ambientes escuros, assim como melhorar a autonomia de dispositivos dotados de ecrãs AMOLED, é boa prática incluir um esquema de

cores dominado por textos claros e fundos escuros, denominado “Modo Escuro” ou “Modo Noite”.

São criados dois esquemas de cores (excerto 32) que poderão ser alternados através de uma entrada de menu de definições:

```
ThemeData lightTheme = ThemeData(
      brightness: Brightness.light,
      primaryColorLight: Colors.indigo,
      primaryColorDark: Colors.indigo,
      backgroundColor: Color.fromARGB(255, 100, 100, 100),
      appBarTheme: AppBarTheme(systemOverlayStyle:
SystemUiOverlayStyle.light),
      focusColor: Colors.indigo, textSelectionTheme:
TextSelectionThemeData(selectionColor: Colors.black87), colorScheme:
ColorScheme.fromSwatch(primarySwatch: Colors.indigo).copyWith(secondary:
Color.fromARGB(255, 248, 248, 248)) ,
);

ThemeData darkTheme = ThemeData(
      brightness: Brightness.dark,
      primaryColorLight: Colors.white70,
      primaryColorDark: Colors.white30,
      backgroundColor: Color.fromARGB(255, 30, 30, 30),
      focusColor: Colors.white70 ,
      textSelectionTheme: TextSelectionThemeData(
        cursorColor: Colors.white70,
      ),
);
```

Excerto 32 – Definição dos temas e respetivos esquemas de cores

Na Figura 43, são apresentados, lado a lado, os dois esquemas apresentados:

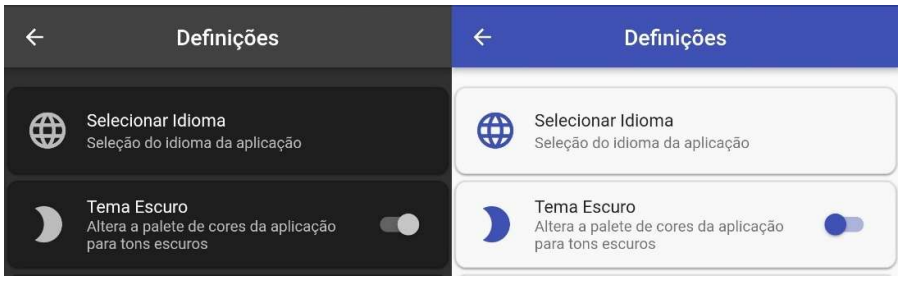

Figura 43 - Modo Escuro vs Modo Claro

6.2.4 Testes à solução desenvolvida

De modo a assegurar a qualidade do software desenvolvido, foram efetuados diferentes tipos de testes:

- **Testes Funcionais:** averiguam as funcionalidades ditadas pelos requisitos impostos pelo cliente. Estes testes são executados com base em pedidos HTTPS ao servidor *backend*, cujas respostas são avaliadas, de forma a entender se os valores devolvidos correspondem aos valores esperados, de acordo com a lógica de negócio.
- **Testes de aceitação:** consistem na reprodução de ações que os vários tipos de utilizadores executam quando utilizam o sistema. Para tal, tanto o *frontend* como o *backend* devem estar totalmente operacionais e interligados. Durantes estes testes, será avaliado o tempo necessário para executar uma ação.

Para a conceção dos testes funcionais, recorreu-se ao *software* Postman para validar as respostas do *backend*, de acordo com a Figura 44.

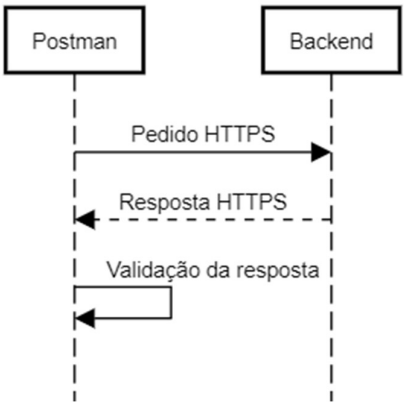

Figura 44 - Metodologia de testes funcionais

Os testes consistem na realização do pedido, na verificação do código, tempo e conteúdo da resposta.

Espera-se uma resposta identificada com o código 200 (sucesso).

No caso deste sistema, foi definido um critério que dita que o tempo de resposta de um pedido deverá ser menor ou igual a 200 milissegundos, de modo a manter a fluidez na navegação entre menus e execução de tarefas.

Seguidamente, analisa-se o conteúdo que deve corresponder ao resultado esperado.

Como exemplo, foi apresentado o resultado dos testes ao pedido de listagem das famílias de artigos através da Figura 45.

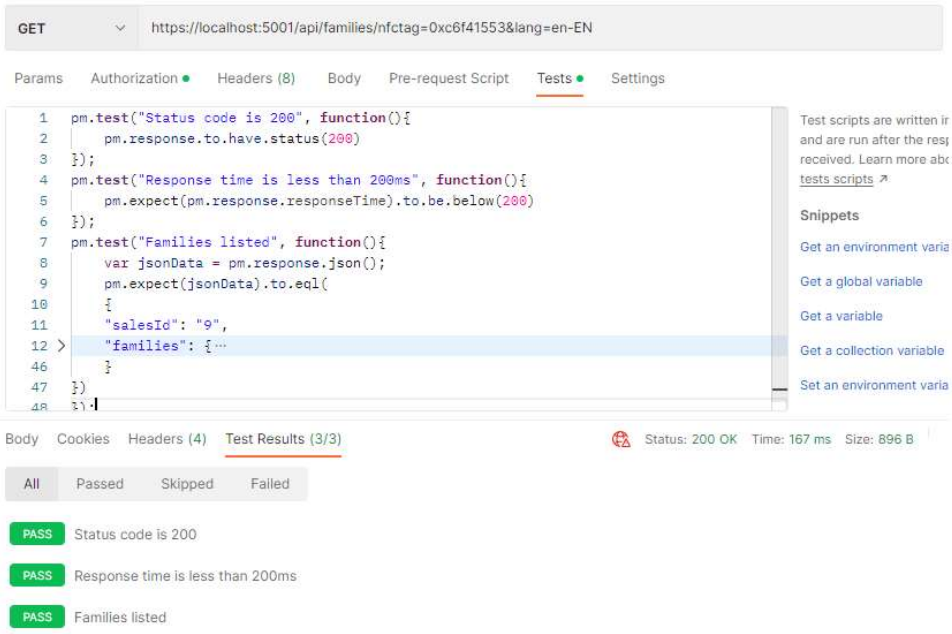

Figura 45 - Lógica de testes funcionais

No que toca aos testes de aceitação, foi executado um fluxo de tarefas, de modo a assegurar o cumprimento dos requisitos do projeto, através da replicação das ações de um utilizador. O processo é descrito através de um diagrama de fluxo, representado na Figura 46.

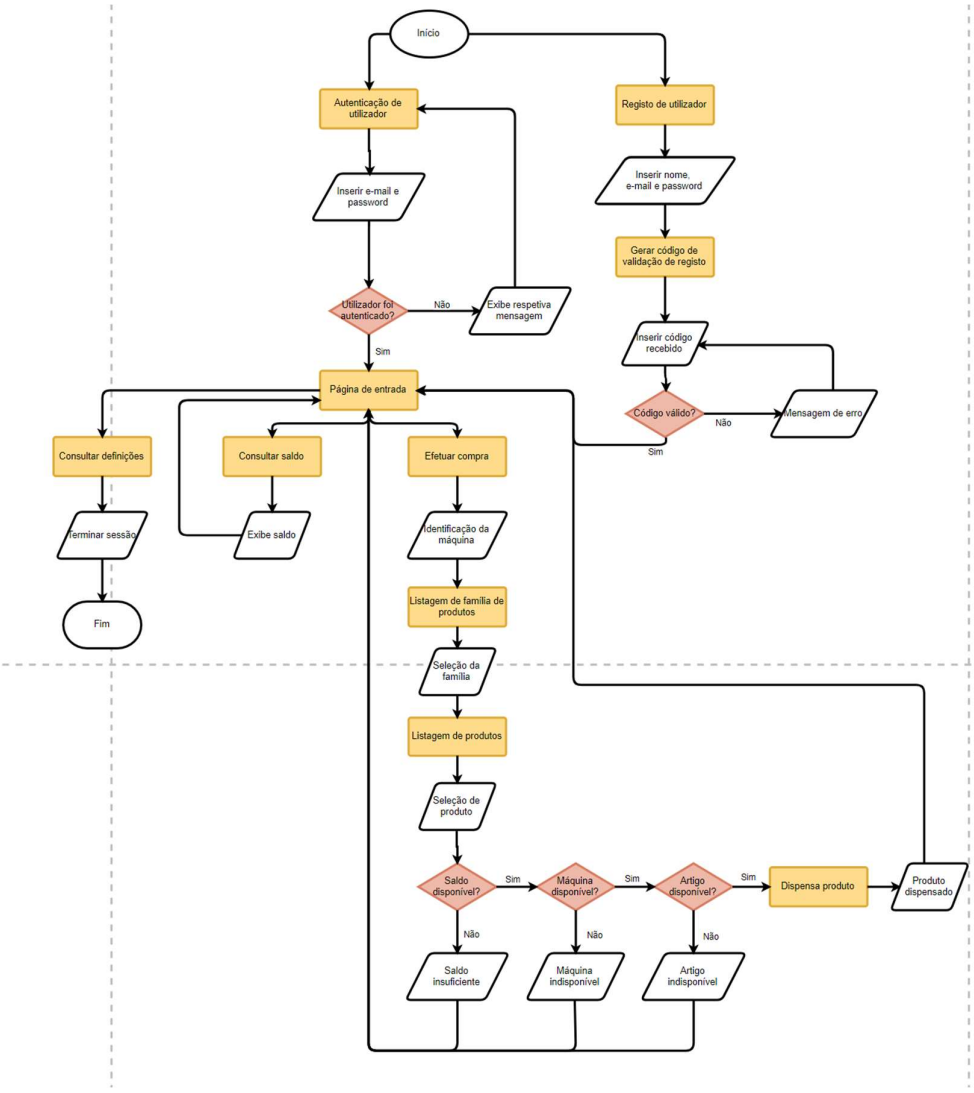

Figura 46 - Fluxograma de execução de testes de aceitação

Os testes de aceitação efetuados não incluem o processo de carregamento de saldo uma vez que, no momento da escrita deste documento, os servidores de teste do fornecedor do respetivo serviço não se encontravam disponíveis, pelo que, a funcionalidade foi ensaiada e revista durante o desenvolvimento do *backend* e a execução de testes funcionais.

6.3 Conclusões

Este capítulo descreve o processo de desenvolvimento de *software*, através de explicações detalhadas da implementação das funcionalidades de *backend* e *frontend*. Para tal, recorreu-se a excertos de código, gráficos e imagens da UI.

Foram descritos pormenores relativos à *Entity Framework .NET*, úteis para a construção do modelo de dados e gestão dos mesmos.

Finalmente, recorreu-se a testes funcionais e de aceitação, de modo a assegurar o bom funcionamento do *software* desenvolvido.

7 Avaliação da Solução

No presente capítulo, a solução implementada é avaliada de forma a determinar se os objetivos definidos foram atingidos.

Para tal, é utilizado o modelo *Quantitative Evaluation Framework* (QEF) para analisar se o produto desenvolvido corresponde às expectativas do cliente e dos consumidores.

7.1 Quantitative Evaluation Framework

O modelo de avaliação QEF (Escudeiro & Bidarra, 2008) é aplicado ao contexto de engenharia informática, através de um conjunto de parâmetros que determinam se a solução desenvolvida se adequa à solução pretendida.

O processo de avaliação de qualidade baseia-se na medição da "distância" entre o **Sistema Ideal** – uma abstração de uma solução que cumpre todos os requisitos impostos, e o **Sistema Real** – a solução concreta, resultado do desenvolvimento descrito neste documento.

A *framework* tem como base a relação entre três conceitos:

- **Dimensão**: composta por um conjunto de fatores
- **Fator**: composto por um conjunto de requisitos
- **Requisito**: condição imposta no planeamento do projeto

As medidas de qualidade que fundamentam o estudo são as seguintes:

- **Peso do requisito**: importância de um requisito, associada ao fator em que está inserido. É utilizada uma escala de medição com menor divisão igual a 2, de 0 a 10
- **Cumprimento do requisito**: valor percentual, correspondente ao nível de concordância entre uma funcionalidade desenvolvida e o requisito correspondente. É utilizada uma escala de medição com menor divisão igual a 50%
- **Peso do fator**: importância de um fator, calculada com base na divisão do número de requisitos incluídos no fator pelo número de fatores da dimensão correspondente
- **Cumprimento do fator**: valor percentual, correspondente à medição do cumprimento global dos requisitos de um fator
- **Cumprimento da dimensão**: valor percentual, correspondente à medição do cumprimento dos fatores da dimensão

Foram estudadas as seguintes dimensões, no contexto de avaliação do projeto:

- **Funcionalidade**

- **Usabilidade**
- **Suportabilidade**

7.1.1 Metodologia de Avaliação

De modo a ser feita uma avaliação coerente, e de acordo com um cenário real, foi selecionado um conjunto de utilizadores internos e externos à faculdade, de modo a testarem e avaliarem o *software* desenvolvido, mais concretamente a dimensão Usabilidade, através de um inquérito cujas respostas servirão de objeto de avaliação. Os utilizadores foram informados do intuito da aplicação, e de noções base para compreensão da mesma.

As dimensões Funcionalidade e Suportabilidade serão avaliadas pelo autor, com base no seu conhecimento do sistema e nos testes efetuados.

O inquérito de Usabilidade consiste na avaliação de afirmações relativas à aplicação. O avaliador pode selecionar um valor de uma escala de 0 a 2, em que 0 significa que discorda totalmente da afirmação e 2 que concorda totalmente com a afirmação. Esta adaptação permite obter uma métrica de cada requisito, diretamente associada a uma métrica do QEF (0 - 0%; 1 – 50%; 2 - 100%). Os seus critérios podem ser encontrados no Apêndice A. O inquérito baseia-se na avaliação das seguintes afirmações:

- **C01** - Os menus da aplicação são fáceis de compreender e navegar
- **C02** - A aplicação responde rapidamente às ações executadas
- **C03** - A interface e as funcionalidades da aplicação assemelham-se à interface e funcionalidades das máquinas de *vending*
- **C04** - A terminologia utilizada adequa-se ao contexto da área de negócio de *vending*
- **C05** - As funcionalidades da aplicação são de fácil acesso
- **C06** - A aplicação está adaptada a utilizadores invisuais e com dificuldades visuais

Nota: A última afirmação apenas deverá ser avaliada por utilizadores que tenham recorrido ao sintetizador de voz do dispositivo para navegar na aplicação.

Adicionalmente, é disponibilizado um campo de texto onde os utilizadores poderão registar sugestões.

7.1.2 Resultados da Avaliação

Uma vez recolhidas as respostas do inquérito, é efetuada a respetiva análise de dados.

Primeiramente, são apresentados e analisados os resultados do inquérito, em forma de gráficos de barras, com as respetivas percentagens e número de utilizadores de cada resposta. Foi possível obter avaliações de 12 utilizadores, internos e externos.

De forma a verificar a tendência dos dados recolhidos, foi calculada a média, mediana e desvio padrão das respostas, por cada afirmação.

Os resultados da avaliação da navegação e compreensão da aplicação encontram-se na Figura 47.

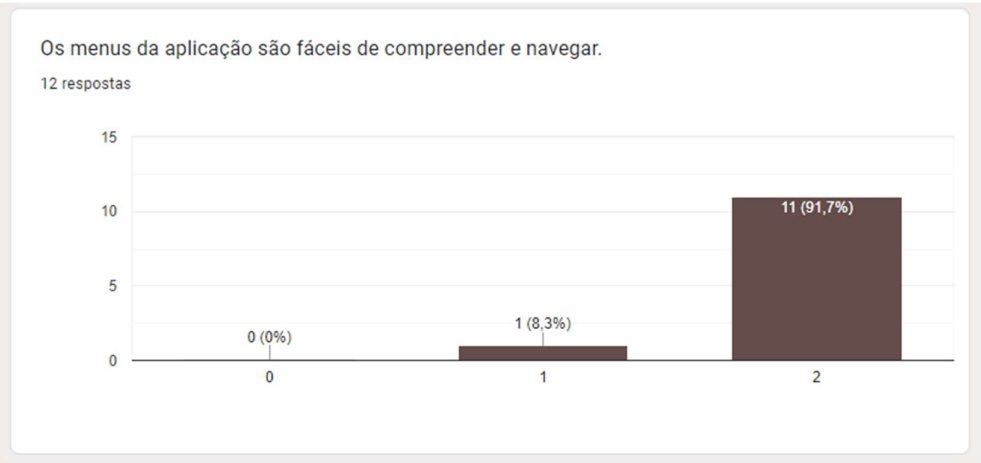

Figura 47 – Resultados da avaliação do critério C01

Os valores estatísticos relativos ao critério C01 são os seguintes:

- **Média:** 1,92
- **Mediana:** 2
- **Desvio Padrão:** 13,82%

Existe um grau elevado de concordância nos resultados, que indica que a maioria dos utilizadores da amostra concorda que a aplicação é fácil de utilizar e compreender.

Os resultados da avaliação da performance da aplicação encontram-se na Figura 48.

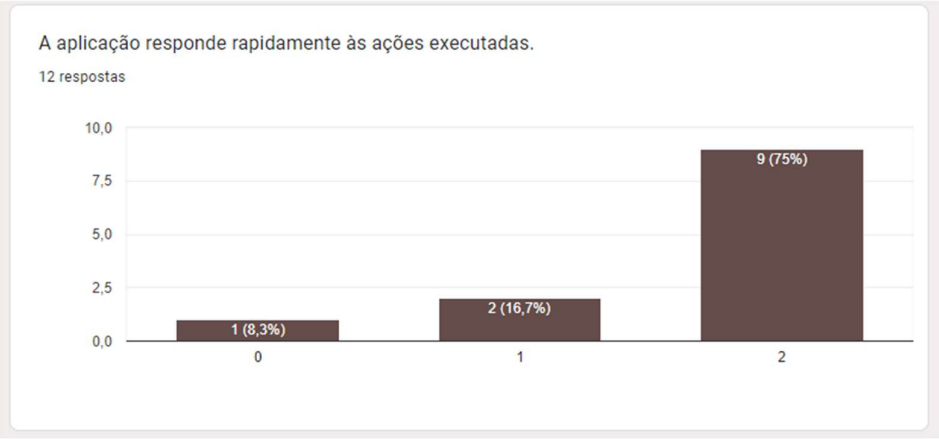

Figura 48 - Resultados da avaliação do critério C02

Os valores estatísticos relativos ao critério C02 são os seguintes:

- **Média:** 1,67
- **Mediana:** 2
- **Desvio Padrão:** 31,18%

Os dados revelam que existe uma pequena divergência de opiniões, verificando-se um desvio padrão mais elevado. A razão para estes resultados poderá estar relacionada com o estado da rede, das máquinas ou variedade de *hardware*, uma vez que o volume de dados processados por cada utilizador é relativamente semelhante. Durante os testes de performance, o tempo de resposta foi, de uma forma geral, semelhante, pelo que não foi encontrado outro fator, para além dos referidos anteriormente.

Os resultados da avaliação da relação entre a aplicação e as máquinas de *vending* são apresentados na Figura 49.

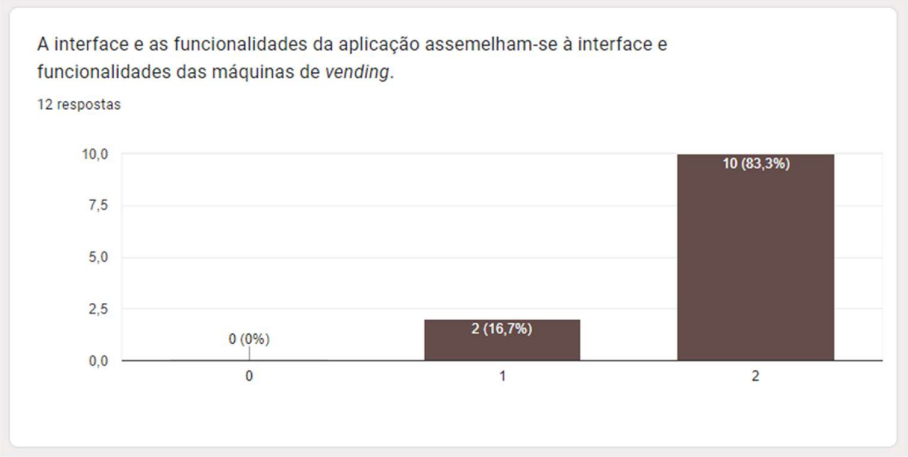

Figura 49 - Resultados da avaliação do critério C03

Os valores estatísticos relativos ao critério C03 são os seguintes:

- **Média:** 1,83
- **Mediana:** 2
- **Desvio Padrão:** 18,63%

A avaliação deste critério não revela uma grande discrepância nas respostas. No entanto, alguns utilizadores poderão ter notado algumas diferenças entre os dois sistemas, no que toca à ordenação ou posição de certos elementos no ecrã.

Os resultados da avaliação da terminologia são apresentados na Figura 50.

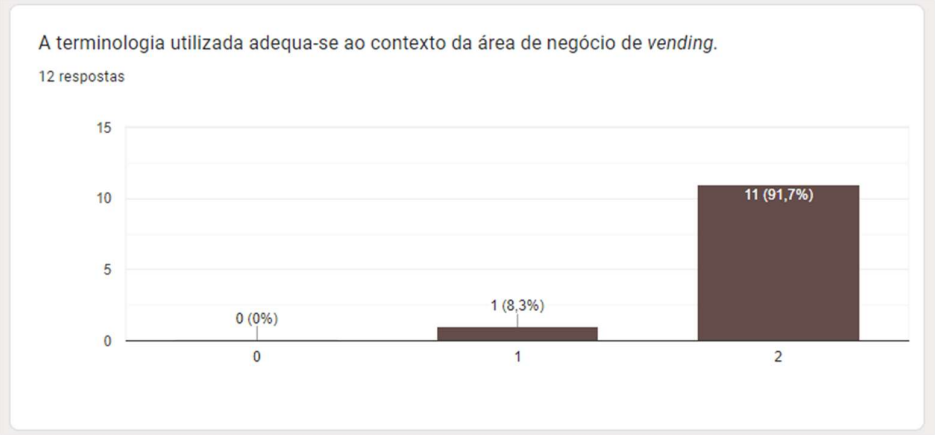

Figura 50 - Resultados da avaliação do critério C04

Os valores estatísticos relativos ao critério C04 são os seguintes:

- **Média:** 1,92
- **Mediana:** 2
- **Desvio Padrão:** 13,82%

No caso da terminologia utilizada, a maioria dos utilizadores da amostra concorda que se adequa ao contexto de *vending*.

Os resultados da avaliação da terminologia são apresentados na Figura 51.

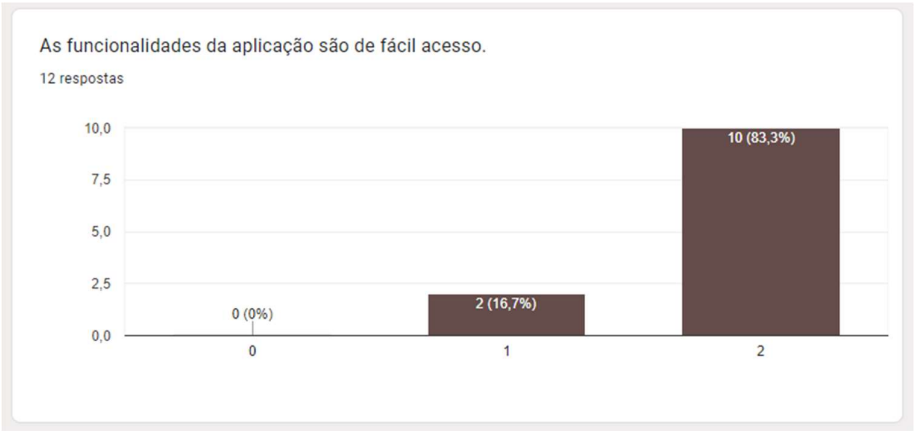

Figura 51 - Resultados da avaliação do critério C05

Os valores estatísticos relativos ao critério C05 são os seguintes:

- **Média:** 1,83
- **Mediana:** 2

- **Desvio Padrão:** 13,63%

De uma forma geral, os utilizadores conseguiram facilmente aceder às funcionalidades que pretendiam executar.

Os resultados da avaliação da acessibilidade da aplicação são apresentados na Figura 52.

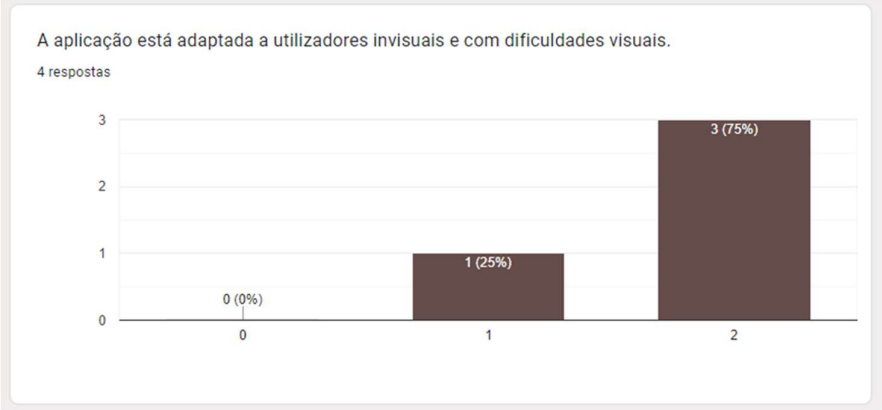

Figura 52 - Resultados da avaliação do critério C06

Os valores estatísticos relativos ao critério C05 são os seguintes:

- **Média:** 1,75
- **Mediana:** 2
- **Desvio Padrão:** 21,65%

A recolha de avaliações de utilizadores cegos ou com dificuldades visuais foi limitada, uma vez que apenas foi possível obter uma amostra de 4 utilizadores. No entanto, a maioria concorda que a aplicação está adaptada para ser utilizada com sintetizadores de voz. O ligeiro desvio verificado poderá estar relacionado com a forma de identificação de certos elementos do ecrã e a carência de detalhe.

Adicionalmente, foram analisadas as sugestões enviadas pelos avaliadores. As mais destacadas foram as seguintes:

- Permitir consultar os últimos movimentos
- Lista de favoritos

Através destas informações, foi possível compreender as principais necessidades do utilizador. A consulta do histórico de transações é uma funcionalidade útil para o utilizador ter uma maior perceção dos seus gastos. A lista de favoritos permite criar atalhos e evitar navegações desnecessárias entre páginas, assim como reduzir o número de pedidos ao servidor.

Uma vez traduzidos os resultados, o QEF, representado na Figura 53, é preenchido de acordo com a concretização de cada requisito.

Q_i	Dimension	Q_i	W_{ij} (Factor Weight j in Dim i) [0,1]	Factor	rw_{jk} (requirement weight k in Factor j) {2, 4, 6, 8, 10}	Requirement	wf_x % requirement fulfillment k) [0,100]
96,15	Funcionalidade	100	0,31	Utilizador	10	FFU01 - Início de sessão através do controlador de domínio (utilizadores internos)	100
					10	FFU02 - Criação de conta (utilizadores externos)	100
					6	FFU03 - Consulta de informações de perfil	100
					10	FFU04 - Recuperação de credenciais de acesso (utilizadores externos)	100
		100	0,08	Segurança	8	FFS01 - Encerrar sessão em todos os dispositivos	100
		75	0,15	Pagamento	10	FFP01 - Carregamento de saldo de conta	50
					10	FFP02 - Consultar saldo de conta	100
		100	0,46	Vending	10	FFV01 - Ativar/desativar perfil de vending	100
					10	FFV02 - Ler identificação de uma máquina de vending	100
					10	FFV03 - Consultar famílias de produtos	100
					10	FFV04 - Consultar produtos de uma máquina	100
					10	FFV05 - Personalização de produtos	100
					10	FFV06 - Solicitação de um produto	100
92,26	Usabilidade	90,625	0,57	UX	10	UU01 - O layout da interface é compreensível e de fácil navegação	95,83
					10	UU02 - As funcionalidades da aplicação são de fácil acesso	91,67
					10	UU03 - Existe congruência entre o software das máquinas de vending e a aplicação	91,67
					10	UU04 - O tempo de resposta das ações executadas pelo utilizador é baixo	83,33
		97,915	0,29	Linguagem	10	UL01 - O sistema suporta os idiomas Português (Portugal) e Inglês	100
					10	UL02 - É utilizada terminologia da área de negócio de vending	95,83
		87,5	0,14	Acessibilidade	10	UA01 - A aplicação está adaptada a utilizadores invisuais e com dificuldades visuais	87,5
83,33	Suportabilidade	100	0,33	Adaptabilidade	10	SA02 - A aplicação está disponível para o sistema Android e iOS	100
		75	0,67	Manutenibilidade	10	SM01 - O sistema trata e regista erros e falhas	50
					10	SM02 - O sistema pode ser facilmente ampliado, introduzindo novas funcionalidades	100

Figura 53 – Resultados da avaliação dos requisitos do QEF

Através das medições, obtiveram-se os seguintes valores, correspondentes a cada dimensão do QEF:

- **Funcionalidade** – 96,15%
- **Usabilidade** – 92,26%
- **Suportabilidade** – 83,33%

De modo a calcular a "distância" entre o **Sistema Ideal** e o **Sistema Real**, recorreu-se fórmula 3, cujo resultado indica o índice de qualidade total q, que representa a percentagem do **Sistema Ideal** que foi coberta pelo **Sistema Real**:

$$q = \frac{\log\left(\frac{1+Q_{funcionalidade}}{100}\right) + \log\left(\frac{1+Q_{usabilidade}}{100}\right) + \log\left(\frac{1+Q_{suportabilidade}}{100}\right)}{3 * \log(2)} \quad (3)$$

$$\Leftrightarrow q = \frac{\log\left(\frac{1+96{,}15}{100}\right) + \log\left(\frac{1+92{,}26}{100}\right) + \log\left(\frac{1+83{,}33}{100}\right)}{3 * \log(2)} = \mathbf{93}\%$$

O resultado obtido significa que o **Sistema Real** representa **93%** do **Sistema Ideal**.

7.1.3 Conclusões

O presente capítulo detalha o processo de avaliação da solução desenvolvida, com recurso ao à ferramenta QEF, que permitiu obter conclusões acerca da percentagem de concretização dos requisitos, com base em métricas de qualidade.

Os valores qualitativos das dimensões Funcionalidade e Suportabilidade indicam que a maioria dos requisitos definidos foram cumpridos. No entanto, como anteriormente referido, houve fatores que impediram a implementação de determinadas funcionalidades, associadas a requisitos integrantes do plano inicial do projeto.

A avaliação da dimensão Usabilidade teve, de uma forma geral, um balanço positivo, tendo-se verificado resultados concordantes das avaliações submetidas pelos utilizadores. No entanto, através desta análise foi possível identificar alguns pontos a melhorar, de acordo com as opiniões dos utilizadores e respetivas sugestões.

8 Conclusões

Findo o processo desenvolvimento do projeto, verifica-se se os objetivos previamente definidos foram alcançados com sucesso. Através dos dados obtidos no capítulo 7, averigua-se se as metas propostas para o projeto foram atingidas.

Ainda neste capítulo, são apresentados fatores que influenciaram o resultado do projeto, assim como procedimentos que poderiam colmatar deficiências da solução. Adicionalmente, são abordados projetos futuros de ampliação do sistema.

Finalmente, é feita uma retrospetiva de todo o processo envolvente.

8.1 Objetivos concretizados

Neste subcapítulo, analisa-se se os objetivos cumpridos, definidos no subcapítulo 1.3.

A aplicação desenvolvida permite aos utilizadores internos e externos usufruir das máquinas de *vending*, sem contacto. Simultaneamente, oferece a possibilidade de carregamento do cartão da instituição através de um dispositivo móvel, de forma conveniente, sem limitações de horário ou deslocação. Sendo um dos resultados da solução uma aplicação inclusiva, é disponibilizada aos utilizadores invisuais e com dificuldades visuais uma ferramenta útil para utilizar os equipamentos, assim como a oportunidade de usufruir de futuras funcionalidades, no âmbito dos serviços disponibilizados pela FLUP.

A adoção da aplicação por parte dos utilizadores promove a redução da circulação de dinheiro nas instalações e de contactos físicos com as máquinas. Embora as estatísticas da Organização Mundial de Saúde indiquem que a pandemia está a abrandar, é ainda recomendado utilizar métodos de prevenção de infeções (Organização Mundial de Saúde, 2022). O resultado do projeto vai ao encontro das necessidades, não só da instituição, mas também da saúde pública, principalmente se a área de ação for alargada.

A arquitetura modular do sistema foi conseguida, de modo a suportar novos desenvolvimentos, sem grandes atritos. Aplicaram-se regras e boas práticas de desenvolvimento de *software*, com

o objetivo de maximizar a compreensão do sistema por futuros desenvolvedores, através de código de qualidade, eficaz e eficiente. A API de *backend* constitui uma porta de ligação a outros sistemas que poderão interagir com a aplicação.

8.2 Limitações e trabalho futuro

Embora os objetivos supramencionados tenham sido atingidos, é possível identificar algumas limitações da solução.

Inicialmente, foi planeada a implementação de múltiplos métodos de pagamento para o carregamento do cartão da instituição. No entanto, em conjunto com o SI da FLUP, concluiu-se que a integração de um sistema de pagamento por cartão de crédito ou débito não seria comportável, uma vez que as taxas associadas ao serviço seriam demasiado elevadas, ultrapassando os valores da maioria dos produtos disponibilizados pelas máquinas de *vending*.

A integração do método de pagamento MBWay tornou-se uma tarefa árdua, dada a pobre documentação, falta de apoio técnico e desfasamento entre o funcionamento dos servidores de produção e de testes. Adicionalmente, notaram-se quebras frequentes de funcionamento da respetiva aplicação móvel de testes.

Apesar destes obstáculos, o entendimento do funcionamento dos métodos de pagamento por cartão e MBWay permitiu a integração dos mesmos no portal do SI onde é possível, atualmente, pagar propinas e inscrições em eventos.

Da mesma forma, a interação entre a aplicação e as máquinas de *vending* não resultou de um desenvolvimento fluído. Mais uma vez, a documentação disponibilizada não era clara e o apoio técnico limitado. Para tal, foi necessário ultrapassar um longo processo de aprendizagem, de tentativas e erros, de modo a entender as nuances dos mecanismos de comunicação com as máquinas de *vending*.

Estas adversidades influenciaram o tempo disponível para a implementação de funcionalidades, que criaram algumas barreiras e, consequentemente, atrasos no processo de desenvolvimento. Mais concretamente, a funcionalidade de apresentação dos últimos movimentos do utilizador, que se encontra em fase de testes e será disponibilizada aos utilizadores muito em breve.

Relativamente a futuros projetos de ampliação da aplicação, foram destacadas as seguintes funcionalidades:

- Lista de movimentos
- Lista de favoritos
- Notificações da aplicação
- Sistema de pontos e/ou ofertas
- Avaliação da qualidade de um produto
- Reportar falhas do sistema

Atualmente, está a decorrer o processo de integração, na aplicação, do módulo de agendamento, seleção e pagamento de refeições dos bares da FLUP, que será também disponibilizado no portal do SI.

8.3 Apreciação final

Finalmente, é feita uma retrospetiva dos resultados e experiência obtidos, fruto do projeto desenvolvido.

De uma forma geral, considerou-se que o projeto teve um balanço positivo, de acordo com o produto resultante, que responde aos requisitos impostos, sendo que foi possível atender aos problemas descritos neste documento.

O desenvolvimento deste sistema permitiu adquirir conhecimentos de conceção de aplicações para os principais sistemas operativos de dispositivos móveis e publicação de software nas respetivas lojas digitais. Foi possível conhecer os procedimentos de integração com sistemas de pagamento MBWay, cartão de crédito e débito, ainda que os obstáculos encontrados neste processo não tenham sido ultrapassados da forma mais ideal.

A interação com máquinas de *vending* foi uma novidade. Permitiu entender melhor o negócio envolvente e a evolução do processo de compra, desde a seleção de produtos à dispensa. O desenvolvimento obrigou ao contacto com empresa de *vending*, que facultou mecanismos de comunicação com os equipamentos, recursos e explicações de métodos a serem aplicados

Desde o início do projeto, foram dadas a conhecer, ao autor e ao SI, as dificuldades com que os utilizadores se deparam diariamente na utilização de serviços tecnológicos. Reconhece-se que a área de *vending* é uma de muitas que não se encontra preparada para acomodar utilizadores invisuais e com dificuldades visuais. A visão do projeto começou por abordar esta área, visto que as vantagens que os utilizadores da FLUP teriam, poderiam mais tarde ser expandidas para outras zonas. Como foi referido anteriormente, a aplicação desenvolvida constitui também uma interface móvel base para acesso a atuais e futuras funcionalidades fora do contexto de *vending*. A adaptação de ferramentas de acessibilidade, integração de idiomas e serviços são exemplos de fatores de valor para a comunidade académica.

O SI da FLUP possui agora uma ferramenta acessível a um maior número de utilizadores, que poderá ser facilmente expandida.

Referências

Almeida, S. A. (2015). *Consumo de café pelos estudantes de Medicina da Universidade da Beira Interior.*

Apple. (5 de outubro de 2022). *Building accessible apps.* Obtido em 5 de outubro de 2022, de Apple Developer: https://developer.apple.com/accessibility/

Apple. (s.d.). *Swift. A powerful open language that lets everyone build amazing apps.* Obtido em 10 de junho de 2022, de Apple: https://www.apple.com/swift/

Bouch, A. (2011). *3-D Secure: A critical review of 3-D.*

Button Barista. (16 de janeiro de 2017). *Button Barista app*. Obtido em 15 de maio de 2022, de Google Play: https://play.google.com/store/apps/details?id=es.unavarra.azkoyen&hl=pt_PT&gl=US)

BuyOn. (s.d.). *Máquinas de Vending*. Obtido em 6 de junho de 2022, de BuyOn: https://mybuyon.com/maquinas-de-vending/

Caporusso, N., Udenze, K., Imaji, A., Cui, Y., Li, Y., & Romeiser, S. (2019). *Accessibility Evaluation of Automated Vending Machines.*

Chaudhary, P. (2018). *IONIC FRAMEWORK.*

Cloudflare. (s.d.). *How CAPTCHAs work | What does CAPTCHA mean?* Obtido em 30 de setembro de 2022, de Cloudflare: https://www.cloudflare.com/learning/bots/how-captchas-work/

Coges Mobile Solutions srl. (2022 de janeiro de 25). *Pay4Vend*. Obtido em 6 de junho de 2022, de Google Play: https://play.google.com/store/apps/details?id=com.pay4vend.pay4vend

Coges. (s.d.). *Pay4Vend*. Obtido em 6 de junho de 2022, de Pay4Vend: https://www.coges.eu/producto/pay4vend/

Crawford, M., & Benedetto, A. D. (s.d.). *New Products Management.*

Deacon, J. (setembro de 2013). *Model-View-Controller Architecture.* Obtido em 25 de julho de 2022, de John Deacon: http://www.johndeacon.net/john-deacon/articles/model-view-controller-architecture/

Design, O. (17 de junho de 2020). *How to use the button barista app*. Obtido em 6 de junho de 2022, de Youtube: https://www.youtube.com/watch?v=3bGJcnK7luc

Eeles, P. (novembro de 2001). *Capturing Architectural Requirements.* Obtido de The Rational Edge: https://www.researchgate.net/profile/Peter-Eeles-

2/publication/329760910_Capturing_Architectural_Requirements/links/5c197761458515a4c7e8bae1/Capturing-Architectural-Requirements.pdf

Eisenman, B. (1 de dezembro de 2015). *Learning React Native.*

Escudeiro, P., & Bidarra, J. (2008). *Quantitative Evaluation Framework (QEF).*

EVOCA S.p.A. (24 de novembro de 2021). *Coffee APPeal*. Obtido em 15 de maio de 2022, de Google Play: https://play.google.com/store/apps/details?id=com.evocagroup.coffeappeal&hl=pt_PT&gl=US

Google. (s.d.). *Comece a usar o Android com o TalkBack.* Obtido em 5 de outubro de 2022, de Google: https://support.google.com/accessibility/android/answer/6283677?hl=pt

Google. (s.d.). *Flutter architectural overview*. Obtido em 5 de maio de 2022, de Flutter: https://docs.flutter.dev/resources/architectural-overview

Graf, A., & Maas, P. (2008). *Customer value from a customer perspective:.* Wien.

Gupta, D. (22 de agosto de 2022). *How to Design Accessibility App for Visually Impaired?* Obtido em 9 de outubro de 2022, de Appinventiv: https://appinventiv.com/blog/design-accessibility-app-for-visually-impaired/

Haire, A. (s.d.). *What Is Ionic*. Obtido em 7 de junho de 2022, de Ionic: https://ionicframework.com/what-is-ionic

Kagan, J. (4 de janeiro de 2022). *Card-Not-Present Fraud*. Obtido em 28 de julho de 2022, de Investopedia: https://www.investopedia.com/terms/c/cardnotpresent-fraud.asp

Koen, P. (2002). *Fuzzy front end: effective methods, tools, and techniques.*

Lavazza, M. (4 de janeiro de 2021). *Coffee APPeal app og La Radiosa*. Obtido em 5 de junho de 2022, de Youtube: https://www.youtube.com/watch?v=cr32s1FU2n4

Madooei, A. (s.d.). *MVC Pattern*. Obtido em 20 de julho de 2022, de OOSE: https://madooei.github.io/cs421_sp20_homepage/mvc/

Microsoft. (5 de outubro de 2022). *Create Data Transfer Objects (DTOs)*. Obtido em 27 de julho de 2022, de Microsoft: https://learn.microsoft.com/en-us/aspnet/web-api/overview/data/using-web-api-with-entity-framework/part-5

Microsoft. (s.d.). *Why Choose .NET?* Obtido em 8 de junho de 2022, de Microsoft: https://dotnet.microsoft.com/en-us/platform/why-choose-dotnet

Mizinska, M. (26 de outubro de 2021). *3DS 2.0: Thoughts, Concerns and Threats*. Obtido em 26 de setembro de 2022, de Straal: https://straal.com/what-is-3ds-2-0-thoughts-concerns-and-threats/

Newis. (s.d.). *Coffee APPeal*. Obtido em 15 de maio de 2022, de Newis: https://newis.evocagroup.com/en/app/coffee-appeal

Nilsson, J. (2006). *Applying Domain-Driven Design and Patterns: With Examples in C# and .NET.*

Obinna, O. (7 de abril de 2020). *How does JIT and AOT work in Dart?* Obtido em 5 de junho de 2022, de Medium: https://learnlanga.medium.com/how-does-jit-and-aot-work-in-dart-cab2f31d9cb5

Organização Mundial de Saúde. (14 de setembro de 2022). *COVID-19, Monkeypox & Other Global Health Issues Virtual Press conference transcript - 14 September 2022.* Obtido em 9 de outubro de 2022, de World Health Organization: https://www.who.int/publications/m/item/covid-19--monkeypox---other-global-health-issues-virtual-press-conference-transcript---14-september-2022

Peter Koen. (s.d.). Obtido de Stevens Institute of Technology: https://faculty.stevens.edu/pkoen

QFD Online. (11 de dezembro de 2007). *Free QFD Templates.* Obtido em junho de 5 de 2022, de QFD Online: http://www.qfdonline.com/templates/

Ramel, D. (3 de dezembro de 2020). *Microsoft: 'We Do Not Plan to Evolve Visual Basic as a Language'*. Obtido em 15 de junho de 2022, de Visual Studio Magazine: https://visualstudiomagazine.com/articles/2020/03/12/vb-in-net-5.aspx

React Native. (19 de janeiro de 2022). *Learn the Basics*. Obtido em 6 de junho de 2022, de React Native: https://reactnative.dev/docs/tutorial

Ripkens, B. (28 de novembro de 2014). *Ionic: An AngularJS based framework on the rise*. Obtido em 5 de junho de 2022, de Codecentric Blog: https://blog.codecentric.de/en/2014/11/ionic-angularjs-framework-on-the-rise/

Saaty, R. (1987). *The analytic hierarchy process—what it is and how it is used.*

Schreiner, L. C., Mello, A. M., & Nascimento, P. T. (2015). *O Fuzzy Front End Integrado Na Gestão do Desenvolvimento De Embalagens.*

Sciandra, L. (16 de abril de 2019). *The New React Native Architecture Explained: Part Four*. Obtido em 5 de junho de 2022, de Formidable: https://formidable.com/blog/2019/lean-core-part-4/

Skuza, B., Mroczkowska, A., & Włodarczyk, D. (10 de maio de 2021). *Flutter vs. React Native – What to Choose in 2022?* Obtido em 5 de junho de 2022, de Droids On Roids: https://www.thedroidsonroids.com/blog/flutter-vs-react-native-what-to-choose-in-2021

Suragch. (9 de dezembro de 2018). *First steps with Flutter- Part 1: Exploring widgets*. Obtido em 5 de junho de 2022, de Pusher: https://pusher.com/tutorials/flutter-widgets/

Thomas, G. (12 de dezembro de 2019). *What is Flutter and Why You Should Learn it in 2020*. Obtido em 5 de junho de 2022, de freeCodeCamp: https://www.freecodecamp.org/news/what-is-flutter-and-why-you-should-learn-it-in-2020/

Wu, W. (2018). *React Native vs Flutter, cross-platform mobile.*

Zafar, A. (20 de dezembro de 2017). *How to design accessible mobile experiences for the blind*. Obtido em 7 de outubro de 2022, de Willowtree: https://www.willowtreeapps.com/craft/how-to-design-accessible-mobile-experiences-for-the-blind

Apêndice A

Métricas QEF

		Wfk – Fullfilment (%)		
Requirement	**Metric Evaluation**	**0**	**50**	**100**
FFU01 - Início de sessão através do controlador de domínio (utilizadores internos)	Os utilizadores da FLUP conseguem autenticar-se com as as credenciais da instituição	Funcionalidade não disponível	-	Funcionalidade totalmente acessível
FFU02 - Criação de conta (utilizadores externos)	Os utilizadores externos á FLUP conseguem registar-se na plataforma	Funcionalidade não disponível	-	Funcionalidade totalmente acessível
FFU03 - Consulta de informações de perfil	Os utilizadores conseguem verificar informações relativas ao seu perfil	Funcionalidade não disponível	-	Funcionalidade totalmente acessível
FFU04 - Recuperação de credenciais de acesso (utilizadores externos)	Os utilizadores externos conseguem recuperar as suas credenciais de acesso	Funcionalidade não disponível	-	Funcionalidade totalmente acessível

Figura 54 – Métricas de avaliação do fator Utilizador, da dimensão Funcionalidade

		Wfk – Fullfilment (%)		
Requirement	**Metric Evaluation**	**0**	**50**	**100**
FFS01 - Encerrar sessão em todos os dispositivos	O utilizador consegue terminar a sessão em todos os dispositivos em que iniciou sessão através da sua conta	Funcionalidade não disponível	-	Funcionalidade totalmente acessível

Figura 55 - Métricas de avaliação do fator Segurança, da dimensão Funcionalidade

		Wfk – Fullfilment (%)		
Requirement	**Metric Evaluation**	**0**	**50**	**100**
FFP01 - Carregamento de saldo de conta	O utilizador consegue carregar a sua conta com saldo, através dos métodos de pagamento disponibilizados na plataforma	Funcionalidade não disponível	A funcionalidade é acessível, através de um método de pagamento único	A funcionalidade é acessível e oferece múltiplos métodos de pagamentos
FFP02 - Consultar saldo de conta	O utilizador consegue consultar o seu saldo da plataforma	Funcionalidade não disponível	-	Funcionalidade totalmente acessível

Figura 56 - Métricas de avaliação do fator Pagamento, da dimensão Funcionalidade

		Wfk – Fullfilment (%)		
Requirement	**Metric Evaluation**	**0**	**50**	**100**
FFV01 - Ativar/desativar perfil de vending	O utilizador consegue ativar/desativar um perfil de vending associado a um conjunto de termos de utilização do serviço	Funcionalidade não disponível	-	Funcionalidade totalmente acessível
FFV02 - Ler identificação de uma máquina de vending	O utilizador consegue identificar uma máquina de vending através da leitura de um código QR ou de uma etiqueta NFC	Funcionalidade não disponível	O utilizador consegue identificar uma máquina através de um dos métodos mencionados	Funcionalidade totalmente acessível
FFV03 - Consultar famílias de produtos	O utilizador consegue visualizar as famílias de produtos de uma máquina de vending	Funcionalidade não disponível	-	Funcionalidade totalmente acessível
FFV04 - Consultar produtos de uma máquina	O utilizador consegue visualizar os produtos de uma máquina de vending	Funcionalidade não disponível	-	Funcionalidade totalmente acessível
FFV05 - Personalização de produtos	O utilizador consegue, através da aplicação, personalizar o nível de açúcar e opção de copo de algumas bebidas	Funcionalidade não disponível	O utilizador apenas consegue personalizar uma das opções mencionadas	Funcionalidade totalmente acessível
FFV06 - Solicitação de um produto	O utilizador consegue solicitar a dispensa de um produto de uma máquina de vending previamente selecionada	Funcionalidade não disponível	O utilizador consegue efetuar a ação, mas com alguns obstáculos	O utilizador consegue efetuar a ação, sem contrangimentos

Figura 57 - Métricas de avaliação do fator *Vending*, da dimensão Funcionalidade

		Wfk – Fullfilment (%)		
Requirement	**Metric Evaluation**	**0**	**50**	**100**
UU01 - O layout da interface é compreensível e de fácil navegação	O utilizador consegue, intuitivamente, navegar pelos menus da aplicação e compreender os elementos visuais da mesma	A aplicação é desorganizada e exige um elevado grau de conhecimento para ser utilizada	A aplicação satisfatória, mas apresenta alguns inconvenientes na compreensão e navegação da UI	A funcionalidade do requisito é totalmente cumprida
UU02 - As funcionalidades da aplicação são de fácil acesso	Os utilizadores conseguem facilmente alcançar as funcionalidades que pretendem	As funcionalidades não são de fácil acesso	As funcionalidades são acessíveis, mas com algum esforço	A funcionalidade do requisito é totalmente cumprida
UU03 - Existe congruência entre o software das máquinas de vending e a aplicação	A interface e funcionalidades das máquinas de vending assemelha-se às da aplicação	Existe uma grande divergência entre os dois softvare	As máquinas de vending e aplicação apresentam algumas semelhanças	Existe uma relação forte entre o software das máquinas de vending e da aplicação
UU04 - O tempo de resposta das ações executadas pelo utilizador é baixo	As ações dos utilizadores resultam em respostas rápidas que não interferem na experiência de utilização	O sistema demora a responder às ações do utilizador	O tempo de resposta é variável	O sistema responde rapidamente às ações do utilizador
UL01 - O sistema suporta os idiomas Português (Portugal) e Inglês	-	Não	-	Sim
UL02 - É utilizada terminologia da área de negócio de vending	Os termos utilizados na aplicação enquadram-se no contexto do negócio de vending	A terminologia utilizada não se adequa ao contexto	Alguns termos utilizados não se enquadram no contexto	A funcionalidade do requisito é totalmente cumprida
UA01 - A aplicação está adaptada a utilizadores invisuais e com dificuldades visuais	Os utilizadores invisuais ou com dificuldades visuais conseguem utilizar todas as funcionalidades da aplicação sem constrangimentos	A aplicação não suporta ferramentas de acessibilidade	Os utilizadores invisuais e com dificuldades visuais têm alguns contrangimentos ao utilizador a aplicação	A aplicação suporta ferramentas de acessibilidade que permitem aos utilizadores invisuais e com dificuldades visuais usufruir de todas as funcionalidades da aplicação sem contrangimentos

Figura 58 - Métricas de avaliação da dimensão Usabilidade

		Wfk – Fullfilment (%)		
Requirement	**Metric Evaluation**	**0**	**50**	**100**
SA02 - A aplicação está disponível para o sistema Android e iOS	-	Não	-	Sim
SM01 - O sistema trata e regista erros e falhas	Em caso de cenários que impedem o bom funcionamento do fluxo dos sistema, são armazenados dados que, posteriormente, permitem localizar e entender o problema. Adicionalmente, o utilizador é devidamente informado de qualquer situação anómala.	Não	A funcionalidade é cumprida mas nem sempre o utilizador é informado com uma resposta concreta	O sistema cumpre todos os requisitos impostos
SM02 - O sistema pode ser facilmente ampliado, introduzindo novas funcionalidades	O sistema foi concebido de forma modular, com vista a receber novas atualizações	Não	-	Sim

Figura 59 - Métricas de avaliação da dimensão Suportabilidade