APLICATIVO MÓVEL PARA A INDÚSTRIA DO VAREJO

Patrick Estevam Lopes – 31613845@mackenzista.com.br
Prof. Dr. Paulo Lopes (Orientador) – paulo.lopes@mackenzie.br

RESUMO

A transformação digital já é realidade no mercado do varejo e a busca por ferramentas para ofertar serviços e produtos de qualidade aos clientes tem se intensificado fortemente. A aplicação de tecnologias como inteligência artificial e aprendizado de máquina, unidas aos atuais meios disponíveis aos clientes apoiam a proposta de valor oferecida e agrega maior visibilidade no processo de venda. Porém, é importante que haja uma mudança no conceito de compra para que seja possível unir dois cenários: a diminuição do tempo gasto no processo e a melhora na experiência do cliente. Baseado nestes pontos, este trabalho busca desenvolver um aplicativo voltado a vertical do varejo, focada na venda de bens de consumo diários, destacando aqui principalmente os supermercados e hipermercados. O aplicativo em questão será desenvolvido sob o conceito de suporte multiplataforma, focando nos principais sistemas operacionais do mercado de dispositivos móveis: Android e IOS. Para que seja possível modificar o processo de compra tradicional, este trabalho tem por objetivo possibilitar, através do rápido escaneamento do código de barras dos produtos dispostos nas prateleiras dos estabelecimentos, a captação das informações de cada produto, propiciando ao usuário criar sua lista dinâmica de compras e ao final realizar o pagamento da compra total pelo próprio aplicativo via a integração com um API de pagamento.

Palavras-chave: Aplicativo, API, Pagamento.

MOBILE APPLICATION FOR RETAIL INDUSTRY

ABSTRACT

Digital transformation is already a reality in the retail market and the search for tools to offer quality services and products to customers has intensified strongly. The application of technologies such as artificial intelligence and machine learning, plus the current means available to customers, support the value proposition offered and add greater visibility to the sales process. However, it is important to have a change in the purchase concept so that it is possible to join two scenarios: the time spent reduction and the customer experience improvement. Based on these points, this work seeks to develop an application aimed at the

retail vertical, focused on the daily consumer goods sales, supermarkets and hypermarkets standing out. The application in question will be developed under the concept of cross-platform support, focusing on the main operating systems in the mobile device market, namely: Android and IOS. In order to modify the traditional purchase process, this work aims to enable, through the rapid scanning of the products barcode displayed on the establishments shelves, the information capture of each product, allowing the user to create a dynamic shopping list and at the end, pay the total purchase by the application via the integration with a payment API.
Keywords: Application, API, Payment.

1 INTRODUÇÃO

Atualmente, é possível encontrar uma imensidão de inovações para *smartphones*, pois o mercado demanda um crescente desenvolvimento de aplicativos móveis, baseado na necessidade de que as pessoas precisam se conectar ao mundo digital. Segundo Ortis (2019), pensando neste contexto, os dispositivos móveis estão cada vez mais avançados, com forte emprego de tecnologia, e suas funções não mais se limitam em apenas receber e efetuar chamadas. Estes dispositivos visam agregar valor aos usuários, rompendo fronteiras e aproximando as pessoas. Além de que, os *smartphones* se tornaram ferramenta de propaganda, vendas e propagação de notícias, sobretudo caminhando com a transformação digital que está presente em todos os setores do mercado e da sociedade.

A transformação digital consiste em um processo pelo qual as organizações estão se adequando ou mudando drasticamente seus modelos de negócio, com objetivo de explorar todo o potencial das tecnologias digitais e com isso conquistar a vantagem competitiva (Sampaio, 2018). Contudo, a tecnologia não é o único aspecto a se considerar nessa jornada. Essa transformação afeta a cultura das organizações, altera o ecossistema organizacional, mudam as crenças, os conceitos de gerenciamento e de suas estratégias. (HININGS, GEGENHUBER, GREENWOOD, 2018; MATT *el al*., 2015).

Segundo a análise de Levy e Weitz (2011), a qual considera que varejistas, são todas as instituições, que tem por atividade principal, a venda de produtos e serviços, destinados ao uso pessoal ou familiar, ou ainda, aos consumidores com a conveniência de tempo e lugar para a aquisição de produtos, é possível identificar, que a consolidação desta classificação só ocorre a partir do reconhecimento destas como empresas que possuem alto nível de satisfação do cliente e que trabalham para melhorar este ponto, recorrentemente.

Segundo a Sociedade Brasileira de Varejo e Consumo (2021), o orçamento dos varejistas brasileiros para transformação digital, aumentou em 87%, o que representa cerca de 0,73% do faturamento bruto destas empresas. Como resultado, 74% das companhias tiveram aumento no faturamento com vendas. A SBVC (Sociedade Brasileira de Varejo e Consumo) monitorou as 300 maiores empresas de varejo no país. Dentre elas, 33% tinham faturamento até R$ 500 milhões, 19% até R$ 1 bilhão e 48% acima disso.

Com a transformação digital, diversas mudanças estão ocorrendo no processo de consumo e que impulsionam novos seguimentos, estreitando o relacionamento de empresas e consumidores. Segundo Soares (2019), desde 2018 as redes varejistas multicanais estão realizando vendas em suas lojas através de aplicativos, ampliando a oferta de produtos através desta plataforma, não somente através de lojas online, como também em pontos físicos.

Se por um lado, o varejo está imerso nesta corrida para a inovação tecnológica, a fim de expandir seu faturamento através de técnicas multicanais, a indústria de tecnologia, a qual desenvolve as ferramentas para serem empregadas no varejo, não para de triplicar seu faturamento mundial. O número de aplicativos móveis oferecidos aos consumidores aumenta de maneira exorbitante. Em 2018, os consumidores poderiam escolher entre quase seis milhões de aplicativos nas lojas do Google ou Apple. Relatórios da indústria indicam que mais de 90% dos aplicativos móveis começam como gratuitos e mais de 90% dos lucros dos aplicativos móveis vêm de aplicativos que começaram como gratuitos. (WU, TEUNTER, ZHU, 2019).

Segundo a empresa de análise de mercado Gobacklog (2018), em estudo da CVA Solutions, cerca de 20% da população brasileira utiliza aplicativos para fazer compras em supermercados. Em análise focada no grupo GPA (Grupo Pão de Açúcar), 77% de seus clientes utilizam o aparelho celular enquanto estão nas lojas. Com base nestes dados, é possível identificar o potencial pouco explorado pela maioria das empresas deste setor, além da força que o emprego desta estratégia de inovação tecnológica, pode influenciar e otimizar a experiência do cliente. Um exemplo da aplicação benéfica da tecnologia em lojas deste setor é a implantação de um sistema chamado *Self-Checkout*, método desenvolvido pelo grupo GPA e aplicado em certas lojas da rede. Neste sistema, o cliente pode finalizar a compra sozinho, com mais rapidez e menor proporção de filas. Esta tecnologia já foi explorada fora do Brasil, mas aqui, pouco adotada pelos supermercadistas do setor. A tecnologia de *Self-Checkout* proporciona uma utilização de espaço, três vezes menor do que os caixas tradicionais, o que mostra que a adoção deste sistema traz benefícios tanto para os clientes como para os funcionários do grupo GPA. Esta redução ocorre primariamente de maneira proposital a fim de sugerir ao cliente uma nova forma de finalizar as compras e com isso aferir se o modelo terá

aderência junto ao consumidor, desta maneira após está validação, é possível atuar com a redução do número de caixas tradicionais.

Segundo Curry (2021), em estudo realizado pela Business Of Apps, mesmo que esses aplicativos sejam gratuitos, o mercado tem tamanho gigantesco, com mais de 100 bilhões de dólares em 2019. A maior categoria de aplicativos são os jogos para celular, cujas receitas atingiram cerca de 70 bilhões de dólares em 2018, ultrapassando os jogos de videogame e jogos para computadores. A monetização dos aplicativos ocorre principalmente de duas maneiras. Dentre elas, a mais praticada pelo mercado é através da venda de espaço publicitário em uma versão gratuita do aplicativo. Já a segunda forma de monetização é comercializar uma versão paga ou uma modalidade de compra no aplicativo que oferece recursos adicionais pagos. Para que esta abordagem continue gerando frutos, os criadores de aplicativos constantemente estão avaliando novos modelos de negócios e estratégias de *marketing*, a fim de captar mais usuários a cada nova versão ou lançamento.

Um aplicativo dedicado ao comércio varejista tende a incrementar os pontos positivos na relação entre a empresa e o cliente pois torna a experiência do consumidor com a companhia muito mais agradável. A expectativa é que o cliente sinta-se muito confortável ao fazer compras naquele estabelecimento e seja fidelizado, podendo até mesmo chegar a pagar mais por se sentir recompensado. E seguindo este viés de elaborar um aplicativo voltado a interação do usuário e a fim de dinamizar e facilitar o processo de compra em um supermercado, este trabalho irá descrever os principais pontos que devem ser levados em conta para um correto desenvolvimento de um aplicativo focado nesta área do varejo e de maneira disruptiva empregar tecnologia em processos clássicos cotidianos, presentes nesta vertical de negócio.

O foco será o desenvolvimento do aplicativo voltado ao atendimento em multiplataforma, o qual, através da interação do usuário, poderá através de seu celular, coletar o código de barras de cada produto das prateleiras do supermercado, identificando o produto em questão e respectivo valor. Desta forma ao coletar todos os produtos que deseja, o usuário poderá finalizar a compra pelo aplicativo e efetuar o pagamento.

2 REVISÃO DA LITERATURA

Diversas pesquisas vêm sendo realizadas a respeito do impacto da implantação de aplicativos móveis e inteligência artificial, na indústria do varejo. Dentre elas, as principais questões-chave tratadas são: quais plataformas de desenvolvimento de aplicativos móveis terão maior aderência aos consumidores no resultado final da aplicação, análise sob a modificação

do processo de compra via *App* e identificar o quão benéfico seria o emprego destes recursos, no cotidiano de um supermercado, baseado na análise de outros aplicativos existentes no mercado.

2.1 PLATAFORMAS DE DESENVOLVIMENTO DE APLICATIVOS MÓVEIS

Segundo o IDC (2021), o mercado de consumo de *smartphones* superou os 380 milhões de dólares em vendas de novos aparelhos no ano de 2020. Dentre esta análise, a Apple, no último trimestre, liderou o mercado alcançando o faturamento de mais 90 milhões de dólares, representando um aumento de 22,2% em relação ao ano de 2019. Analisando estes números e de acordo com a Tabela 1, nota-se que excluindo a Apple com seu sistema operacional IOS, o restante dos fabricantes tem como base o sistema operacional Android, sendo possível concluir então, que mais de 70% dos usuários de *smartphones*, no mundo, possuem dispositivos que atuam com base neste sistema móvel. Segundo Meyer (2015), o sistema operacional Android foi desenvolvido pela então Android Inc. em 2003, empresa a qual, foi adquirida pelo Google em 2005, por 50 milhões de dólares.

Tabela 1 – Top 5 principais empresas de *smartphones*, remessas mundiais, participação de mercado e crescimento ano após ano, quarto trimestre de 2020 (remessas em milhões de unidades).

TOP 5 PRINCIPAIS EMPRESAS DE *SMARTPHONES* – Q4/2020					
Companhia	2020 Q4 Volumes de Remessa	2020 Q4 Quota de Mercado	2019 Q4 Volumes de Remessa	2019 Q4 Quota de Mercado	Ano por Ano
Apple	90,1	23,4%	73,8	19,9%	22,2%
Samsung	73,9	19,1%	69,5	18,8%	6,2%
Xiaomi	43,3	11,2%	32,8	8,9%	32,0%
OPPO	33,8	8,8%	30,6	8,3%	10,7%
Huawei	32,3	8,4%	56,2	15,2%	-42,4%
Outras	112,4	29,1	107,1	28,9%	5,0%
Total	**385,9**	**100%**	**369,9**	**100%**	**4,3%**

Fonte: IDC Quarterly Mobile Phone Tracker (2021)

Considerando os dois maiores *players* que atuam com serviço de distribuição digital de aplicativos do mercado, a App Store e Google Play, juntos geraram uma receita de 111 bilhões de dólares em compras, contabilizando compras dentro dos aplicativos, assinaturas e downloads de *Apps* em versões superiores. Segundo Chan (2021), pesquisadora de estratégia e dados da Sensor Tower Intelligence, a receita da App Store chegou em 2020 a 72,3 bilhões de dólares, registrando um aumento de mais de 30% em relação a 2019. A App Store manteve seu faturamento 87,3% maior do que a Google Play, a qual também atingiu crescimento de 30%, em comparação com o ano anterior, registrando um faturamento de 38,6 bilhões de dólares.

Levando em conta essa necessidade de atender múltiplos sistemas operacionais e analisando os *softwares* de desenvolvimento de aplicativos móveis, identifica-se que a melhor decisão será atuar sob a ótica de aplicações híbridas. O desenvolvimento de um aplicativo híbrido é mais rápido e também mais barato, pois o código é desenvolvido apenas uma vez e pode ser distribuído em ambas as plataformas, Android e IOS, e até mesmo para plataformas *web*.

Segundo Madureira (2017), o aplicativo híbrido pode ser construído utilizando as linguagens HTML5, CSS, C# e JavaScript, assim como um site para dispositivos móveis. O código completo é então copiado para um container, possibilitando a integração entre as funcionalidades que o dispositivo oferece. Com a utilização de um *framework* específico, o código desenvolvido com tecnologia *web* pode ser compilado para diferentes plataformas. A manutenção acaba sendo mais barata, pois a mão de obra é mais genérica e simples de ser encontrada no mercado.

Outro ponto de suma importância, é que o tempo de desenvolvimento diminui se comparado a outros métodos de desenvolvimento de aplicativos, como por exemplo, os *Apps* nativos. Não há a necessidade de replicação de código para cada sistema operacional, facilitando assim o trabalho e gerenciamento do processo criativo, além de que os testes podem ser gerais, ou seja, as plataformas permitem que o programador teste a aplicação rodando o código em simuladores de Android e IOS ao mesmo tempo. Desta forma é possível avaliar o desempenho, em tempo real. Tendo em vista as diversas opções existentes no mundo das plataformas de desenvolvimento de *Apps*, para este presente trabalho, a plataforma escolhida foi o Xamarin.

O desenvolvimento de plataforma cruzada nasceu da ideia de tornar possível a execução de um único aplicativo em ambos os sistemas operacionais sem ter que reescrever todo o código do zero. De acordo com a Statista (2021), em 2020 o mercado de software multiplataforma deve ultrapassar 8,5 bilhões de dólares. A introdução da plataforma cruzada cresceu com o advento

do React Native do Facebook em 2015. Hoje, existem muitos *frameworks* para o desenvolvimento de plataforma cruzada, mas os mais populares e eficazes são: PhoneGap, Xamarin, React Native e Flutter . Como já citado, diversos pontos são benéficos em adotar o uso de plataformas cruzadas de desenvolvimento, mas vale ressaltar os benefícios do emprego de *frameworks* cruzados por meio dos seguintes itens:

a) Economia de tempo: A rapidez é gigantesca, pois os desenvolvedores podem lançar mais rapidamente os aplicativos híbridos utilizando uma plataforma móvel cruzada.
b) Economia de custos: Com menor tempo gasto para o desenvolvimento dos aplicativos, menor será o custo com a equipe de desenvolvedores e por consequência, menor custo final.
c) Base de código única: Estas plataformas podem permitir que os desenvolvedores utilizem uma base de código única em vários sistemas operacionais e aplicativos de forma eficaz, o que beneficia na manutenção e revisão da base de código da melhor maneira possível.

Dentre as plataformas citadas, destaca-se a evolução de uma em específico, o Flutter, esta plataforma se tornou popular principalmente por ser usada em grandes empresas do mercado de tecnologia, como o Facebook, Airbnb e Instagram. Por meio da Figura 1, nota-se o uso destas plataformas e o grau de evolução no decorrer dos anos, evidenciando picos de crescimento, principalmente do Xamarin, plataforma popular dentre os desenvolvedores mais familiarizados com o mundo Microsoft e aqueles que possuem aplicações hospedadas no serviço de nuvem pública, Azure.

Figura 1 - Popularidade de estruturas para desenvolvimento de plataforma cruzada nas consultas de pesquisa do Google.

Fonte: Statista worldwide cross mobile platform (2021).

2.1.1 Xamarin

Em meados do ano 2000, um grupo de desenvolvedores liderados por Miguel de Icaza, estavam desenvolvendo uma versão do .NET para o Linux. Mais à frente, este projeto ficou conhecido por Mono. (HERMES, MAZLOUMI, 2019). Com o decorrer do desenvolvimento e aprimoramento deste projeto, na fase seguinte, o mesmo alcançou a consolidação de atuação em plataformas cruzadas. Fase a qual, foi oficialmente chamada de Xamarin, em 2011. Ainda nesta fase a equipe de desenvolvedores alcançaram a atuação também em plataformas móveis, como Android e IOS. Em 2016, a Microsoft firmou um acordo de aquisição da plataforma, em um valor estimado de 400 milhões de dólares.

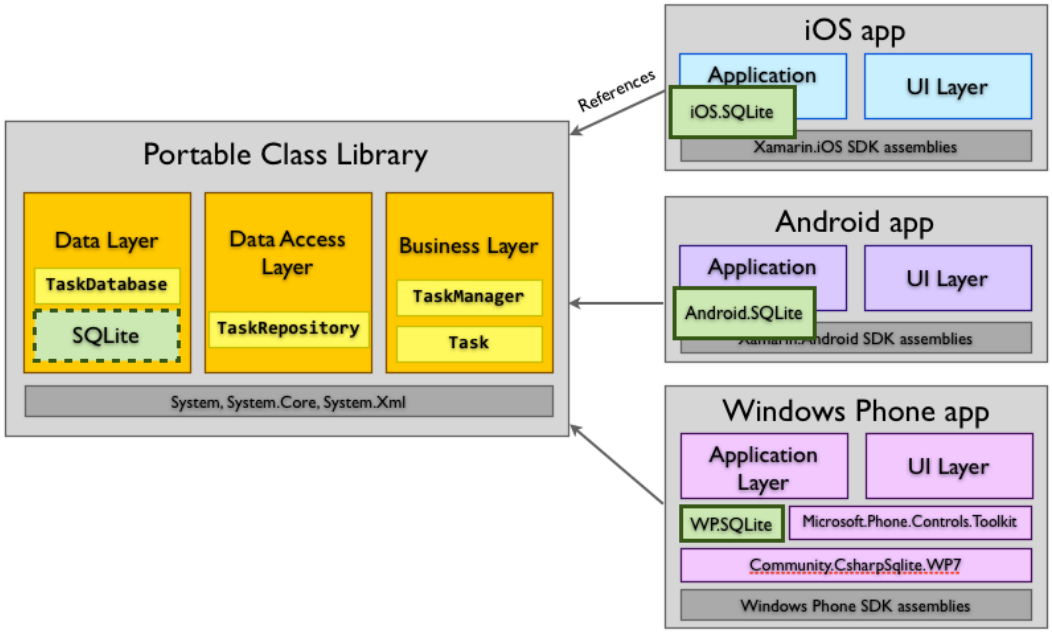
Segundo Hermes e Mazloumi (2019), o trabalho de toda a equipe ajudou a convencer Microsoft, uma das maiores empresas de software do mundo, do valor de códigos fonte abertos e de plataformas cruzadas. A aquisição do Xamarin pela Microsoft ajudou a consolidar a plataforma dentro do Visual Studio. O Xamarin é agora o carro-chefe do desenvolvimento de aplicativos móveis multiplataforma da Microsoft e atualmente, é desenvolvido e suportado como um dos produtos oferecidos pela Microsoft, com grande integração com o IDE do Visual Studio, e ainda possui seu próprio IDE, denominado Xamarin Studio, que está disponível para MacOS. Esta integração com a Microsoft possibilitou ao usuário a opção de atuar com aplicações baseadas em .NET, o que resultou na divisão da plataforma em duas frentes: Xamarin.Android e Xamarin.IOS. Cada uma das soluções criadas tinha ferramentas dedicadas para cada sistema operacional, na interface de usuário. Além disso, o desenvolvedor passou a ter a opção de escolher o método de compartilhamento do código, ao decorrer do desenvolvimento do *script* raiz, podendo atuar de duas maneiras:

a) Projeto Compartilhado - Compilação do código usando as bibliotecas da plataforma de maneira conjunta (Android ou iOS);
b) Bibliotecas de classes portáteis - Projeto compilado de forma independente, contendo bibliotecas comuns para ambas plataformas.

A abordagem mais tradicional para criar aplicativos usando Xamarin é o método de desenvolvimento nativo. Desta forma, uma parte do código implementado será compartilhado entre as plataformas. Estas partes são constituídas especificamente por implementações na camada de regras de negócio, acesso a dados e clientes de API (Interface de Programação de Aplicação). Por meio da Figura 2, é possível identificar outro método de desenvolvimento citado neste presente trabalho, mas que, observado através da ótica do Xamarin, merece uma explicação. O método de desenvolvimento híbrido ou código compartilhado refere-se a camada

de acesso a dados local, como um banco de dados, serviços em nuvem, camada de regras de negócios e seus modelos, baseado no conceito de compartilhamento de arquivos. Essa abordagem faz o compartilhamento direto dos arquivos do código entre todos os projetos que fazem referência ao projeto principal *Shared Project*, conforme a Figura 2. Outro aspecto relevante é que o código específico a ser implementado restringe-se a camada de aplicação, interface e acesso aos recursos do dispositivo, através de SDK (Kit de Desenvolvimento de Software) específicos de cada plataforma.

Figura 2 - Diagrama da plataforma Xamarin, abordagem nativa.



Fonte: Portal de documentação da Microsoft (2021).

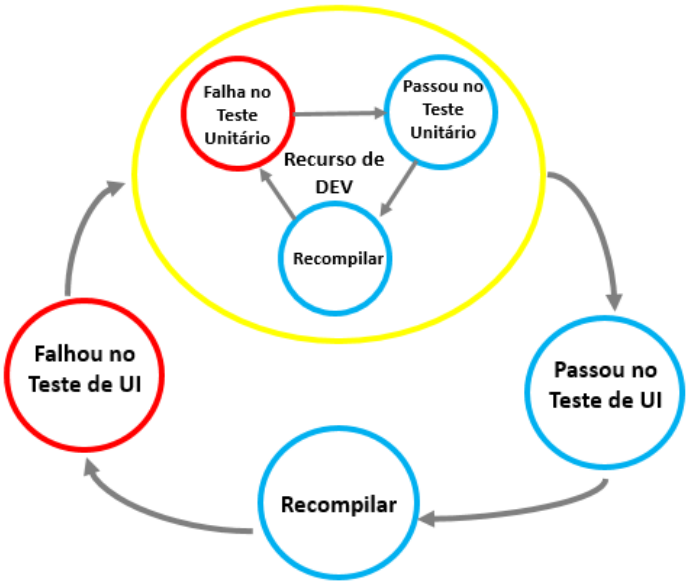
2.2 IDENTIFICAÇÃO DE PROCESSO PADRÃO DE TESTES PARA APLICATIVOS MÓVEIS

Segundo o portal de desenvolvedores do Google (2020), os testes de *software* são maneiras de validar as regras de atuação que foram implementadas no código, além de ter como objetivo, a verificação de correto comportamento do programa, segundo premissas inicias. Outro ponto, é identificar através de testes, se as requisições estão alcançando os resultados esperados em suas saídas. Quando um aplicativo é interativo, ou seja, o usuário, por meio de ações, irá receber ou visualizar resultados, é de suma importância que os testes esgotem todas as interações possíveis, incluindo interações-padrão, entradas inválidas e casos em que os

recursos não estão disponíveis. E adotar ciclos comuns de fluxo de processos é sempre vital, como podemos notar na Figura 3.

O fluxograma de ciclos de trabalho, mostrado na Figura 3, apresenta uma série de ciclos interativos alinhados, em um ciclo longo, lento e controlado pela interface do usuário, sendo possível o teste da integração de unidades de código, através das próprias unidades usando ciclos de desenvolvimento mais curtos e rápidos. Esse conjunto de ciclos continua até que o *App* satisfaça todos os casos de uso.

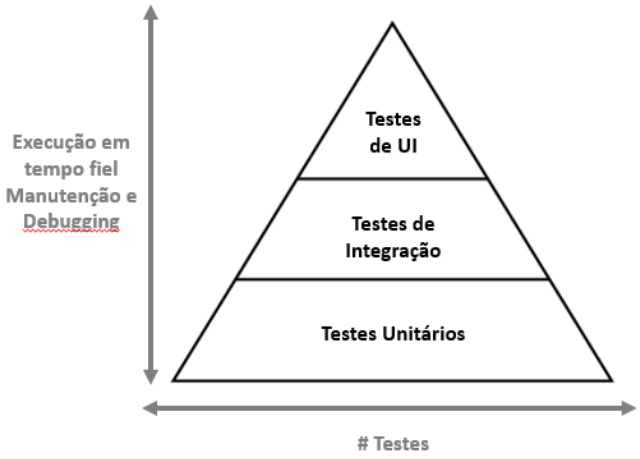
Figura 3 - Os dois ciclos associados ao desenvolvimento interativo voltado para testes.



Fonte: Modificado de Portal de desenvolvedores Google (2020)

Depois de configurar o ambiente de teste, será necessário programar testes que avaliem a funcionalidade do aplicativo. Um conceito bastante usado para atingir resultados e atingir pontos intrínsecos da aplicação é atuar com os níveis de pirâmide de testes. A pirâmide de testes apresentada na Figura 4, ilustra as três principais categorias de teste: pequeno, médio e grande. Os testes pequenos, são testes de unidade, de uma classe por vez, que validam o comportamento do aplicativo. Os testes médios, são testes de integração que validam interações entre níveis da pilha dentro de um módulo ou interações entre módulos relacionados. E os testes grandes, são testes de ponta a ponta, que validam as jornadas do usuário, alcançando vários módulos da aplicação. Mais à frente neste presente trabalho, a aplicação destes conceitos será mostrada na seção de Metodologia escolhida.

Figura 4 - Pirâmide de testes, mostrando as três categorias inclusas no conjunto de testes do *App*.



Fonte: Modificado de Portal de desenvolvedores Google (2020)

Ainda segundo o Google (2020), à medida que ocorre a subida nos níveis da pirâmide, dos testes pequenos até os testes grandes, a assertividade de cada teste se eleva. Todavia, o tempo de execução e o esforço para manutenção e depuração, também se tornam maiores. Logo, será benéfico que se programe mais testes de unidade que testes de integração, e por consequência, que sejam aplicados mais testes de integração que testes de ponta a ponta. Apesar de que a proporção de testes para cada categoria possa variar de acordo com os casos de uso, em geral, é recomendado a seguinte proporção para cada categoria: 70% pequenos, 20% médios e 10% grandes.

3 METODOLOGIA

A pesquisa documental elaborada com base nas seções anteriores sustenta o desenvolvimento de um aplicativo, o qual será aplicado em supermercados, a fim de facilitar o processo de compra padrão consolidado na vertical de negócio do varejo tendo como premissa a melhora da experiência do consumidor, que já está cansado de colocar produtos em um carrinho de compras, muitas vezes, não conseguir identificar o valor de certos produtos dispostos nas prateleiras e principalmente enfrentar grandes períodos em filas de caixas de supermercados.

O aplicativo em questão irá promover a conexão de todos os pontos do processo de compra, em um só lugar. Através de um software que será desenvolvido na plataforma Xamarin, o consumidor, utilizando o seu *smartphone*, seja ele baseado em Android ou IOS, poderá baixar o aplicativo e, após um simples cadastro, terá acesso a uma tela de leitura de código de barras dos produtos existentes nas prateleiras e já previamente cadastrados no banco de dados da

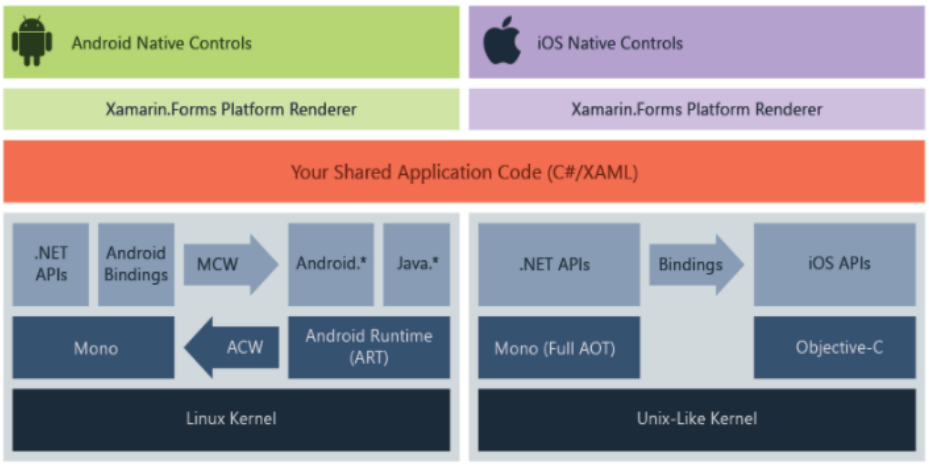
solução. Esta leitura ocorrerá por meio da câmera do celular. A partir da identificação do produto, o consumidor poderá visualizar o respectivo valor e descrição, e, por meio de botões interativos, poderá inserir este produto em sua lista dinâmica, tela subsequente a tela de leitura. Nesta tela, o usuário poderá excluir o produto do carrinho virtual, caso julgue necessário.

Cada produto terá seu valor unitário apresentado em um local específico, ao lado do valor total, caso a quantidade do mesmo seja maior que um. Ainda nesta tela, será possível regressar a tela de leitura e continuar adicionando mais produtos diferentes. O usuário poderá finalizar sua compra através de um botão, o qual o irá direcionar a um API (Interface de Programação de Aplicação) de pagamento. Neste momento, o usuário poderá efetuar o pagamento da compra total realizada no aplicativo.

Para que todas as *features* existentes neste aplicativo sejam de fato implementadas, a programação será baseada no conceito de bibliotecas, as quais, já possuem um padrão de integração com a plataforma Xamarin. Como já citado neste trabalho, esta plataforma possibilita a programação em blocos, os quais se aplicarão para cada sistema operacional desejado. A Figura 5 apresenta de que forma a plataforma irá separar estes blocos, formando uma arquitetura de comunicação entre o código desenvolvido e os sistemas operacionais.

Segundo Balivo (2016), com este tipo de conceito, é possível reaproveitar o código em maior escala, tornando a proposta da plataforma muto mais atraente aos desenvolvedores e como consequência, melhorar a relação custo e benefício. Outro fato importante, é que do ponto de vista do usuário, é imperceptível a diferença, visto que a camada de abstração será sintetizada para cada visão interpretada pelos distintos sistemas operacionais, os quais o aplicativo será instalado.

Figura 5 – Diagrama e arquitetura da plataforma Xamarin.



Fonte: Portal de documentação da Microsoft (2016).

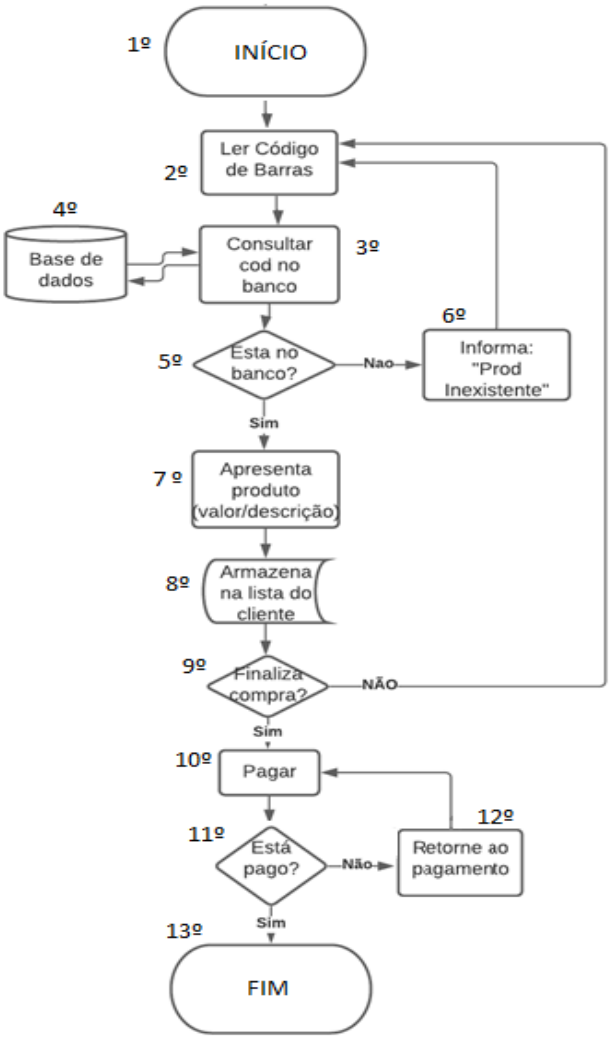
A linguagem de programação utilizada no Xamarin é o C#. Para tanto, a estrutura lógica deste aplicativo será baseada no Xamarin.Forms. Segundo a Microsoft (2021), o Xamarin.Forms é um recurso do Xamarin, a popular estrutura de desenvolvimento móvel que estende a plataforma de desenvolvedor .NET com ferramentas e bibliotecas para construir aplicativos móveis. Xamarin.Forms é uma estrutura de plataforma cruzada de código aberto da Microsoft para a construção de aplicativos IOS, Android e Windows com .NET a partir de uma única base de código compartilhada. Segundo George (2021), para que seja possível desenvolver uma programação estruturada utilizando o Xamarin.Forms e criar as interfaces gráficas ao usuário, será necessário o uso de uma linguagem de marcação, conhecida como XAML.

> O XAML é uma linguagem de marcação declarativa. Conforme aplicado ao modelo de programação do .NET Core, o XAML simplifica a criação de uma interface do usuário para um aplicativo .NET Core. Você pode criar elementos visíveis da interface do usuário na marcação XAML declarativa e, em seguida, separar a definição da interface do usuário da lógica de tempo de execução usando arquivos de código por de trás, que são adicionados à marcação por meio de definições de classe parcial. O XAML representa diretamente a instanciação de objetos em um conjunto específico de tipos de suporte definidos em *assembly*. Isso é diferente da maioria das outras linguagens de marcação, que são geralmente uma linguagem interpretada sem um vínculo direto para um sistema de tipos de suporte. (MICROSOFT, 2021).

Conforme mencionado anteriormente, outras plataformas cruzadas, como o Flutter, estão em crescente uso no mercado de aplicativos. Para este trabalho justifica-se a escolha do Xamarin, baseado na grande facilidade que esta plataforma fornece ao desenvolvedor que possui maior familiaridade com o mundo Microsoft, além de que o suporte nativo a nuvem pública Azure, facilitará a migração de estruturas como banco de dados e micro serviços diretamente com *frameworks* já existentes na plataforma.

A fim de facilitar o entendimento da metodologia que será aplicada na construção deste aplicativo e clarear a estruturação de atuação, foi elaborado um fluxograma apresentado por meio da Figura 6. Neste fluxograma, é possível visualizar de que forma a sequência de ações levará o usuário a finalizar sua compra. Apresentando todas as possíveis alternativas lógicas disponíveis perante uma escolha do usuário e baseando no conceito de testes de ponta a ponta, será possível testar todas as possiblidades, e após este método, serão aplicados testes unitários, os chamados testes pequenos, a fim de garantir o funcionamento esperado.

Figura 6 – Fluxograma do aplicativo



Fonte: Próprio autor (2021)

O Fluxograma da Figura 6 demonstra a estruturação da programação do aplicativo de maneira mais clara, é possível identificar que nos diversos blocos, a sequência de próximas ações depende da intervenção do usuário. O bloco 1º, representa o início do processo, seguido do bloco 2º, o qual representa o processo de leitura do código de barras do produto por meio da câmera do celular, que será acionada pelo aplicativo. Após a correta leitura e identificação do código de barras, no bloco 3º, ocorre a consulta e validação deste código de barras lido no banco de dados da solução, representado pelo bloco 4º. Este passo descrito representa a atuação conjunta entre os blocos 3º e 4º.

Seguindo para o bloco 5º, neste ocorre a validação dos dados lidos no banco de dados, para a correta visualização do usuário, caso o produto exista na base de dados, as informações do mesmo serão apresentadas, conforme o bloco 7º, que comprova a descrição da ação. Por outro lado, caso o código lido seja consultado na base de dados e não exista, será apresentado ao usuário a informação de que o produto é inexistente, conforme denota o bloco 6º.

Seguindo para o bloco 8º, neste o usuário pode inserir o produto lido corretamente ao carrinho virtual, armazenando esta informação em uma lista dinâmica. Ao passo, que no bloco 9º, o usuário poderá decidir se retorna para a leitura de mais códigos de barras, regressando ao bloco 2º ou escolha finalizar sua compra e seguir para o bloco 10º, o qual representa a área do aplicativo destinada ao pagamento da compra finalizada no passo anterior. Após o pagamento, o aplicativo irá realizar a verificação deste processo no bloco 11º, caso a validação seja positiva, o usuário chegará ao fim do fluxo de atuações, passo representado pelo bloco 13º. Caso a verificação do pagamento seja negativa, o usuário será direcionado a tela de pagamento novamente, regressando ao bloco 10º.

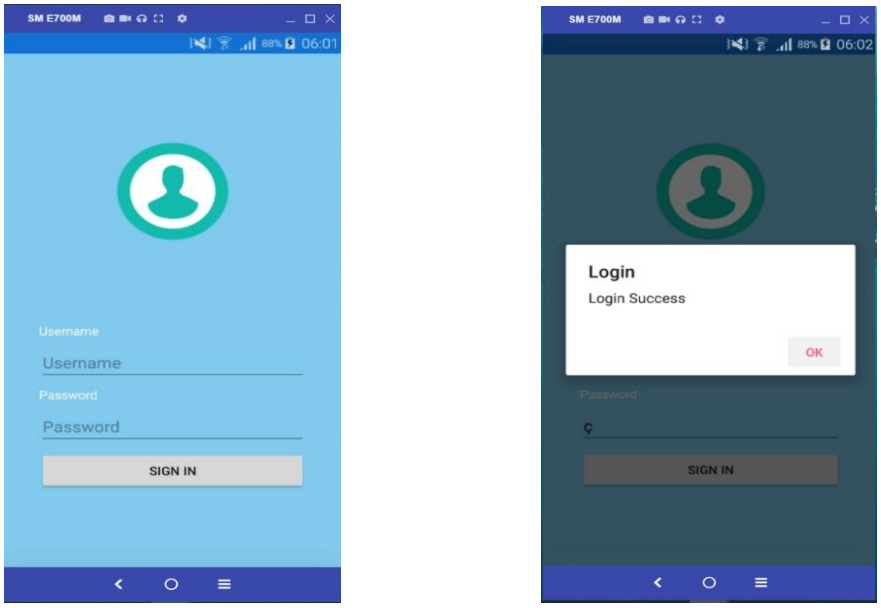
4 RESULTADOS E DISCUSSÃO

Baseado na metodologia apresentada, o desenvolvimento deste aplicativo seguiu as premissas citadas a fim de cobrir os objetivos e por fim alcançar passos importantes. As fases de desenvolvimento foram divididas em 4 etapas: *layout* primário com tela de login, implantação do método de leitura de código de barras por meio da câmera, conexão a uma base de dados e integração com API de pagamento.

4.1 PRIMEIRA ETAPA

A primeira etapa compreendeu o primeiro contato com a plataforma Xamarin, por meio do Visual Studio, possibilitando a conexão e *deployment* do código primário e uso do método de recarga dinâmica do *App*, a fim de testar a programação em um dispositivo real. Desta maneira, seguindo esta pequena arquitetura foi possível obter os primeiros resultados de testes com um código mais simples, apenas para simular emulação do aplicativo e compreender as conexões com bibliotecas relacionadas a versão do sistema operacional do dispositivo de testes, que neste caso era um Android Lollipop 5.1 implantado em um Samsung Galaxy E7. Este código primário foi formado pelo *layout* inicial do aplicativo, o qual era constituído por uma tela de login, conforme mostra a Figura 7, com o propósito de que o usuário só pudesse acessar os recursos do *App* se estivesse cadastrado na base de dados que seria conectada mais a diante.

Figura 7 – Tela de Login



Fonte: Próprio autor (2021)

4.2 SEGUNDA ETAPA

A segunda etapa do desenvolvimento deste aplicativo compreendia a implantação do código, o qual possibilitou a leitura do código de barras em diversos padrões, por meio da câmera do celular. Este código foi baseado na biblioteca Zxing, um framework desenvolvido pela empresa Zebra Technologies e massivamente usado em projetos que visam a leitura de código de barras nos padrões 1D e 2D, suportando os seguintes formatos, conforme Tabela 2.

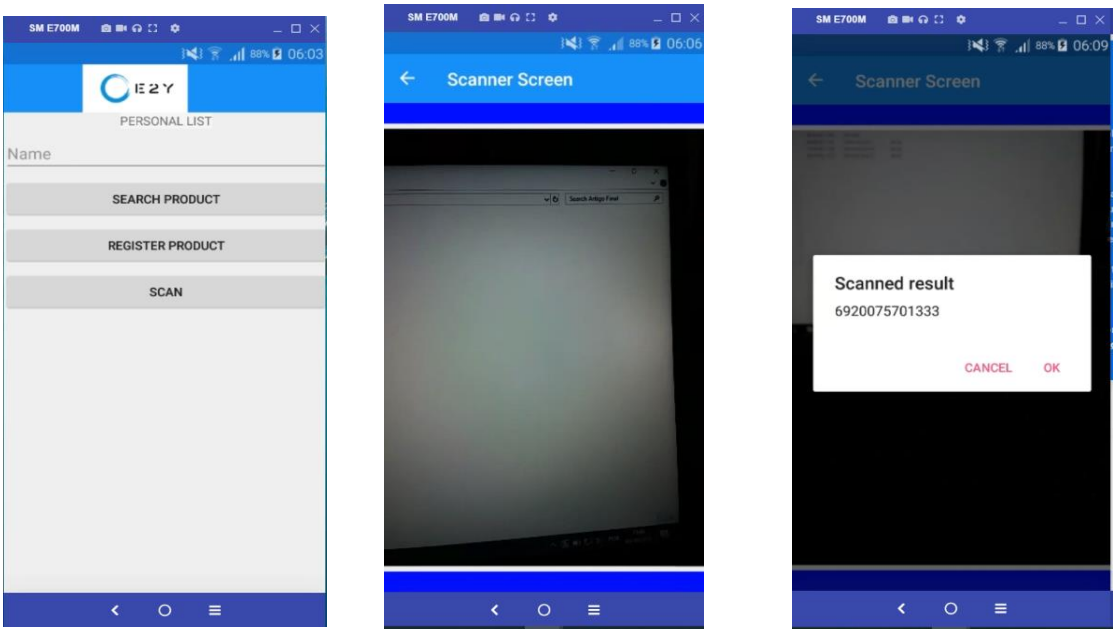
Tabela 2 – Formatos de código de barras suportados pela biblioteca Zxing

FORMATOS DE CÓDIGO DE BARRAS SUPORTADOS		
1D	1D Industrial	2D
UPC-A	Code 39	QR Code
UPC-E	Code 93	Data Matrix
EAN-8	Code 128	Aztec
EAN-13	Codabar	PDF 417
UPC/EAN Extension 2/5	ITF	Maxi Code
		RSS-14/Expanded

Fonte: Zebra Technologies (2021)

A estrutura da página de *back-end* que suporta a integração com o framework Zxing foi desenvolvida sob a ótica de seguir o layout padrão que já vinha sendo aplicado e através de um botão, chamado “SCAN”, como mostra a Figura 8 abaixo, acionado ainda na página principal, promover a mudança de página, para a então página de escaneamento e ativar a câmera que aguardará a captação de um produto real com o código de barras assim que alinhado a sua lente. Ainda nesta página de escaneamento, foi implantada a conexão com o banco dados, que atua com duas funções principais: inicialmente registrar os produtos e suas respectivas informações em uma tabela e em um segundo momento, diante a chamada de consulta do *scanner* após leitura, realizar a validação na base de dados e apresentar que o produto existe ou não, nesta base em questão, possibilitando que o usuário selecione este produto, quando existente, e adicione-o a uma lista dinâmica. Quando o produto é identificado dentro do banco de dados, um *pop-up* aparece na tela de escaneamento, apresentando o código lido e o usuário poderá seguir para a lista clicando no botão “OK”.

Figura 8 – Tela Principal e Tela de Escaneamento



Fonte: Próprio autor (2021)

4.3 TERCEIRA ETAPA

Conforme já brevemente mencionado nas etapas anteriores, para o desenvolvimento deste aplicativo foi usado um banco de dados SQLite, através da biblioteca SQLite integrada ao Xamarin Forms. O SQLite é uma base de dados relacional de código aberto e que dispensa o uso de um servidor para sua alocação. Por meio da arquitetura do banco, é possível armazenar os arquivos dentro da própria estrutura, tornado o seu uso fácil, dinâmico, fluído e leve. O

SQLite é capaz de atuar em diversas aplicações e principalmente em sistemas móveis, além de que suporta diversas linguagens de programação.

O banco de dados foi dividido em duas tabelas principais, são elas: *User* e *T_Product*. Estas tabelas compõem as principais operações relacionais das estruturas de *back-end* deste aplicativo. A tabela *User* compreende o modelo que irá balizar como o usuário acessará ao aplicativo, aplicando três parâmetros: O ID, que representa qual será o identificador do usuário dentro no banco de dados, o parâmetro *Username*, que será o nome do usuário e o parâmetro *Password*, o qual será a senha de acesso ao aplicativo. Desta forma, com estes parâmetros o usuário poderá acessar os recursos do *App* e correlacionar sua lista dinâmica ao seu respectivo ID, além é claro que as informações ficarão salvas na base de dados e sempre serão atreladas ao usuário, a menos que ele as modifique.

A tabela *T_Product* é composta pelas informações relacionadas a cada produto cadastrado no banco de dados. E esta tabela baseia-se nos seguintes parâmetros relacionais:

a) ID – Parâmetro que atua como identificador do produto dentro da base de dados;
b) Name – Parâmetro relacionado ao nome do produto;
c) Barcode – Parâmetro que refere-se ao código de barras do produto;
d) Cost – Parâmetro relacionado ao valor do produto.

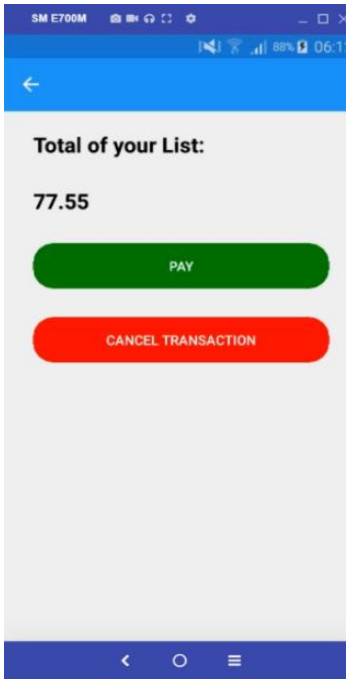
Através destes parâmetros, cada produto cadastrado deverá seguir estas especificações e nos campos de leitura, seja pelo escaneamento via câmera ou pesquisa padrão, o usuário poderá encontrar as informações dos produtos e adicioná-los a sua lista dinâmica.

4.4 QUARTA ETAPA

A quarta etapa do desenvolvimento deste aplicativo compreende o processo de integração de um API de *gateway* de pagamento conectado a página de pagamento do *App*. Foi utilizado o API da empresa Stripe, uma das líderes no mercado de desenvolvimento de sistemas para pagamentos e controle contra fraudes em transações comerciais, e também uma das maiores *startups* de pagamento em linha e softwares comerciais transacionais do mundo. Por meio do API atrelado a uma chave privada de propriedade do desenvolvedor do aplicativo, foi possível criar um botão chamado “PAY”, conforme apresentado na Figura 9, o qual faz uma chamada de conexão com a nuvem privada da Stripe e relaciona as informações do valor total da compra do usuário, o nome do mesmo e as informações do cartão de crédito. Estas informações coletadas no momento da transação garantem grande confiabilidade entre as duas partes, o cliente que utiliza seu cartão de crédito e o sistema do aplicativo que estabelece uma conexão criptografada ponto a ponto com a nuvem Stripe. Um outro ponto importante, é que

cada transação possui um ID e este é totalmente relacionado a cada transação feita pelo usuário, baseado em seus dados, além de que é possível realizar um controle granular das transações garantindo visibilidade do uso do aplicativo.

Figura 9 – Tela de Pagamento

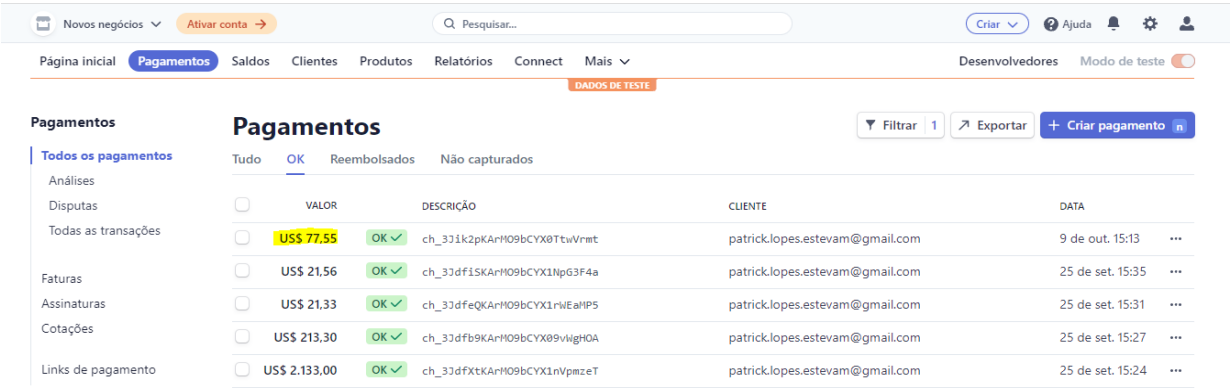


Fonte: Próprio autor (2021)

Após a correta conexão e finalização da compra, o usuário recebe um *pop-up* alertando que o pagamento foi concluído com sucesso. O usuário também pode clicar no botão “Cancel Transaction”, para enviar uma solicitação de cancelamento de pagamento caso identifique algum erro ou de fato, deseje por vontade própria cancelar a compra.

Desta forma, ao acessar o *dashboard* de administrador do API no portal Stripe, conforme Figura 10, é possível identificar a compra feita pelo cliente, com o respectivo valor apresentado no aplicativo ao finalizar a compra, as informações de cadastro, como e-mail, data e ID identificador da transação.

Figura 10 – *Dashboard* Stripe



Fonte: Próprio autor (2021)

Para tanto, todo a estrutura de programação desenvolvida, bem como as integrações com os APIs citados e o código fonte, podem ser consultados através do acesso ao repositório no Github. (LOPES, 2021).

5 CONSIDERAÇÕES FINAIS

Baseado nos resultados obtidos pode-se ressaltar que os objetivos anteriormente pretendidos foram alcançados. O aplicativo foi desenvolvido sob a ótica de oferecer simplicidade no uso e principalmente, prover ao usuário uma experiência de telas e navegação padrão, bem próxima a visão que o mercado de aplicativos aplica e que os usuários já estão acostumados. Todavia, quando foca é em eficiência e a respeito da resolução do problema de pesquisa, é possível elencar que o escaneamento de código de barras agregou extrema facilidade ao processo de compra, alcançando otimização do tempo do cliente e garantindo um sistema que atue de maneira autônoma.

Em relação às telas, há espaço para melhorias futuras baseadas nos conceitos de UX (User Experience), um tema que cresce diariamente no mercado e atualmente é presente em diversos aplicativos. O uso destas ferramentas auxilia o usuário durante o uso do aplicativo e o cativa a continuar usando os serviços disponíveis, o que muitas vezes, se transforma em atrativo para investimentos e melhorias. Visto que grande parte dos aplicativos disponibilizados na Apple Store e Google Play, são baseados na oferta de serviços, o que garante uma utilização média mensal, gere lucros e seja possível estimar a receita recorrente.

Um desafio encontrado ao longo do desenvolvimento deste projeto foi o contato com a interface de programação Xamarin, uma plataforma relativamente nova e que une dois conceitos antes abordados separadamente, o universo de programação para Android e Apple.

Um outro ponto que deve ser ressaltado é a questão do uso do banco de dados, neste modelo adotado, o aplicativo é executado junto a base de dados unido a última informação recebida do servidor. Um melhoria que traria alta disponibilidade e principalmente, otimização do espaço de armazenamento, seria atuar com a alocação deste banco de dados em uma instância de nuvem pública, replicada sob o conceito de geolocalização, a fim de garantir balanceamento de acessos e de carga, entre os usuários e repositórios de *downloads*.

REFERÊNCIAS

BALIVO, Jefferson. **Desenvolvendo aplicativos Cross-Platform com Xamarin.** 2016. Disponível em: https://www.balivo.com.br/desenvolvendo-aplicativos-cross-platform-com-xamarin/. Acesso em: 13 maio 2021.

CHAN, Stephanie. **Global Consumer Spending in Mobile Apps Reached a Record $111 Billion in 2020, Up 30% from 2019**. 2021. Disponível em: https://sensortower.com/blog/app-revenue-and-downloads-2020#:~:text=Worldwide%20Mobile%20App%20Revenue%20and,in%202019%20to%20%2438.6%20billion.. Acesso em: 21 maio 2021.

CURRY, Davd. **App Revenue Data (2021).** 2021. Disponível em: https://www.businessofapps.com/data/app-revenues/. Acesso em: 30 abr. 2021.

GEORGE, Andy de. **Visão geral do XAML (WPF .NET).** 2021. Disponível em: https://docs.microsoft.com/pt-br/dotnet/desktop/wpf/xaml/?view=netdesktop-5.0. Acesso em: 13 maio 2021.

GOBACKLOG. **VAREJO SUPERMERCADISTA: perspectiva sobre o crescimento no brasil. PERSPECTIVA SOBRE O CRESCIMENTO NO BRASIL**. 2018. Disponível em: https://gobacklog.com/varejo-supermercadista/. Acesso em: 24 abr. 2021.

GOOGLE. **Conceitos básicos de testes**. 2020. Disponível em: https://developer.android.com/training/testing/fundamentals#testing-pyramid. Acesso em: 24 abr. 2021.

HERMES, Dan; MAZLOUMI, Nima. **Building Xamarin.Forms Mobile Apps Using XAML: mobile cross-platform xaml and xamarin.forms fundamentals.** New York: Apress, 2019. 445 p.

HININGS, B., GEGENHUBER, T., GREENWOOD, R., A., MATT (2018**). "Digital innovation and transformation: An institutional perspective".** Information and Organization 28(1), pp. 52-61.

IDC. **Smartphone Shipments Return to Positive Growth in the Fourth Quarter Driven by Record Performance by Apple, According to IDC**. 2021. Disponível em: https://www.idc.com/getdoc.jsp?containerId=prUS47410621. Acesso em: 21 abr. 2021.

LEVY, Michael; WEITZ, Barton. **RETAILING MANAGEMENT**. 8. ed. Boston: McGraw-Hill, 2011. 704 p.

LOPES, Patrick. **Easy2Pay**. 2021. Disponível em: https://github.com/Patrick-EsLo/Easy2Pay. Acesso em: 01 dez. 2021.

MADUREIRA, Daniel. **Aplicativo nativo, web App ou aplicativo híbrido?** .Net.8 mar. 2017. Disponível em: https://usemobile.com.br/aplicativo-nativo-web-hibrido/ Acesso em: 21 abr. 2021.

MEYER, Maximiliano. **A história do Android**. 2015. Disponível em: https://www.oficinadanet.com.br/post/13939-a-historia-do-android. Acesso em: 21 abr. 2021.

MICROSOFT. **Xamarin.Forms: uma estrutura de código aberto para a construção de aplicativos ios, android e windows. Uma estrutura de código aberto para a construção de aplicativos iOS, Android e Windows.** 2021. Disponível em: https://dotnet.microsoft.com/apps/xamarin/xamarin-forms. Acesso em: 13 maio 2021.

ORTIS, Fabiano Soares. **FUTURO DO VAREJO NO BRASIL: aplicações móveis para facilitação de venda em lojas**. 2019. 18 f. TCC (Graduação) - Curso de Tecnólogo em Gestão da Tecnologia da Informação, Universidade do Sul de Santa Catarina, Palhoça, 2019.

SAMPAIO, R. (2018). **"Vantagem digital: Um guia prático para a transformação digital"**. Rio de Janeiro: Alta Books 160p.

STATISTA. **Worldwide mobile app revenues in 2014 to 2023.** 2021. Disponível em: https://www.statista.com/statistics/269025/worldwide-mobile-app-revenue-forecast/. Acesso em: 02 maio 2021

WU, Meng; TEUNTER, Ruud H.; ZHU, Stuart X.. Online marketing: when to offer a refund for advanced sales. **International Journal Of Research In Marketing**, Chengdu, v. 36, n. 3, p. 471-491, set. 2019. Elsevier BV. http://dx.doi.org/10.1016/j.ijresmar.2018.11.003. Disponível em: https://reader.elsevier.com/reader/sd/pii/S0167811618300661?token=546C899A05F56660FA4328863629D91C14498ED8DA76DAD6814332E2D7A986E60158F3DA0887761DC919495708832EBE&originRegion=us-east-1&originCreation=20210525104901. Acesso em: 21 maio 2021.

AGRADECIMENTOS

Gostaria de agradecer primeiramente a Deus por me conduzir e abençoar em todas as coisas; a minha família, em especial minha mãe Viviane Aparecida Estevam e minha avó Aparecida Devita Estevam pelo exemplo de apoio e compreensão, ao Prof. Dr. Paulo Lopes por todo apoio, tempo e atenção.

Agradeço ao Instituto Presbiteriano Mackenzie, bem como a Universidade Presbiteriana Mackenzie por possibilitar um plano de ensino de alto nível, construindo uma importante base para minha carreira profissional.