INSTITUTO SUPERIOR DE TECNOLOGIAS AVANÇADAS
LICENCIATURA EM ENGENHARIA MULTIMÉDIA

InvoiceSafe
Gestão e Armazenamento de Finanças Pessoais

Realizado por: Ruben Filipe dos Santos Alves Brito, Nº 21304
Coordenador - Professor Dr. Pedro Brandão
Orientadora - Professora Dra. Dulce Mourato

Lisboa, Ano letivo 2022/2023

INSTITUTO SUPERIOR DE TECNOLOGIAS AVANÇADAS
LICENCIATURA EM ENGENHARIA MULTIMÉDIA

InvoiceSafe
Gestão e Armazenamento de Finanças Pessoais

Realizado por: Ruben Filipe dos Santos Alves Brito, Nº 21304
Coordenador - Professor Dr. Pedro Brandão
Orientadora - Professora Dra. Dulce Mourato

Lisboa, Ano letivo 2022/2023

Dedicatória

Ao concluir esta etapa crucial na minha jornada académica, quero expressar a minha profunda gratidão à minha querida esposa, Laura Alves Brito, e ao nosso precioso filho, Rafael Alves Brito, que são a minha maior fonte de inspiração e alegria.

Minha querida esposa, és a âncora que me mantém firme e a luz que guia os meus passos. Agradeço por estares sempre ao meu lado, sustentando-me com amor e compreensão a cada desafio nesta jornada desafiadora.

Ao nosso filho, és a minha maior motivação. O teu sorriso e a tua presença enchem os meus dias de alegria e significado. És a minha inspiração para alcançar o melhor em tudo o que faço.

Este projeto final não representa apenas o meu esforço académico, mas também o apoio incondicional da minha família. Agradeço a todos os amigos, professores e colegas que compartilharam este percurso comigo.

Que este trabalho académico seja um testemunho do meu compromisso e dedicação, e que possa inspirar outros da mesma forma que a minha família me inspira diariamente.

Com gratidão e determinação

Agradecimentos

É com enorme apreço e gratidão que me dirijo a todos aqueles que estiveram a meu lado durante a realização deste projeto académico. Primeiramente, um agradecimento especial à minha família, cujo apoio incondicional foi o meu bastião em todos os momentos.

Não posso deixar de expressar a minha sincera gratidão aos meus colegas do grupo "Multimédia Gang": Rui D'Azevedo, Gonçalo Braz, Imran Rahim, Nuno Matos, Edgar Dias, Carlos Andrade, José Neto e Tiago Mateus. A colaboração, o espírito de equipa e o apoio de cada um foram pilares fundamentais nesta jornada académica.

Um reconhecimento profundo ao corpo docente do ISTEC, cuja dedicação e sabedoria enriqueceram grandemente a minha formação. Em especial, quero salientar o Professor Doutor Pedro Ramos Brandão, cuja orientação, disponibilidade e suporte metodológico foram essenciais para o êxito deste projeto. A sua paixão pelo conhecimento e dedicação ao ensino foram fontes de inspiração inestimáveis para mim.

Agradeço também aos professores Filomena Marques, José Neves, Vitor Joaquim e Dulce Mourato pelas suas contribuições preciosas, orientações e ensinamentos que foram cruciais para o enriquecimento deste trabalho.

A todos vós, o meu mais sincero agradecimento. Esta jornada não teria sido a mesma sem o vosso apoio, orientação e colaboração. Este projeto não é apenas um marco na minha carreira académica, mas também um reflexo do espírito de comunidade e da partilha de conhecimento que todos vós personificais. Espero levar as lições aprendidas e as experiências partilhadas comigo ao longo da minha jornada académica e profissional.

Com profundo respeito e gratidão, muito obrigado.

Resumo

Este projeto académico centrou-se no desenvolvimento de uma aplicação móvel (InvoiceSafe), destinada à gestão de faturas, garantias e prescrições médicas. A motivação para tal iniciativa surgiu no contexto da crescente digitalização dos serviços financeiros, aliada à necessidade de otimizar a gestão do tempo e dos recursos pessoais. O principal objetivo deste projeto foi o desenvolvimento de uma interface intuitiva, que centralizasse a administração destes documentos vitais, simplificando assim o dia-a-dia dos utilizadores.

Entre os objetivos secundários, destacaram-se a implementação de uma funcionalidade de leitura automática de dados através de códigos QR e a possibilidade de inserção manual de registos. Além disso, este estudo explorou a identificação do sistema operacional mais utilizado pelo público-alvo e avaliou a viabilidade de centralizar documentos em um único banco de dados robusto, em oposição ao armazenamento fragmentado.

A metodologia aplicada incluiu a análise das preferências dos utilizadores através de inquéritos e uma revisão abrangente da literatura, garantindo que a aplicação satisfazia as necessidades evidenciadas. O objetivo deste projeto foi facilitar a gestão de documentos financeiros e médicos, proporcionando uma solução integrada que respondesse aos desafios impostos pelo dia-a-dia acelerado.

Palavras-Chave: Android, SQLite, Gestão, Faturas, QRCode.

Abstract

This academic project focused on the development of a mobile application (InvoiceSafe), aimed at the management of invoices, warranties, and medical prescriptions. The motivation for such an initiative arose in the context of the increasing digitalization of financial services, combined with the need to optimize the management of time and personal resources. The main objective of this project was the development of an intuitive interface, which would centralize the administration of these vital documents, thus simplifying the day-to-day life of users.

Among the secondary objectives, the implementation of functionality for automatic data reading through QR codes and the possibility of manual entry of records were highlighted. In addition, this study explored the identification of the operating system most used by the target audience and evaluated the feasibility of centralizing documents in a single robust database, as opposed to fragmented storage.

The methodology applied included the analysis of user preferences through surveys and a comprehensive literature review, ensuring that the application met the needs evidenced. The objective of this project was to facilitate the management of financial and medical documents, providing an integrated solution that would respond to the challenges imposed by fast-paced day-to-day life.

Keywords: Android, SQLite, Management, Invoices, QR Code.

Abreviaturas

2D - Bidimensional

3D - Tridimensional

API - Application Programming Interface

APP - Application (Aplicação)

ATCUD - Código de Barras Bidimensional Único por Documento

BD - Base de Dados

CSS - Cascading Style Sheets

ES - Engine System

HTTP - HyperText Transfer Protocol

HTTPS - HyperText Transfer Protocol Secure

IDE - Integrated Development Environment

NoSQL - Not Only SQL

QR Code - Quick Response Code

SDK - Software Development Kit

SGL - Scalable Graphics Library.

SQL - Structured Query Language

SSL - Secure Sockets Layer

UI - User Interface

UX - User Experience

CBC - Cipher Block Chaining

Índice

1. INTRODUÇÃO 10

2. REVISÃO DE LITERATURA 11

2.1. A História do Android 12

2.2. Flexibilidade de Uso e Interfaces 17

2.3. Desenvolver Aplicações Móveis e Web 19

3. MATERIAIS E MÉTODOS DE DESENVOLVIMENTO 24

4. RESULTADOS E DISCUSSÃO DE RESULTADOS 29

4.1. Como funciona a aplicação InvoiceSafe 35

4.2. Interface Amigável e Intuitiva 49

4.3. Integração de Funcionalidades e Testes 51

4.4. Desafios e Soluções 55

5. CONCLUSÃO 59

6. BIBLIOGRAFIA 61

Anexo 1 - Evolução cronológica do Android por ordem decrescente 63

Anexo 2 - Especificações Técnicas do Código de Barras 2D - QR Code **Erro! Marcador não definido.**

Apêndice A – Questionário de investigação 72

Apêndice B – Formulário de testes para a aplicação InvoiceSafe.V01 78

Apêndice C - Acesso ao artefacto digital: 80

Índice de Tabelas

Tabela 1 - Recolha de dados sobre a APP Android de literacia financeira 25
Tabela 2 - Versões da plataforma Android 63
Tabela 3 - Especificações técnicas do código 2D (QR) 70

Índice de Figuras

Figura 1: Estrutura das camadas Android 1 13
Figura 2 - Princípio contraste 18
Figura 3 - Princípio do Alinhamento 18
Figura 4 - Princípio da repetição 19
Figura 5 - Princípio de proximidade 19
Figura 6 - Os pilares do Firebase 20
Figura 7 - Código ATCUD 22
Figura 8: Cronograma da implementação da App Android InvoiceSafe 27
Figura 9: Wireframe review InvoiceSafe 28
Figura 10: Faixa Etária dos respondentes 29
Figura 11: Género dos Participantes 30
Figura 12: Nível de escolaridade 30
Figura 13: Ocupação atual 30
Figura 14: Frequência de Utilização de Dispositivos Móveis 31
Figura 15: Sistema Operativo Utilizado. 31
Figura 16: Uso de Aplicações de Gestão Financeira 31
Figura 17: Funcionalidades Essenciais em Aplicações de Gestão 32
Figura 18: Disposição para Pagar por Aplicações 32
Figura 19: Preocupações ao Utilizar Aplicações de Gestão de Documentos 32
Figura 20: Frequência de Sincronização de Dados em Aplicações 33
Figura 21: Volume de Dados Usados Mensalmente, em dispositivos móveis 33
Figura 23: Problemas de Conectividade ou Velocidade, em dispositivos móveis. 34
Figura 24: Preferências de sincronização automática versus manual em aplicações 34
Figura 25: Respostas livre sobre funcionalidades 35
Figura 26: Método onUpgrade 37
Figura 27: Tabela invoice. 37
Figura 28: Função SQL onOpen 38
Figura 29: Tabela prescription 38
Figura 30: Tabela warranty 38
Figura 31: Tabela issuer 39
Figura 32: Ecrã de login 40

Figura 33: Ecrã Registar o perfil 40
Figura 34: Wireframe review InvoiceSafe 44
Figura 35: Ecrãs adicionar/Editar: Fatura/Garantia/Prescrição médica 46
Figura 36: Ecrã Captura de QR Code 47
Figura 37: Fragment Lista de faturas 49
Figura 38 - Logótipo 49
Figura 39: Package utils 54
Figura 40: Ecrã Login 55
Figura 41: Ecrã adicionar fatura.....56
Figura 42: IInvoiceDAO 57
Figura 43 - Invoice Contract 57
Figura 44: Model Invoice.....57
Figura 45: InvoiceDAO.....57

1. Introdução

O presente trabalho académico centrou-se no desenvolvimento de uma aplicação móvel denominada InvoiceSafe, destinada à gestão e armazenamento eficiente de faturas, recibos e prescrições médicas. Esta escolha temática advém da necessidade identificada de simplificar a gestão das finanças pessoais através da digitalização e organização automática destes documentos essenciais no quotidiano dos utilizadores. A relevância do tema justifica-se tanto pela crescente digitalização dos serviços financeiros quanto pela otimização do tempo e recursos na administração pessoal.

O objetivo principal deste projeto é criar uma interface intuitiva que permita aos utilizadores uma gestão descomplicada e centralizada dos seus documentos financeiros e médicos. Entre os objetivos secundários, destaca-se a funcionalidade de leitura automática de dados através do código QR das faturas, a inserção manual de documentos, a associação de imagens e dados adicionais a cada registo, e a capacidade de efetuar pesquisas avançadas por meio de filtros SQL numa base de dados robusta.

Este estudo pesquisou diversas questões fundamentais para o desenvolvimento do artefacto digital, incluindo a determinação do sistema operativo mais prevalente entre os potenciais utilizadores, a identificação do público-alvo específico para a aplicação, e a viabilidade de centralizar documentos financeiros e médicos numa base de dados em detrimento do armazenamento convencional em galerias de fotos nos dispositivos móveis. Adicionalmente, será explorada a estrutura de dados mais adequada para suportar as funcionalidades propostas.

A metodologia adotada para este projeto incluiu a seleção da plataforma de desenvolvimento com base na popularidade e crescimento dos sistemas operativos móveis, a realização de um inquérito para recolher preferências dos utilizadores e uma revisão bibliográfica extensiva usando recursos digitais e impressos especializados.

Com este estudo, foi possível desenvolver um sistema de informação que visou não apenas simplificar a gestão diária de documentos financeiros e médicos dos utilizadores, mas também proporcionar uma experiência de utilização diferente. Este projeto pretendeu, portanto, estimular uma reflexão sobre as melhores práticas de gestão digital de documentos, instigando a curiosidade do leitor acerca das soluções tecnológicas que podem transformar a organização pessoal no contexto da modernidade digital.

2. Revisão de Literatura

Nos últimos anos, a integração da tecnologia móvel no quotidiano, revolucionou a forma como gerimos as nossas vidas, particularmente no que respeita ao controlo financeiro, cuidados de saúde e praticidade nas transações comerciais. A emergência de sistemas operativos como o Android, baseado no Kernel Linux e de código-fonte aberto, cujas licenças pertencem à Google Inc., foi fundamental nesta transformação. Este trabalho centrou-se no desenvolvimento de uma aplicação móvel Android, que visa a gestão integrada de diversos aspetos fundamentais do dia-a-dia.

O Android distingue-se não só pela sua disponibilidade gratuita para fabricantes, que podem adaptar o sistema sem alterar o seu núcleo, mas também pelo seu design altamente interativo. Concebido inicialmente para dispositivos com ecrãs tácteis, como smartphones e tablets, o Android facilita a manipulação direta de conteúdos gráficos através do toque. Ações como zoom com dois dedos, toques duplos para interação, e arrastar e soltar objetos no ecrã tornam a interface extremamente intuitiva. Estas características são cruciais para o sucesso de aplicações destinadas a simplificar a gestão quotidiana, permitindo uma experiência de utilizador mais fluída e adaptável. Além disso, a expansão do Android para diferentes plataformas como Wearables, TVs e dispositivos automotivos, amplia ainda mais o alcance e a relevância do sistema operativo na implementação de soluções inovadoras para a gestão de atividades diárias.

A Android, Inc., fundada em outubro de 2003 em Palo Alto, Califórnia, por Andy Rubin**,** Rich Miner**,** Nick Sears **e** Chris White, teve como objetivo a criação de um sistema operativo para dispositivos móveis inteligentes que respondesse melhor à localização e às preferências dos seus utilizadores. Esta visão inicial, segundo Rubin (2003), era desenvolver "dispositivos móveis mais inteligentes que fossem mais conscientes da localização e das preferências de seus donos", sublinhou Callaham (2023, parágrafo 5). Com a aquisição da Android Inc. pelo Google, Rubin liderou o desenvolvimento de uma plataforma baseada no kernel de Linux, que ofereceu um sistema de evolução flexível, revelado apenas ao público em 2007 com a fundação do consórcio Open Handset Alliance. Este consórcio, composto por entidades de hardware, software e telecomunicações, foi estabelecido para criar padrões abertos para dispositivos móveis. Desde o seu lançamento público em 2008, o sistema Android tem recebido atualizações sucessivas, introduzindo novas funcionalidades e corrigindo falhas das versões anteriores.

As evoluções do sistema operativo Android podem ser traçadas através dos seus codinomes, que curiosamente são nomes de sobremesas ou doces, seguindo uma ordem alfabética. Esta peculiaridade começou com o codinome 'Apple Pie' para o Android 1.0, seguido por 'Banana Bread' para a versão 1.1. A tradição continuou até à versão atual, que é o Android 13, conhecido pelo codinome 'Tiramisu'. A escolha desses nomes não é apenas uma curiosidade cultural, mas reflete uma estratégia de marketing para tornar cada versão mais memorável e acessível ao público. As atualizações não apenas melhoraram a funcionalidade e a segurança do sistema operativo, mas também introduziram novas ferramentas que permitem aos programadores, como no caso do projeto presente, criar aplicações mais eficazes e intuitivas. A evolução cronológica dessas versões pode ser visualizada na tabela cronológica nº1 (Anexo1).

2.1.A História do Android

Para uma compreensão aprofundada deste Sistema Operativo, foi efetuada uma exploração temporal das principais características das versões do Android, identificadas através dos seus codinomes apresentados na Tabela 1 - Versões da Plataforma Android (Anexo1). Cada uma das versões principais é nomeada com um termo doce, seguindo uma ordem alfabética. Por exemplo, o primeiro codinome para o Android 1.0 é 'Apple Pie', seguido por 'Banana Bread' para a versão 1.1, estando atualmente na versão 7 com o codinome 'Nougat', conforme documentado (android.stackexchange.com, 2015).

Segundo os dados disponibilizados pela documentação técnica do Android Studio em meados de janeiro de 2023, embora se destaque no gráfico do Google o Android 13, não existe distinção efetiva entre o Android 12 e o 12L. Observando as versões mais antigas, constata-se que a utilização do Android Oreo finalmente decresceu para abaixo dos 10 por cento, com reduções percentuais semelhantes para outras versões. O Android Jelly Bean, que anteriormente registava 0,3 por cento de uso, já não figura na lista, enquanto o KitKat passou de 0,9 por cento para 0,7 por cento. Indubitavelmente, a rápida adoção do Android 13 deve-se, em parte, à eficácia com que os fabricantes de dispositivos móveis, incluindo Google, Samsung, OnePlus, Sony, entre outros, têm implementado as atualizações, como Bradshaw (Android Open Source Project, 2023, para. 1-3) concluiu.

Na Figura 1 (Android Open Source Project, 2024, para.1) é possível identificar a estrutura das camadas Android, que vão identificadas de seguida.

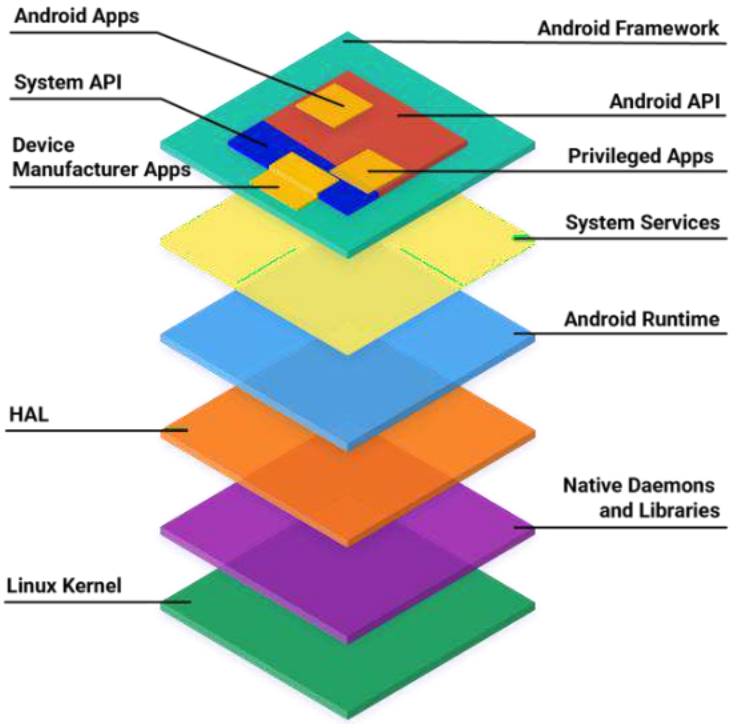


Figura 1: Estrutura das camadas Android 1 (Android Open Source Project, 2023, para.1).

A **camada kernel** Linux, anteriormente mencionada, adota o sistema operativo Linux dada a sua comprovada eficácia em sistemas de ambiente de trabalho, onde, frequentemente, não necessita de alterações nos *drivers*. Adicionalmente, o sistema operativo Linux proporciona um sistema de memória virtual, rede, drivers e controlo de consumo energético robusto, conforme descrito em developer.android.com (Android Open Source Project, 2023b, para. 1-3).

Na **camada de bibliotecas nativas**, o Android dispõe das suas funcionalidades essenciais. É nesta camada que se encontram integrados o SGL, que atua como o principal motor gráfico 2D, e o ES, que proporciona suporte gráfico 3D. Inclui também o sistema de base de dados SQLite, responsável pelo armazenamento de dados. O motor de renderização web WebKit, adaptado para visualização de páginas Web em ecrãs de menores dimensões, faz igualmente parte desta camada, conforme descrito em developer.android.com (Android Open Source Project, 2023, para. 1-3).

Destaca-se nesta camada a máquina virtual Dalvik, um intérprete otimizado para bytecodes e particularmente adaptado a plataformas móveis. Os bytecodes são convertidos em binários JAVA, facilitando o processamento pelos microprocessadores dos dispositivos móveis.

Esta camada alberga bibliotecas nativas escritas em JAVA, proporcionando um vasto conjunto de classes de programação core, típicas de uma máquina virtual JAVA, conforme documentado em developer.android.com (Android Open Source Project, 2023b).

A **Camada Application Framework** é integralmente desenvolvida na linguagem de programação JAVA e constitui a principal camada de suporte para as aplicações. É responsável pela gestão do acesso partilhado aos recursos de armazenamento e ao sistema de telefonia, além de administrar as notificações. Esta camada é crucial em qualquer sistema operativo Android, independentemente do seu programador ou da versão em utilização, pois todas as aplicações dependem desta camada para interagir com as camadas inferiores, conforme está documentado em developer.android.com (Android Open Source Project, 2023b).

Na **Camada das Aplicações** encontram-se todas as aplicações desenvolvidas para dispositivos móveis, desde o software que gere os contactos do telemóvel até aos programas criados por outros programadores. Assim, independentemente do programador, é possível criar aplicações que enriquecem a experiência do utilizador com o sistema Android. Este modelo de gestão livre das aplicações que controlam as funções do dispositivo móvel contribui para que o sistema seja percebido como mais robusto e facilite a interação com o utilizador, conforme descrito em developer.android.com (Android Open Source Project, 2023b).

O Android foi concebido para ser um sistema operativo tanto livre quanto padrão, por isso se denomina liberdade padronizada, não se restringindo a um tipo específico de dispositivos móveis. Este sistema assenta em quatro ideias fundamentais:

- **Liberdade:** O Android é extremamente favorável à programação, oferecendo total liberdade no uso de ferramentas para o desenvolvimento de aplicações, tal como descrito em developer.android.com (Android Open Source Project, 2023b, para. 12-13).
- **Igualdade nas Aplicações:** No Android, as aplicações do sistema e aquelas desenvolvidas por terceiros têm igualdade no acesso e gestão de recursos, bem como na interação com as camadas mais profundas do sistema. Uma aplicação externa pode até substituir as aplicações centrais, conforme indicado em developer.android.com (Android Open Source Project, 2023b, para.5).
- **Sem Limites para Aplicações:** Neste sistema, nenhuma funcionalidade é ocultada aos programadores. É possível criar aplicações que acedam diretamente a funcionalidades como o sistema de telefonia ou o GPS, como mencionado em developer.android.com (Android Open Source Project, 2023b, para.10).

- **Rapidez e Facilidade de Comunicação:** A camada 'application framework' facilita a comunicação entre as camadas superiores e as camadas core da estrutura. Esta característica agiliza o desenvolvimento no Android, permitindo aos programadores concentrarem-se mais no design gráfico do que nos detalhes técnicos de implementação, conforme descrito em developer.android.com (Android Open Source Project, 2023b, para. 5-10).

O **IDE Android Studio** Conforme indicado por Gusmão (2013, para.1), é uma plataforma de código aberto e foi estabelecido como o ambiente de desenvolvimento integrado (IDE) oficial para a criação de aplicações para Android. Lançado no ano de 2013, este IDE foi desenvolvido com base no ambiente integrado de JAVA 'IntelliJ IDEA', tendo substituído o Eclipse, que era anteriormente utilizado no desenvolvimento de aplicações para Android.

O Android Studio vem equipado com uma diversidade de ferramentas destinadas a auxiliar os programadores de Android, entre as quais se destacam:

- **Ferramentas SDK para Android:** Conjunto de ferramentas de desenvolvimento de software específicas para Android.
- **Plataformas Android:** Versões do sistema operativo Android disponíveis para desenvolvimento e teste.
- **Simulador de dispositivos móveis:** Permite a simulação de aplicações em diversos tamanhos de ecrã e versões do sistema operativo Android.

A versão mais recente do Android Studio, denominada 'Android Studio - Giraffe | 2022.3.1 Canary 9', foi lançada em Março de 2023. Esta atualização visa corrigir defeitos anteriores e introduzir as novas versões do sistema Android, conforme descrito em developer.android.com (Android Open Source Project, 2023b, para. 1-5).

O desenvolvimento com o Android SDK inclui um conjunto extensivo de interfaces de programação aplicacionais, conhecidas como API's. "O núcleo do sistema Android é aberto e acessível, permitindo que diversas aplicações garantam as permissões apropriadas e partilhem de forma segura o acesso aos dados do sistema", como referido por Holla & Mahima (2012, p. 488). "Esta plataforma de desenvolvimento permite aos programadores construir aplicações inovadoras que aproveitam ao máximo as capacidades do hardware do dispositivo. Com o uso das API's, torna-se viável a reutilização de componentes já existentes", conforme constatado por Holla & Mahima (2012, pp. 488-490).

O desenvolvimento Android pode ser estruturado através de camadas, que incluem: camada HTTP, API, camada genérica de dados, plataforma dependente da camada de dados e a UI (Interface do Utilizador). Na camada HTTP, é gerida a comunicação entre o dispositivo e

o servidor. As informações recebidas são então transmitidas em formato de texto (*string*) para a camada API, onde as informações necessárias são extraídas e encaminhadas para a camada genérica de dados. Esta camada contém as definições estruturais conforme as necessidades do negócio, incluindo funcionalidades como armazenamento temporário, criação de ficheiros de registo e validações. Posteriormente, na plataforma dependente da camada de dados, os dados são transformados em classes, listas e outros métodos de armazenamento definidos pelo programador. Finalmente, na camada UI, "os dados são apresentados de forma que o utilizador possa interagir com o dispositivo, sendo esta camada responsável pela parte gráfica exibida no visor de dispositivos com sistema Android", como explicado por Holla & Mahima (2012, pp. 488-490).

O modelo aplicacional do Android é composto por um conjunto de componentes, que são instanciados e executados conforme necessário. Estes componentes dividem-se nas seguintes categorias:

- **Atividade:** Componentes que constituem a base da interação com o utilizador; cada janela da aplicação é gerida por uma atividade.
- **Serviço:** Componentes que operam de forma oculta ao utilizador, mantendo-se ativos mesmo quando se navega para outra janela. Os serviços podem oferecer métodos de comunicação com outras aplicações.
- **Receção:** Componentes responsáveis pela receção assíncrona de mensagens de outras aplicações.
- **Provedores:** Componentes que armazenam dados relevantes, geralmente utilizados em bases de dados e frequentemente partilhados por várias aplicações.

Holla & Mahima (2012, pp. 488-490) explicam que "no SDK Android, existem classes base para cada tipo de componente, que são invocadas através de métodos em diferentes fases do ciclo de vida destas classes".

Gewirtz (2014, para.4) salientou que, em comparação com a usabilidade dos ecrãs entre Android e iOS, o sistema iOS é bastante limitado. Nota-se que o iOS não dispõe de 'widgets', e os ícones apresentados são simplificados e não personalizáveis pelo utilizador, resultando numa grelha estática de ícones de aplicações. Em contraste, o sistema Android foi desenhado para oferecer total liberdade de personalização do ecrã, permitindo a alteração de ícones e a seleção de 'widgets' que otimizam a interação com o dispositivo. Esta abordagem torna o ecrã do sistema Android dinâmico, possibilitando uma personalização constante por parte do utilizador.

2.2. Flexibilidade de Uso e Interfaces

Princípios de Usabilidade para Interfaces de Utilizador: A navegabilidade da interface é crucial no desenvolvimento da interação com o utilizador, facilitando o acesso para aqueles que encontram dificuldades com certos tipos de interfaces gráficas, como explicado pelo U. S. Department of Health and Human Services (DHHS, 2009, pp. 30-187).

Com base nas práticas recomendadas para o desenvolvimento de interfaces, propõe-se o seguinte conjunto de recomendações (DHHS, 2009, pp. 30-187):

- **Padronizar a Sequência de Tarefas:** Funções semelhantes devem ser executadas seguindo sempre a mesma sequência de atividades para manter a consistência.
- **Criar Hiperligações Descritivas:** O texto associado às hiperligações deve ser o mais explícito possível para evitar ambiguidades.
- **Cabeçalhos Distintos para Funcionalidades Distintas:** Cada conteúdo apresentado deve ter o seu próprio cabeçalho, clarificando a estrutura da informação.
- **Utilizar Caixas de Seleção para Escolhas Binárias:** Em respostas do tipo sim/não, o uso de caixas de seleção clarifica as opções disponíveis ao utilizador.
- **Permitir Impressão do Ecrã:** As interfaces devem ser desenvolvidas de modo que permitam a impressão de tudo o que é exibido no ecrã do utilizador.
- **Usar Imagens Pequenas para Pré-visualização:** Imagens no tamanho original devem ser substituídas por miniaturas para não sobrecarregar a interface.

Estas diretrizes ajudam a evitar interpretações erradas por parte dos utilizadores, organizando as informações de forma mais eficaz e melhorando a visualização dos conteúdos, conforme indicado por Shneiderman & Plaisant (2009). Outras recomendações incluem:

- **Consistência na Exibição dos Dados:** Terminologias, abreviações e capitalizações devem ser uniformizadas e documentadas em forma de dicionário para uso consistente pelos responsáveis pela parte gráfica.
- **Facilitar a Assimilação de Informação pelo Utilizador:** A linguagem utilizada deve ser clara e os dados devem ser apresentados de maneira a facilitar a sua compreensão. Aspetos como espaçamento e alinhamento dos textos são cruciais para uma boa visualização.
- **Minimizar a Carga de Memória do Utilizador:** Informações devem ser apresentadas de modo que cada atividade possa ser completada com o mínimo de passos necessários, dividindo grandes quantidades de informação em partes gerenciáveis.

- **Compatibilidade de Entrada de Dados e Conteúdo Exibido:** Informações exibidas devem corresponder aos elementos de inserção de dados, integrando as descrições necessárias nos próprios campos de entrada.
- **Flexibilidade de Uso e Acesso às Informações**: Deve-se permitir que os utilizadores adaptem a apresentação de tabelas e outras informações conforme sua preferência, aumentando a usabilidade da interface, como mencionado por Smith & Mosier (1986, para. 10-25).

Williams (2005, p. 13) explica que existe um conjunto de princípios fundamentais para o desenvolvimento de qualquer conteúdo gráfico, conhecidos pela sigla **CARP: Contraste, Alinhamento, Repetição e Proximidade.**

- **Princípio do contraste**: implica que os elementos mais importantes de uma aplicação devem destacar-se dos demais em relação ao fundo. "A visão humana tende a ser atraída por elementos que se destacam num padrão, um fator crucial na elaboração de interfaces" (Williams, 2005, p. 84).

Figura 2 - Princípio contraste (viesdigita, 2023, para. 2).

- **Princípio do alinhamento**: Os elementos visuais devem ser organizados de maneira estratégica e não de forma aleatória. "A linearidade no alinhamento dos elementos facilita a compreensão do utilizador, tornando a experiência visual mais confortável e captando de forma mais eficaz a atenção", de acordo com Williams (2005, p. 54) que descreve a importância deste princípio na organização visual de interfaces.

Figura 3 - Princípio do Alinhamento (viesdigita, 2023, para. 3).

- P**rincípio da repetição:** A repetição de elementos gráficos numa interface serve como uma âncora visual, facilitando a transição do utilizador de um elemento para outro.

"Esta repetição indica ao utilizador que os elementos são parte de um mesmo grupo, simplificando a interpretação visual", destaca Williams (2005, p. 68).

Figura 4 - Princípio da repetição (viesdigita, 2023, par. 4)

- O **princípio da proximidade** : Este princípio estipula que um grupo de elementos relacionados deve estar mais próximo entre si do que de outros grupos de elementos não relacionados. "Tal disposição promove a unificação e a organização visual do layout, facilitando a perceção de agrupamentos relacionados pelo utilizador", segundo Williams (2005, p. 32) que detalha a importância deste princípio na estruturação de interfaces gráficas.

Figura 5 - Princípio de proximidade (viesdigita, 2023, par. 5).

Dix et al. (2004, p. 117) enfatizam que "a aplicação dos princípios de design deve estar em consonância com o contexto de uso". É fundamental que o contraste não interfira no fluxo natural da leitura de textos e que o design gráfico considere as limitações dos utilizadores em relação aos dispositivos utilizados. Adicionalmente, a criação de interfaces deve ser um processo contínuo durante o desenvolvimento, envolvendo a produção de protótipos para a avaliação da usabilidade.

2.3.Desenvolver Aplicações Móveis e Web

O Firebase, uma plataforma desenvolvida pela Google, tem-se afirmado como uma ferramenta essencial no desenvolvimento de aplicações móveis e Web. Segundo Moroney (2017, pp. 1-8), o Firebase foi construído sobre três pilares conforme representado na Figura 6 - Os pilares do Firebase Moroney (2017, p. 2):

- Desenvolver:
- Crescer
- Ganhar

.

Figura 6 - Os pilares do Firebase Moroney (2017, p. 2)

Cada uma destas tecnologias está interligada através do Analytics. O Firebase Analytics é uma ferramenta oferecida pelo Google que possibilita a análise de dados gerados por aplicações móveis. Esta plataforma recolhe, processa e apresenta informações sobre o comportamento dos utilizadores das aplicações, permitindo assim uma compreensão aprofundada sobre a interação destes com as várias funcionalidades do aplicativo. Através da análise destes dados, os criadores de aplicações podem otimizar o desempenho da aplicação e melhorar a experiência do utilizador. O Firebase Analytics é integrado com outras ferramentas do Google, proporcionando um ambiente robusto para a gestão eficaz dos dados analíticos. (Google LLC, 2023b)

A escolha de que tecnologia será usada é definida por quem desenvolve as aplicações, é de salientar que muitas destas tecnologias estão disponíveis gratuitamente, os preços praticados são reduzidos na sua maioria, o que torna o uso desta plataforma bastante atrativa porque permite ter acesso a tecnologia de topo a baixo custo.

Segundo Moroney (2017, pp. 1-8), o Firebase inclui oito tecnologias principais projetadas para aprimorar o desenvolvimento de aplicações em diversas plataformas como Android, iOS, websites, Smart TVs, Wearables e auto. Estes serviços combinados oferecem uma série de ferramentas poderosas que permitem aos programadores criar e aperfeiçoar as suas aplicações de maneira eficiente e escalável.

- **Autenticação Firebase:** Permite que os utilizadores se registem e façam login na aplicação utilizando diversos métodos, incluindo e-mail/senha, Google, Facebook, Twitter, entre outros.

- **Cloud Firestore:** Trata-se de uma base de dados NoSQL que armazena e sincroniza dados em tempo real entre os dispositivos dos utilizadores, garantindo escalabilidade e uma experiência fluida.
- **Funções na Cloud:** Facilita a criação de funções de backend sem a necessidade de gerir infraestrutura de servidor, podendo ser ativadas por eventos de outros produtos Firebase, como alterações na base de dados ou uploads no Storage.
- **Firebase Cloud Messaging (FCM):** Oferece uma solução para envio de notificações push, mantendo os utilizadores atualizados sobre novidades e outros conteúdos relevantes.
- **Base de Dados em Tempo Real do Firebase:** Proporciona um armazenamento seguro e escalável na nuvem para ficheiros como imagens, vídeos e áudios, ou outros conteúdos gerados pelos utilizadores.
- **Armazenamento do Firebase:** Disponibiliza uma hospedagem segura e eficiente para conteúdo estático, como HTML, CSS e JavaScript, facilitando lançamentos rápidos e eficazes das aplicações web.
- **Hospedagem do Firebase:** Semelhante ao armazenamento, oferece hospedagem rápida e segura para conteúdos estáticos, permitindo implementações ágeis das aplicações web.
- **Remote Config do Firebase:** Permite personalizar o comportamento e a aparência da aplicação sem a necessidade de distribuir uma atualização oficial. Alterações na configuração podem ser feitas em tempo real.

Transitando para uma tecnologia igualmente inovadora, o Quick Response Code (QR Code) é um tipo de código de barras bidimensional que pode ser digitalizado e lido por dispositivos eletrónicos equipados com câmara e *software* específico. Conforme descrito pela Denso Wave (2021, para. 1-7), "a empresa japonesa responsável pela invenção do QR Code em 1994, Este código é constituído por uma matriz de pontos pretos e brancos, dispostos segundo um padrão quadrado que obedece à proporção de 1:1:3:1:1". Estes pontos possuem a capacidade de codificar uma multiplicidade de tipos de informação, abrangendo desde ligações para sítios eletrónicos, números de telefone, endereços de correio eletrónico, textos, até ficheiros de imagem. Adicionalmente, é possível a leitura eficaz do mesmo com até trinta por cento do código a apresentar-se ilegível.

A popularidade global do QR Code deve-se ao fato da Denso Wave (2021, par.15) "ter abdicado dos seus direitos de patente sobre esta tecnologia e à sua adoção pela indústria automobilística para uso em seu 'Kanban' eletrónico, assim como pelas indústrias alimentar e farmacêutica".

Yiğit (2021, para. 18-21) afirmou que “Portugal tem sido pioneiro nas obrigações de IVA e indicou que os grandes contribuintes já são obrigados a seguir o Facturamento Eletrônico obrigatório em transações Business-to-Government (B2G)”.

Code	Description
A	Issuer tax ID
B	Customer tax ID
C	Customer country
D	Document type
E	Document status
F	Document date
G	Invoice number
H	ATCUD
I1, J1, K1	Tax country region
I2, J2, K2	Tax base, exempt
I3, J3, K3	Tax base, reduced rate
I4, J4, K4	VAT amount, reduced rate

Code	Description
I5, J5, K5	Tax base, intermediate rate
I6, J6, K6	VAT amount, intermediate rate
I7, J7, K7	Tax base, standard rate
I8, J8, K8	VAT amount, standard rate
L	Amount not subject to VAT
M	Stamp duty
N	Total amount of tax (VAT + stamp duty)
O	Gross amount
P	Withholding tax
Q	4 characters from *Hash*
R	Software certificate number
S	Other information e.g. IBAN

A:123456789*B:999999990*C:PT*D:FT*E:N*F:20191231*G:FT AB2019/0035*H:CSDF7T5H0035*I1:PT*I2:12000.00*I3:15000.00*I4: 900.00*I5:50000.00*I6:6500.00*I7:80 000.00*I8:18400.00*J1:PTAC*J2:10000.00*J3:25000.56*J4:1000.02* J5:75000.00*J6:6750.00*J7:10000 0.00*J8:18000.00*K1:PTMA*K2:5000.00*K3:12500.00*K4:625.00*K5: 25000.00*K6:3000.00*K7:40000. 00*K8:8800.00*L:100.00*M:25.00*N:64000.02*O:513600.58*P:100.0 0*Q:kLp 0*R:9999*S:TB;PT000000000000000000000000;513500.58

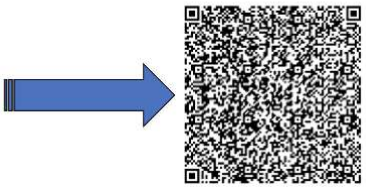


Figura 7 - Código ATCUD (Yiğit, 2021, para.17)

Yigit (2021, para. 18) considera os exemplos acima de iniciativas fiscais digitais e menciona que não é surpreendente que Portugal seja pioneiro na inclusão de códigos QR nas faturas. Além disso, prevê que é muito provável que outros países sigam o exemplo português e também considerem a introdução de códigos QR.

A Autoridade Tributária e Aduaneira (AT, 2023) disponibilizou as especificações técnicas para a elaboração do código de barras bidimensional (código QR) que resulta das Regras de faturação (D.L. n.º 28/2019, de 15/02) que serviu como base para a estrutura de dados SQL (Anexo 2).

Para guardar os dados o SQLite, uma biblioteca que oferece um motor de base de dados SQL embutido e autónomo, destaca-se pela sua capacidade de operar sem necessidade de configuração prévia, diretamente sobre ficheiros de disco convencionais. Esta solução, de acordo com uma grande maioria de programadores é robusta, eficaz e simples para a gestão de bases de dados é ideal para aplicações Android, dado que facilita a integração e manipulação de dados sem os encargos de manutenção típicos de sistemas de base de dados mais complexos. Além disso, a sua natureza de código aberto e a compatibilidade multiplataforma tornam-no uma escolha versátil e acessível para programadores, que procuram maximizar a eficiência e performance das suas aplicações móveis (SQLite, 2019).

O SQLCipher é uma extensão de código aberto do SQLite, que incorpora encriptação AES de 256 bits de forma transparente às bases de dados, apresentando um design resistente a manipulações. Esta ferramenta é compatível com várias plataformas e não necessita de configuração adicional, estando presente em milhares de aplicações e em centenas de milhões de dispositivos. “Oferece segurança robusta, empregando práticas como o modo CBC e a derivação de chaves, e é compatível com uma vasta gama de plataformas de desenvolvimento e linguagens de programação” (Zetetic, LLC, 2024, para.1-9).

Esta revisão destaca a utilização do QR Code como elemento fundamental da aplicação, justificando a escolha do Android e Firebase para o desenvolvimento, o que leva à reflexão sobre a eficácia global do projeto.

Em conclusão a revisão da literatura forneceu uma fundamentação sólida que justificou a escolha do Android como plataforma para o desenvolvimento da InvoiceSafe. A combinação da sua flexibilidade, escalabilidade e evolução contínua do sistema operativo alinhou-se perfeitamente com as necessidades do projeto, garantindo que a aplicação não só era relevante no contexto atual, mas também preparada para se adaptar às exigências tecnológicas futuras. A implementação do Firebase como solução de autenticação reforça a robustez do sistema em termos de segurança e gestão de utilizadores, que são fundamentais para o sucesso da aplicação em ambientes cada vez mais digitais e interligados.

3. Materiais e Métodos de Desenvolvimento

Esta secção descreve os materiais, recursos e abordagens metodológicas adotadas durante o desenvolvimento da aplicação móvel 'InvoiceSafe', destinada à gestão integrada de finanças pessoais, documentação médica e leitura de códigos QR presentes em faturas, incluindo a gestão de garantias de produtos.

Adotou-se uma abordagem metodológica qualitativa e descritiva, configurando-se como um estudo de caso, com o intuito de abranger as várias dimensões associadas ao objeto de estudo. Alguns investigadores, como Simione (2020, p.86) inspirados pela potencialidade da metodologia qualitativa na descrição da realidade, acreditam que se baseia na observação "de procedimentos dinâmicos cujos elementos provocam novas interpretações teóricas. Além disso, baseia-se na análise das conexões comunicativas e sociais entre indivíduos dentro de um contexto específico, abordado através de uma perspetiva interpretativa, descritiva, indutiva e realista., "preocupando-se com o estudo de pessoas, casos, fenômenos, situações sociais e processos em suas configurações naturais" (Simione, 2020, p.86).

E mais concretamente sobre estudo de caso:

> Os estudos de caso apresentam-se como recurso de pesquisa apropriado quando se colocam questões do tipo 'como' e 'porque' (essência daquilo que realmente se está interessado em responder durante o processo de estudo) e o investigador interessado tem pouco controle sobre os eventos, sendo que o foco se encontra em fenômenos contemporâneos inseridos em algum contexto da vida real (Simione, 2020, p.93).

Este método revelou-se fundamental para captar a profundidade das experiências individuais bem como para possibilitar a extensibilidade dos resultados a um contexto mais amplo.

Durante a fase inicial de conceção da aplicação 'InvoiceSafe', realizou-se um inquérito para identificar as preferências dos potenciais utilizadores em diversos aspetos da aplicação, incluindo o sistema operativo dos dispositivos móveis. A predominância do uso constante de dispositivos móveis sublinha a relevância de desenvolver uma aplicação móvel que seja acessível e eficiente.

O inquérito, realizado online e de forma anónima, constituiu uma parte essencial da recolha de dados. Este inquérito incluiu questões relacionadas com a faixa etária, género, nível de escolaridade, ocupação atual, frequência de utilização de dispositivos móveis, sistema

operativo dos dispositivos utilizados, uso de aplicações de gestão financeira, preferências por determinadas funcionalidades, preocupações com a segurança dos dados, e a frequência de sincronização dos mesmos.

Tabela 1 - Recolha de dados sobre a APP Android de literacia financeira (Brito, 2024, pp.5-9).

Princípios de Desenvolvimento	**Questões**
Perfil Pessoal?	- Qual é a sua faixa etária? - Qual é o seu género? - Qual é o seu nível de escolaridade? - Qual é a sua ocupação atual?
Uso de Dispositivos Móveis?	- Com que frequência utiliza dispositivos móveis como smartphones ou tablets? - Qual o volume de dados que estima usar mensalmente no dispositivo móvel (incluindo navegação na web, uso de apps, streaming de vídeos, etc.)? - Qual a velocidade média de sua conexão à internet no dispositivo móvel?
Tipo de Aplicações – Literacia Financeira?	- Já utiliza alguma aplicação para gerir suas finanças pessoais (ex.: faturas, bancos, investimentos)? - Quais funcionalidades considera essenciais numa aplicação de gestão de documentos financeiros e médicos? - Estaria disposto(a) a pagar por uma aplicação que oferecesse estes serviços?
Qual o SO mais usado nos smartphones pelas pessoas?	- Que sistema operativo utiliza no seu smartphone ou tablet?
Qual o público-alvo deste tipo de aplicação?	- Qual é a sua faixa etária? - Qual é o seu nível de escolaridade? - Qual é a sua ocupação atual? - Com que frequência utiliza dispositivos móveis como smartphones ou tablets?
Faz sentido promover a aglutinação e compilação das faturas, garantias numa base de dados na vez de guardar na galeria de fotos dentro de um dispositivo móvel?	- Quais funcionalidades considera essenciais numa aplicação de gestão de documentos financeiros e médicos? - Tem alguma sugestão ou comentário que gostaria de adicionar sobre o tipo de aplicação que estamos desenvolvendo?
Que tipo de estrutura de dados será mais adequada para assegurar as funcionalidades da aplicação?	- Quais funcionalidades considera essenciais numa aplicação de gestão de documentos financeiros e médicos? - Quais são as suas principais preocupações ao utilizar uma aplicação para gestão de documentos pessoais?
Preocupações com a Segurança dos Dados	- Quais são as suas principais preocupações ao utilizar uma aplicação para gestão de documentos pessoais?

	- Preferiria uma aplicação que sincronize os dados automaticamente sempre que alterados ou uma que permita escolher quando sincronizar?
Frequência de Sincronização e Atualização de Dados	- Com que frequência sincroniza ou atualiza dados em aplicações móveis que utiliza (como e-mail, apps de mensagens, apps financeiras, etc.)?
Integração com Outros Serviços Digitais	- Quais funcionalidades considera essenciais numa aplicação de gestão de documentos financeiros e médicos? (ex.: integração com e-fatura, bancos, seguros)
Disposição para Investir em Aplicações	- Estaria disposto(a) a pagar por uma aplicação que oferecesse estes serviços?
Problemas de Conectividade e Performance	- Com que frequência encontra problemas de conectividade ou velocidade lenta de internet ao usar aplicativos no seu smartphone ou tablet?
Preferências de Interface e Usabilidade	- Tem alguma sugestão ou comentário que gostaria de adicionar sobre o tipo de aplicação que estamos desenvolvendo? (isso pode dar insights sobre preferências de design de interface e usabilidade)
Volume de Dados e Impacto na Tarifa de Dados Móveis	- Qual o volume de dados que estima usar mensalmente no dispositivo móvel (incluindo navegação na web, uso de apps, streaming de vídeos, etc.)?
Velocidade e Qualidade da Conexão de Internet	- Qual a velocidade média de sua conexão à internet no dispositivo móvel?
Preferências sobre a Sincronização Automática ou Manual dos Dados	- Preferiria uma aplicação que sincronize os dados automaticamente sempre que alterados ou uma que permita escolher quando sincronizar?

O inquérito de levantamento de requisitos para o desenvolvimento da aplicação foi realizado de forma anónima e online, com o objetivo de recolher dados sobre as preferências e comportamentos dos utilizadores relativamente a uma aplicação móvel destinada à gestão de documentos financeiros e médicos. A amostra incluiu diversas faixas etárias, níveis de escolaridade e contextos profissionais, assegurando uma diversidade que contribui para a representatividade dos dados. Os gráficos dos resultados completos podem ser consultados no

Questionário de investigação sobre Gestão Digital de Documentos

Caro(a) participante,

Obrigado por participar neste inquérito, parte de um projeto académico do curso de Engenharia Multimédia no Instituto Superior de Tecnologias Avançadas de Lisboa. A sua participação é crucial para o sucesso deste estudo, que explora a aceitação de uma nova aplicação móvel para a gestão de documentos importantes, facilitando a organização pessoal e contribuindo para uma melhor gestão financeira e de saúde.

Este inquérito permitirá compreender melhor as preferências e necessidades do nosso público-alvo, ajudando-nos a desenvolver uma aplicação que seja tanto segura quanto intuitiva.

Por favor, responda honestamente às questões seguintes, asseguramos a confidencialidade de todas as suas respostas. Agradecemos imensamente o seu tempo e colaboração.

* Required

Parte 1: Informação Demográfica

1. Qual é a sua faixa etária? *

- ◯ Menos de 18 anos
- ◯ 18-24 anos
- ◯ 25-34 anos
- ◯ 35-44 anos
- ◯ 45-54 anos
- ◯ 55-64 anos
- ◯ 65 anos ou mais

2. Qual é o seu género? *

- ◯ Masculino
- ◯ Feminino
- ◯ Prefiro não dizer
- ◯ Other

3. Qual é o seu nível de escolaridade? *

- ◯ Ensino Básico Incompleto
- ◯ Ensino Básico Completo
- ◯ Ensino Secundário Incompleto
- ◯ Ensino Secundário Completo
- ◯ Ensino superior incompleto
- ◯ Ensino superior completo
- ◯ Pós-graduação

4. Qual é a sua ocupação atual? *

- ◯ Estudante
- ◯ Empregado(a)
- ◯ Trabalhador(a) Independente
- ◯ Desempregado(a)
- ◯ Reformado(a)
- ◯ Other

5. Com que frequência utiliza dispositivos móveis como smartphones ou tablets? *

- ◯ Nunca
- ◯ Raramente
- ◯ Mensalmente
- ◯ Semanalmente
- ◯ Diariamente
- ◯ Constantemente ao longo do dia

6. Que sistema operativo utiliza no seu smartphone ou tablet? *

- ◯ Android
- ◯ iOS (Apple)
- ◯ Windows Mobile
- ◯ Other

7. Já utiliza alguma aplicação para gerir suas finanças pessoais (ex: faturas, bancos, investimentos)? *

- ◯ Sim
- ◯ Não

8. Quais funcionalidades considera essenciais numa aplicação de gestão de documentos financeiros e médicos? (Marque todas as que se aplicarem) *

- ☐ Leitura automática de códigos QR
- ☐ Inserção manual de dados
- ☐ Associação de fotos aos documentos
- ☐ Armazenamento seguro de dados
- ☐ Pesquisa avançada de documentos
- ☐ Lembretes de pagamentos ou vencimentos de garantias
- ☐ Integração com outros serviços digitais (e-fatura, bancos, seguros)

9. Estaria disposto(a) a pagar por uma aplicação que oferecesse estes serviços? *

- ◯ Sim
- ◯ Não
- ◯ Depende do preço

10. Quais são as suas principais preocupações ao utilizar uma aplicação para gestão de documentos pessoais? *

- ◯ Segurança dos dados
- ◯ Facilidade de uso
- ◯ Custo da aplicação
- ◯ Fiabilidade do serviço
- ◯ Other

11. Com que frequência sincroniza ou atualiza dados em aplicações moveis que utiliza (como e-mail, apps de mensagens, apps financeiras, etc.)? *

- ◯ Nunca
- ◯ Raramente
- ◯ Mensalmente
- ◯ Semanalmente
- ◯ Diariamente
- ◯ Várias vezes ao dia

12. Qual o volume de dados que estima usar mensalmente no dispositivo móvel (incluindo navegação na web, uso de apps, streaming de vídeos, etc.)? *

- ◯ Menos de 500 MB
- ◯ 500 MB a 1 GB
- ◯ 1 GB a 3 GB
- ◯ 3 GB a 5 GB
- ◯ Mais de 5 GB

13. Qual a velocidade média de sua conexão à internet no dispositivo móvel? *

- ◯ Muito lenta (menos de 3 Mbps)
- ◯ Lenta (3 a 10 Mbps)
- ◯ Moderada (10 a 25 Mbps)
- ◯ Rápida (25 a 50 Mbps)
- ◯ Muito rápida (mais de 50 Mbps)

14. Com que frequência encontra problemas de conectividade ou velocidade lenta de internet ao usar aplicativos no seu smartphone ou tablet? *

- ◯ Nunca
- ◯ Raramente
- ◯ Às vezes
- ◯ Frequentemente
- ◯ Sempre

15. Preferiria uma aplicação que sincronize os dados automaticamente sempre que alterados ou uma que permita escolher quando sincronizar? *

- ◯ Sincronização automática sempre que houver alterações
- ◯ Preferiria escolher manualmente quando sincronizar
- ◯ Indiferente

16. Tem alguma sugestão ou comentário que gostaria de adicionar sobre o tipo de aplicação que estamos desenvolvendo?

This content is neither created nor endorsed by Microsoft. The data you submit will be sent to the form owner.

Microsoft Forms

Apêndice B.

A população do estudo abrangeu uma vasta gama de idades, destacando-se a participação significativa dos grupos de 25-34 e 45-54 anos, o que sugere um interesse generalizado pela gestão digital de documentos. A distribuição por género foi equilibrada, refletindo a acessibilidade e a pertinência da aplicação para ambos os sexos. Os níveis de escolaridade dos participantes variaram desde o ensino secundário incompleto até ao nível pós-graduado, indicando que a aplicação deve ser intuitiva e acessível a utilizadores com variados níveis de educação formal.

As medidas foram cuidadosamente selecionadas para garantir que o desenvolvimento da aplicação estivesse fundamentado num entendimento robusto das necessidades e expectativas dos utilizadores finais. O cronograma seguinte ilustra a evolução temporal do projeto, no qual foram despendidas aproximadamente 3256 horas.

Nome do Projeto	InvoiceSafe
Nome do Aluno	Ruben Alves Brito
Prazo do Projeto	30 de Junho de 2024
Declaração do Âmbito	Projeto Académico
Início	1 de outubro de 2022
Término	30 de Junho de 2023
Progresso	100%

Tarefa	Responsável	Início	Término	Dias	Status	out/22	nov/22	dez/22	jan/23	fev/23	mar/23	abr/23	mai/23	jun/23
Pré-requisitos														
Pesquisa	Ruben Brito	1/out	1/dez	61	Completo									
Apresentação do tema.aceitação do tema	Ruben Brito	1/out	1/dez	61	Completo									
Iniciação														
Desenvolvimento do pré projeto	Ruben Brito	1/out	1/dez	61	Completo									
Entrega do pré-projecto	Ruben Brito	13/mar	19/mar	6	Completo									
apresentação do Pitch	Ruben Brito	23/mar	31/mai	69	Completo									
Desenvolvimento														
Pesquisa	Ruben Brito	1/mar	31/mar	30	Completo									
Desenvolvimento	Ruben Brito	1/abr	31/mai	60	Completo									
Teste	Ruben Brito	1/mai	31/mai	30	Completo									
Desenvolvimento Concluído	Ruben Brito	1/jun	30/jun	29	Completo									
Lançamento		30/jun	30/jun											

Figura 8: Cronograma da implementação da App Android InvoiceSafe (Brito, 2024, p. 36)

Começou-se por esquematizar os ecrãs, a fim de avaliar a sua interligação. Aqui é possível perceber a relação que existe entre os utilizadores e os diferentes ecrãs, que o conduzem ao longo da aplicação. O resultado deste trabalho está ilustrado no Wireframe apresentado na Figura 34.

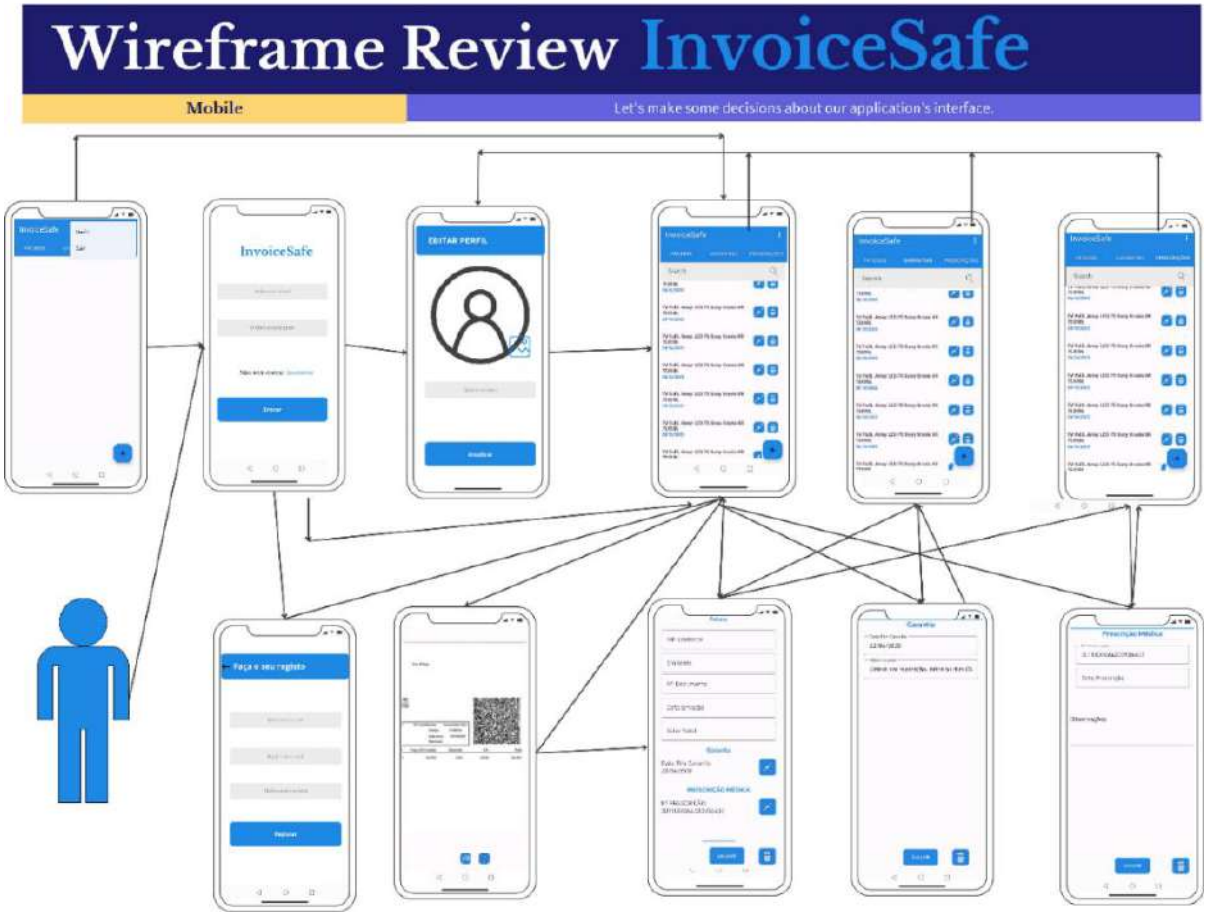


Figura 9: Wireframe review InvoiceSafe (Brito, 2024, p. 25)

A segurança e a privacidade dos dados inseridos foram garantidas pelo Firebase Auth, o qual proporcionou uma abordagem robusta e segura na gestão de acessos. Quanto aos dados locais, a privacidade foi assegurada por meio da implementação de encriptação na base de dados SQLite, utilizando o SQLCipher.

4. Resultados e Discussão de Resultados

Para entender como se enquadravam as necessidades dos utilizadores com o desenvolvimento da aplicação InvoiceSafe, apresentam-se aqui as respostas do inquérito efetuado. Os resultados mais significativos contribuíram para um desenvolvimento mais adequado ao perfil dos potenciais utilizadores.

A maioria dos inquiridos relatou utilizar dispositivos móveis constantemente, o que reforça a importância de uma aplicação móvel neste contexto. O sistema operativo mais utilizado foi o Android, embora uma proporção significativa de utilizadores também utilize iOS, sugerindo a necessidade de desenvolvimento *cross-platform* para abranger um público mais vasto.

Nas figuras subsequentes, apresentam-se os resultados obtidos a partir das respostas dos inquiridos. A análise das faixas etárias revela uma distribuição diversificada, com uma prevalência notável na faixa dos 35 aos 44 anos, seguida pelas faixas dos 45 aos 54 anos e dos 25 aos 34 anos. As categorias etárias dos 18 aos 24 anos e dos 55 aos 64 anos também estão representadas, embora em menor proporção, enquanto as extremidades abaixo dos 18 anos e acima dos 65 anos registam a menor frequência de respostas.

Figura 10: Faixa Etária dos respondentes (Brito, 2024, p.5).

A proporção entre os géneros mostrou-se equilibrada, refletindo a diversidade do grupo de respondentes.

Figura 11: Género dos Participantes (Brito, 2024, p.5).

A maioria dos participantes possuía formação superior, indicando um nível educacional elevado entre os respondentes.

Figura 12: Nível de escolaridade (Brito, 2024, p.5).

Predominaram empregados e autônomos, sugerindo que a utilização de dispositivos móveis e aplicativos é frequente entre profissionais ativos.

Figura 13: Ocupação atual (Brito, 2024, p.6).

Foi relatado uso diário e constante de dispositivos móveis, sublinhando a integração destes na rotina diária dos indivíduos.

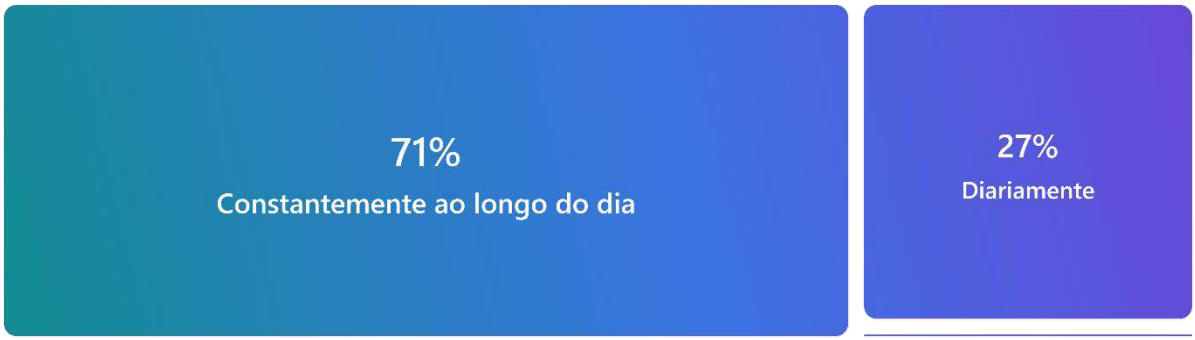


Figura 14: Frequência de Utilização de Dispositivos Móveis (Brito, 2024, p.6).

Android foi o sistema operacional mais utilizado, seguido de iOS, refletindo a popularidade destas plataformas no mercado.

Figura 15: Sistema Operativo Utilizado (Brito, 2024, p.6).

Uma significativa parcela dos inquiridos já utilizava aplicações para gestão de finanças, evidenciando uma tendência de digitalização das finanças pessoais.

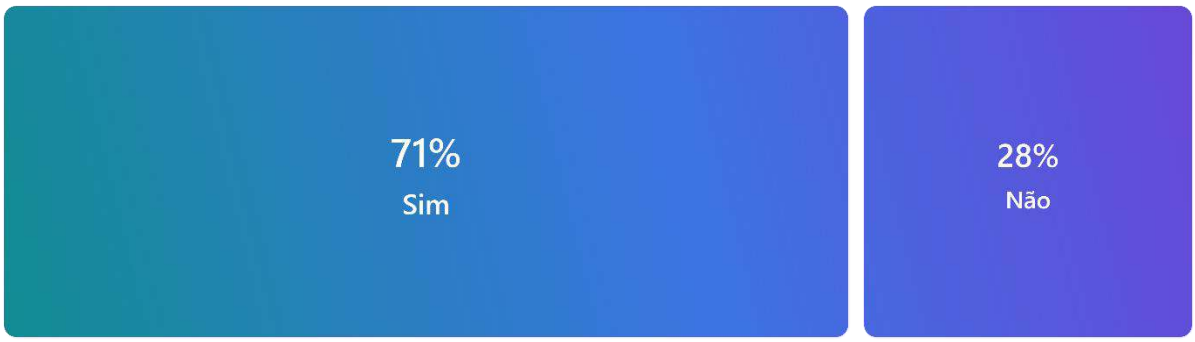


Figura 16: Uso de Aplicações de Gestão Financeira (Brito, 2024, p.6).

Funcionalidades como armazenamento seguro de dados e integração com outros serviços

digitais foram altamente valorizadas, destacando a importância da segurança e da conectividade.

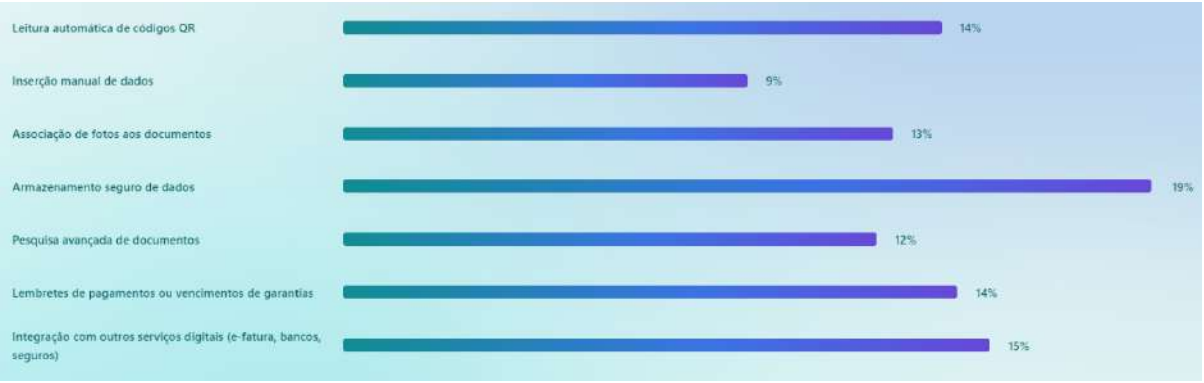


Figura 17: Funcionalidades Essenciais em Aplicações de Gestão (Brito, 2024, p.7)

A disposição para pagar dependeu do preço, indicando que os consumidores são sensíveis ao custo, embora reconheçam o valor de tais aplicativos.

Figura 18: Disposição para Pagar por Aplicações (Brito, 2024, p.7).

A segurança dos dados emergiu como a principal preocupação, refletindo conscientização sobre riscos de privacidade e segurança.

Figura 19: Preocupações ao Utilizar Aplicações de Gestão de Documentos (Brito, 2024, p.7).

A maioria dos respondentes atualizava dados em aplicações móveis diariamente, ilustrando a necessidade de acesso contínuo e atualizado às informações.

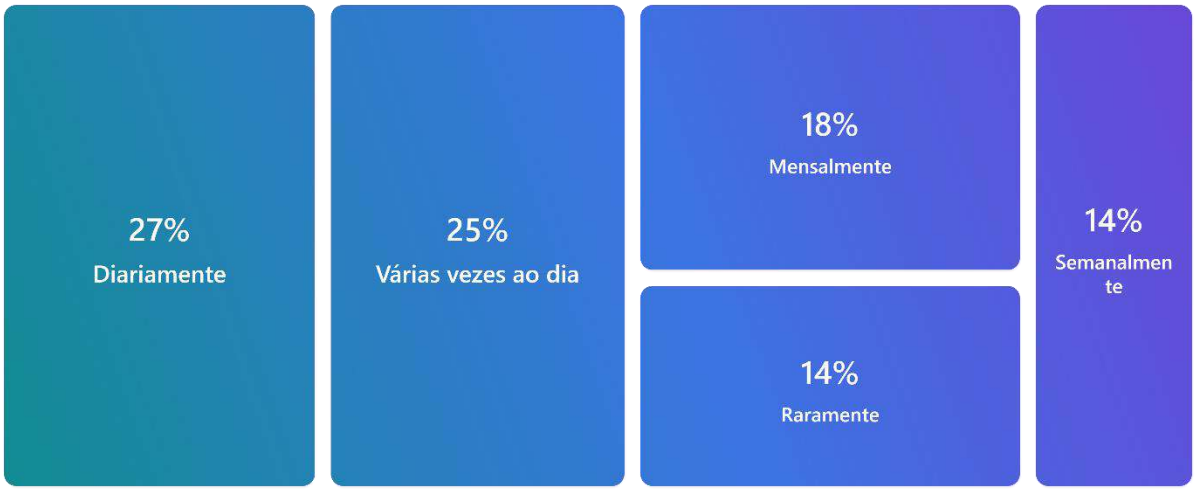


Figura 20: Frequência de Sincronização de Dados em Aplicações (Brito, 2024, p.8).

O volume de dados mensal variou, com muitos utilizadores a consumir mais de 5 GB, o que pode ser indicativo do alto consumo de media e serviços online.

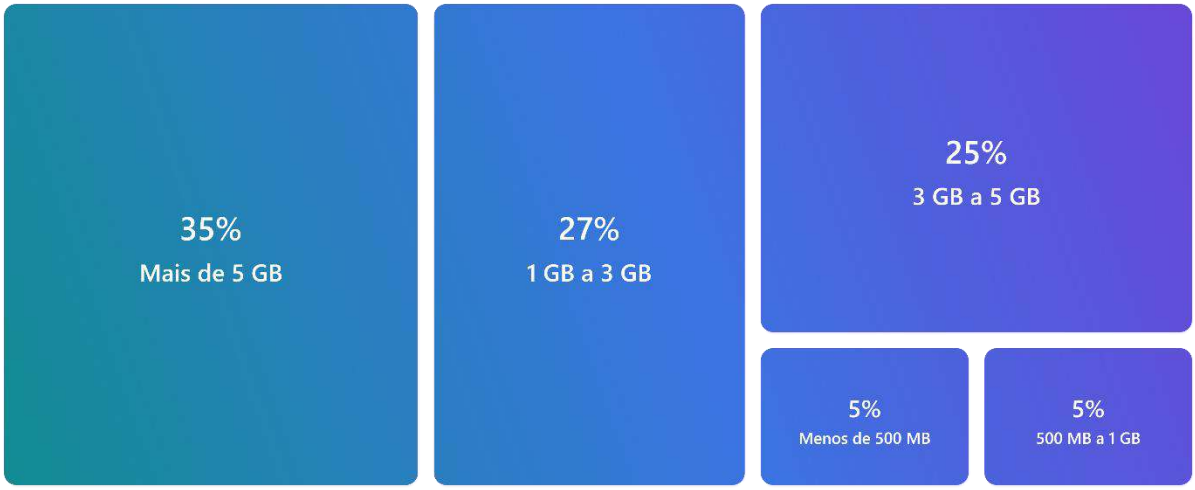


Figura 21: Volume de Dados Usados Mensalmente, em dispositivos móveis (Brito, 2024, p.8).

As velocidades de conexão à internet variaram de moderada a muito rápida, afetando diretamente a experiência do utilizador ao aceder a serviços online.

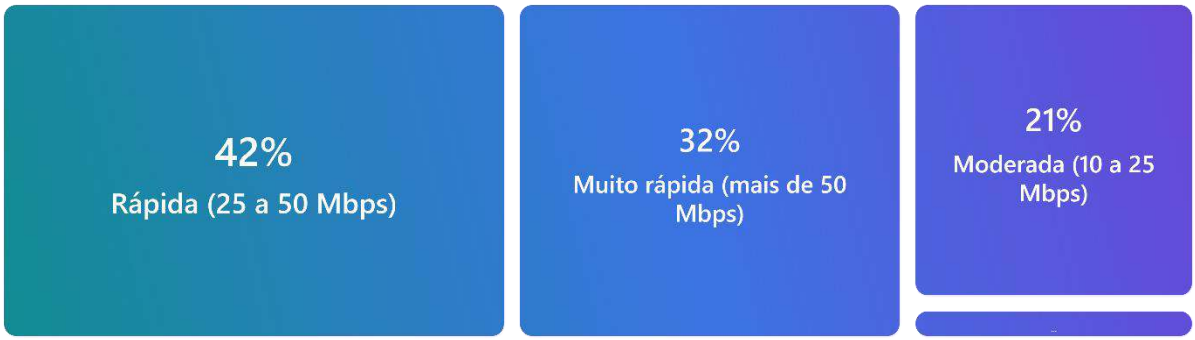


Figura 22: Velocidade média da ligação à internet, em dispositivos móveis. (Brito, 2024, p.8).

Problemas de conectividade foram ocasionalmente experienciados por alguns utilizadores, impactando a utilização eficaz de aplicações móveis.

Figura 23: Problemas de Conectividade ou Velocidade, em dispositivos móveis. (Brito, 2024, p.9).

Problemas de conectividade foram ocasionalmente experienciados por alguns utilizadores, impactando a utilização eficaz de aplicações móveis.

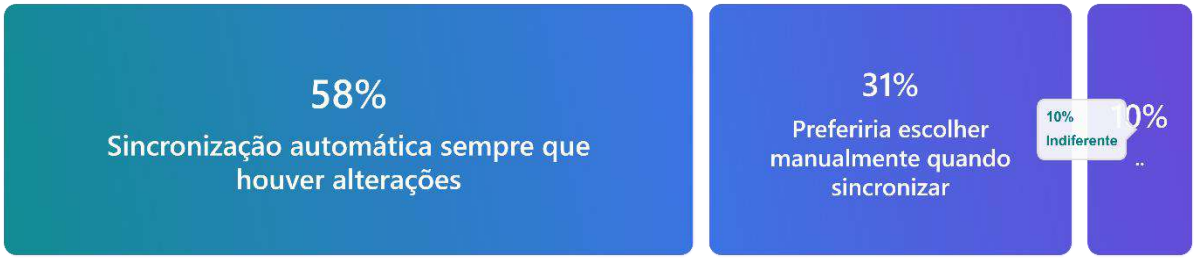


Figura 24: Preferências de sincronização automática versus manual em aplicações. (Brito, 2024, p.9).

As questões de resposta aberta não demonstraram elevada utilidade. Contudo, visto que não eram obrigatórias, registaram-se algumas respostas que proporcionaram insights relevantes acerca da usabilidade.

12 Responses

ID ↑	Name	Responses
1	anonymous	Solução para gravar os itens no caso de querer rever no final e não ter de fazer tudo outra vez
2	anonymous	Não
3	anonymous	Não
4	anonymous	Interessante
5	anonymous	PoupaTempo
6	anonymous	Não tenho.
7	anonymous	Integração com email
8	anonymous	Não
9	anonymous	Bom dia, espero ter sido útil, obrigado.
10	anonymous	T
11	anonymous	Não
12	anonymous	Nada.

Figura 25: Respostas livre sobre funcionalidades (Brito, 2024, p.9).

4.1. Como funciona a aplicação InvoiceSafe

A partir das respostas ao inquérito, identificaram-se os seguintes requisitos que a aplicação 'InvoiceSafe' deveria incorporar.

Ecrãs de Interface Adicionais:

- Ecrã de configurações: Permitiria aos utilizadores personalizar as opções de sincronização e notificações.
- Ecrã de ajuda e suporte: Disponibilizaria informações para orientar os utilizadores no uso da aplicação e na resolução de problemas comuns.
- Ecrã de estatísticas e relatórios: Apresentaria resumos financeiros e lembretes de vencimentos, baseados nas faturas e prescrições armazenadas.

Funcionalidades de Gestão Avançadas:

- Lembretes automáticos: Enviaria notificações ao utilizador, baseadas nas datas de vencimento de garantias e pagamentos.

- Associação de fotos e documentos: Permitiria aos utilizadores anexar fotos ou documentos digitalizados a uma entrada de fatura ou garantia para referência futura.
- Exportação de dados: Os utilizadores poderiam exportar suas faturas e documentos relacionados para formatos como CSV ou PDF.

Modelagem e Design da Arquitetura:

- Integração com serviços em nuvem: Para backup e restauração de dados, permitindo a sincronização entre dispositivos.
- Suporte para múltiplas línguas e moedas: A fim de atender a uma base de utilizadores global e diversificada.

Desenvolvimento de Funcionalidades Adicionais:

- Pesquisa avançada: Capacidade de pesquisar documentos por data, valor, categoria ou outros meta dados.
- Categorização automática: Uso de inteligência artificial para classificar automaticamente os documentos digitalizados em categorias relevantes.
- Integração com sistemas bancários e de seguros: Para importação automática de documentos e informações relevantes.

Melhorias na Segurança:

- Encriptação da Base de dados local: Garantiria que os dados locais estivessem salvaguardados contra acessos não autorizados.
- Encriptação de ponta a ponta: Asseguraria que os dados estivessem protegidos tanto em trânsito quanto em repouso.
- Controlo de acesso baseado em funções: Permitiria aos utilizadores definir permissões para diferentes tipos de informações, protegendo dados sensíveis.

Otimização do Desempenho:

- Gestão eficiente de dados móveis: A aplicação seria otimizada para o uso de dados, reduzindo o consumo em conexões limitadas ou de baixa velocidade.
- Performance em diversas condições de rede: Funcionaria de forma estável independentemente da velocidade da internet, com funcionalidades offline quando necessário.
- Após a definição dos requisitos, iniciou-se o desenho da arquitetura da base de dados, concebida com base nos requisitos técnicos fornecidos pela Autoridade Tributária e Aduaneira (2023, para. 1).

De seguida, elaborou-se um protótipo para testar os métodos CRUD da base de dados na plataforma sqliteonline.com.

Inicialmente, considerou-se a criação de uma tabela 'invoice' com todos os parâmetros indicados pela Autoridade Tributária e Aduaneira (2023). No entanto, verificou-se que isso resultaria numa tabela com muitos parâmetros vazios, como se pode verificar na Figura 27, o que não seria benéfico para a performance. Assim, optou-se pela estratégia de incluir um novo parâmetro para armazenar o texto completo obtido a partir do QR code. Esta abordagem assegura que, caso no futuro seja necessário implementar novas funcionalidades que requeiram parâmetros adicionais, apenas será necessário adicionar novos campos à tabela. Desta forma, é possível resgatar a informação sem que seja necessário o utilizador carregar novamente as faturas, ficando todas as atualizações a cargo do script de atualização da aplicação. Para gerir essas atualizações futuras, foi criado o método **onUpgrade**.

```
override fun onUpgrade(db: SQLiteDatabase?, oldVersion: Int, newVersion: Int) {
    /*        db?.execSQL("DROP TABLE IF EXISTS $InvoiceContract.InvoiceEntry.TABLE_NAME_INVOICE")
            onCreate(db)*/
    if (oldVersion < 2) {
        db?.execSQL(createTableWarranty)
    }

}
```

Figura 26: Método onUpgrade (Brito, 2024, p.22).

```
CREATE TABLE invoice(
  IdI INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
  DatecreateRegister DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  DateUpdateRegister DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
  QRCodeText TEXT,
  NameIssuer TEXT,
  ProductDescription TEXT,
  warranty INTEGER DEFAULT 0,
  prescription INTEGER DEFAULT 0,
  A text,
  D text,  C text,  B text,  E text,
  F text,
  G text,
  H text,  I1 real,  I2 real,  I3 real,  I4 real,  I5 real,  I6 real,  I7 real,  I8 real,  J1 text,  J2 real,
  O real,
  P real,  Q text,  R text,  S text
);
PRAGMA foreign_keys = ON;

```

Figura 27: Tabela invoice (Brito, 2024, p.23).

Para assegurar a coerência dos dados, definiram-se restrições de integridade referencial nas tabelas 'warranty' e 'prescription' da base de dados SQLite. Optou-se por utilizar como chave estrangeira o 'id' da tabela 'invoice', com as opções de eliminar e atualizar em cascata. O

comando 'PRAGMA foreign_keys = ON;' foi incluído na função 'onOpen', que é responsável por abrir a ligação à base de dados, para garantir o cumprimento das restrições de chave estrangeira.

```
override fun onOpen(db: SQLiteDatabase) {
    super.onOpen(db)
    // Habilita o suporte a chaves estrangeiras
    if (!db.isReadOnly) db.execSQL( sql: "PRAGMA foreign_keys=ON;")
}
```

Figura 28: Função SQL onOpen (Brito, 2024, p.23).

```
CREATE TABLE IF NOT EXISTS prescription (
    idp INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    dateRegister DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    prescriptionDate TEXT NOT NULL,
    observations TEXT,
    prescriptionId TEXT NOT NULL,
    doctorName TEXT NOT NULL,
    medicineName TEXT NOT NULL,
    dosage TEXT NOT NULL,
    usageInstructions TEXT,
    clinicName TEXT,
    idi INTEGER,
    FOREIGN KEY (Idi) REFERENCES invoice(idi) ON DELETE CASCADE ON UPDATE CASCADE
);
```

Figura 29: Tabela prescription (Brito, 2024, p.23).

```
CREATE TABLE IF NOT EXISTS warranty (
    idw INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
    dateRegister DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    warrantyId TEXT,
    warrantyDurationMonths INTEGER DEFAULT 36,
    warrantyStartDate DATE,
    warrantyEndDate DATE,
    observations TEXT,
    idi INTEGER,
    FOREIGN KEY (idi) REFERENCES invoice(idi) ON DELETE CASCADE ON UPDATE CASCADE
);
```

Figura 30: Tabela warranty (Brito, 2024, p.24).

Ao criar as tabelas, verificou-se que seria conveniente utilizar uma quarta tabela para armazenar os dados dos emitentes, com o objetivo de diminuir o número de interações por parte

do utilizador. Dessa forma, faturas registadas posteriormente de um emitente já existente na base de dados terão o nome automaticamente preenchido.

```
CREATE TABLE IF NOT EXISTS issuer (
 ide INTEGER NOT NULL PRIMARY KEY AUTOINCREMENT,
 dateRegister DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
 a TEXT NOT NULL,
 issuerName TEXT,
 nif INT);
```

Figura 31: Tabela issuer (Brito, 2024, p.24).

Após a conclusão do esquema da base de dados, procedeu-se à sua implementação. Foram criadas classes dedicadas exclusivamente às transações SQL, separadas da lógica do negócio, visando facilitar futuras migrações de base de dados. A estruturação do SQL foi realizada da seguinte forma:

DBHelper: Utilizado para gerir a criação e a versão da base de dados, este componente estende a classe SQLiteOpenHelper, facilitando a abertura da base de dados existente, a sua criação quando inexistente, e a atualização conforme necessário.

Vantagens: Simplifica a gestão do ciclo de vida da base de dados, centraliza a criação de tabelas e versões num único local, facilitando manutenções e atualizações.

InvoiceDAO/WarrantyDAO/PrescriptionDAO (Data Access Object): Estas interfaces definem os métodos para aceder aos dados das faturas, garantias e prescrições médicas, incluindo inserção, atualização, exclusão e recuperação.

Vantagens: Isola o acesso aos dados do resto da aplicação, reduzindo o acoplamento entre a lógica de negócios e a base de dados. Facilita a manutenção e a simplifica os testes ao abstrair as operações de base de dados.

IInvoiceDAO/IWarrantyDAO/IPrescriptionDAO: Uma variante da interface InvoiceDAO que permite definir um contrato mais específico ou detalhado para operações com faturas, garantias e prescrições médicas.

Vantagens: Oferece flexibilidade ao permitir diferentes implementações que podem ser otimizadas ou adaptadas conforme as necessidades específicas do projeto.

InvoiceContract: Este componente define constantes que auxiliam no acesso seguro à base de dados, como os nomes das tabelas e das colunas.

Vantagens: Promove a utilização de nomes consistentes nas várias partes da aplicação, minimizando erros de digitação e simplificando as refatorações no código. Aumenta a legibilidade do código e facilita a colaboração entre programadores.

O sistema de login, essencial para a segurança, foi inicialmente desenvolvido do zero. No entanto, após uma primeira tentativa, optou-se pela adoção do sistema de login do Firebase, centralizando assim a gestão de utilizadores nesta plataforma. Esta decisão baseou-se na segurança avançada que o Firebase proporciona, ao utilizar tecnologia de ponta e permitir uma economia de recursos. Ativou-se exclusivamente o ambiente de desenvolvimento, configurado com o mínimo de proteções, para facilitar a implementação e a realização dos testes alfa.

Figura 32: Ecrã de login (Brito, 2024, p.24).

Figura 33: Ecrã Registar o perfil (Brito, 2024, p.24).

A **classe LoginActivity** é responsável pela interface e lógica de autenticação de utilizadores num aplicativo Android que utiliza o Firebase para gestão de autenticação. Segue uma descrição das suas principais funções:

onCreate(Bundle savedInstanceState): Este método é acionado quando a atividade é criada inicialmente. Configura a interface do utilizador utilizando binding.root para inflar o layout associado, inicializa os eventos de clique dos botões e outros elementos interativos através do método initializeClickEvents(), e prepara o ActivityResultLauncher para lidar com o resultado de atividades iniciadas, incluindo logins com serviços externos como o Google.

onStart(): Executado quando a atividade está prestes a tornar-se visível ao utilizador. Verifica se o utilizador já está autenticado com CheckUserLogged(), redirecionando diretamente para a MainActivity se estiver autenticado.

CheckUserLogged(): Verifica a existência de um utilizador atualmente autenticado com o Firebase Auth. Se positivo, redireciona para a MainActivity.

initializeClickEvents(): Define os manipuladores para os eventos de clique nos elementos da UI, o que inclui abrir a atividade de registo e processar a tentativa de login quando o botão de login é pressionado. Métodos de login com o Google ou outros serviços também são configurados aqui.

loginUser(): Tenta autenticar o utilizador usando o Firebase Auth com o e-mail e senha fornecidos. Em caso de sucesso, redireciona para a MainActivity e exibe uma mensagem de sucesso. Em caso de falha, trata diferentes exceções relacionadas a credenciais inválidas ou utilizadores não encontrados, exibindo mensagens de erro adequadas.

validateFields(): Valida os campos de e-mail e senha preenchidos pelo utilizador. Verifica a completude dos campos e exibe mensagens de erro nos campos de entrada, se necessário. Retorna verdadeiro se os campos estiverem corretamente preenchidos, permitindo que o processo de login prossiga.

Cada uma destas funções desempenha um papel vital na autenticação do utilizador, assegurando que os dados de entrada sejam validados e que o processo de login seja gerido de forma eficaz e segura.

A **classe registerActivity** constitui-se como um componente de interface de utilizador na plataforma Android, integrando a biblioteca AppCompatActivity. O seu objetivo é gerir o registo de novos utilizadores através da autenticação Firebase e da subsequente gravação dos dados do utilizador no Firebase Firestore. Seguem-se descrições das funções incorporadas nesta classe:

onCreate(Bundle savedInstanceState): Este método é invocado durante a criação da atividade. Executa a inicialização da interface de utilizador, associando-a ao layout definido em ActivityRegisterBinding. Adicionalmente, invoca o método initializeToolbar() para configurar a barra de ferramentas e InitializeClickEvents() para definir os eventos de clique nos componentes interativos.

InitializeClickEvents(): Define a lógica de interação para o botão de registo. Verifica a validade dos campos de entrada através do método validateFields() e, se válidos, procede ao registo do utilizador utilizando a função registerUser(name, email, password).

registerUser(String, String, String): Responsável pelo registo de um novo utilizador utilizando o serviço FirebaseAuth. Em caso de sucesso na criação do utilizador, este método invoca saveUserFilestore(user) para armazenar os dados do utilizador no Firestore. Em caso de falha, trata as exceções específicas associadas ao processo de registo, como senha fraca, e-mail já utilizado ou credenciais inválidas, informando o utilizador através de mensagens apropriadas.

saveUserFilestore(User): Armazena os dados de um utilizador na coleção “users” do Firebase Firestore. Em caso de sucesso, redireciona para a MainActivity e apresenta uma mensagem de confirmação. Caso ocorra um erro durante o processo de armazenamento, informa o utilizador através de uma mensagem.

validateFields(): Verifica a completude e a validade dos campos de entrada para nome, e-mail e senha. Retorna verdadeiro apenas se todos os campos estiverem corretamente preenchidos, caso contrário, apresenta mensagens de erro específicas e impede o avanço do processo de registo.

initializeToolbar(): Configura a barra de ferramentas da atividade, definindo o título e ativando a opção de retorno à atividade anterior. A função ajusta as propriedades da barra de ferramentas para alinhar com o contexto de registo do utilizador.

Cada função delineada nesta descrição desempenha um papel crítico na facilitação do registo de novos utilizadores, garantindo uma interação fluída e eficiente com a plataforma Firebase, tanto para autenticação quanto para armazenamento de dados. As validações e tratamentos de erro incorporados asseguram uma experiência de utilizador robusta e segura.

A **classe ProfileActivity** é uma componente do Android destinada à manipulação dos dados de perfil dos utilizadores, integrando funcionalidades de autenticação e armazenamento providenciadas pelo Firebase. Esta atividade permite a atualização do nome do utilizador e a escolha e carregamento de uma imagem de perfil para o Firebase Storage, bem como a subsequente atualização dos dados de perfil no Firestore. As funcionalidades são asseguradas através de uma série de métodos, que são descritos detalhadamente a seguir:

onCreate(Bundle savedInstanceState): Este método é invocado quando a atividade é criada. Configura o conteúdo da vista utilizando binding.root para inflar o layout e chama initializeToolbar() para configurar a barra de ferramentas. Adicionalmente, requestPermissions() é chamado para solicitar as permissões necessárias ao utilizador e initializeEventsClick() para configurar os eventos de cliques.

onStart(): Método chamado quando a atividade se torna visível. Aqui, é invocado RecoverInitialUserData() para recuperar e exibir os dados existentes do utilizador.

RecoverInitialUserData(): Consulta o Firestore para obter os dados atuais do utilizador autenticado. Se os dados estiverem disponíveis, o nome é atualizado na interface e a imagem de perfil é carregada utilizando a biblioteca Picasso.

uploadImageStorage(Uri uri): Carrega uma imagem selecionada pelo utilizador para o Firebase Storage e, em caso de sucesso, obtém a URL da imagem carregada, atualizando o Firestore com o novo link da foto de perfil através do método updateProfileData().

updateProfileData(String idUser, Map<String, String> data): Atualiza os dados do perfil do utilizador no Firestore. Em caso de sucesso, uma mensagem de confirmação é exibida; se falhar, uma mensagem de erro é mostrada.

initializeEventsClick(): Configura os eventos de clique para os botões de seleção de imagem da galeria e atualização do perfil. Verifica as permissões necessárias e, se concedidas, permite a escolha de uma imagem. Também permite atualizar o nome do utilizador no Firestore se o campo de texto correspondente não estiver vazio.

requestPermissions(): Verifica as permissões de câmara e galeria, solicitando-as ao utilizador se não estiverem já concedidas. Utiliza o ActivityResultContracts.RequestMultiplePermissions para gerir as solicitações de múltiplas permissões simultaneamente.

initializeToolbar(): Configura a barra de ferramentas da atividade, estabelecendo o título da barra e a opção de retorno à atividade anterior.

Cada um destes métodos contribui para a funcionalidade global da atividade, facilitando a interação do utilizador com o seu perfil, permitindo a personalização dos dados visíveis e garantindo a segurança e a privacidade através do correto manejo das permissões.

A parte da imagem serviu de teste para avaliação da viabilidade em usar o Firebase Storage para guardar imagens das faturas e fotos dos respetivos produtos.

Após a definição da estratégia de login, elemento crucial para assegurar a segurança de acesso aos dados inseridos na aplicação InvoiceSafe, iniciou-se o desenvolvimento das funcionalidades essenciais da mesma. Começou-se por esquematizar os ecrãs, a fim de avaliar a sua interligação. O resultado deste trabalho está ilustrado no wireframe apresentado na Figura 34:

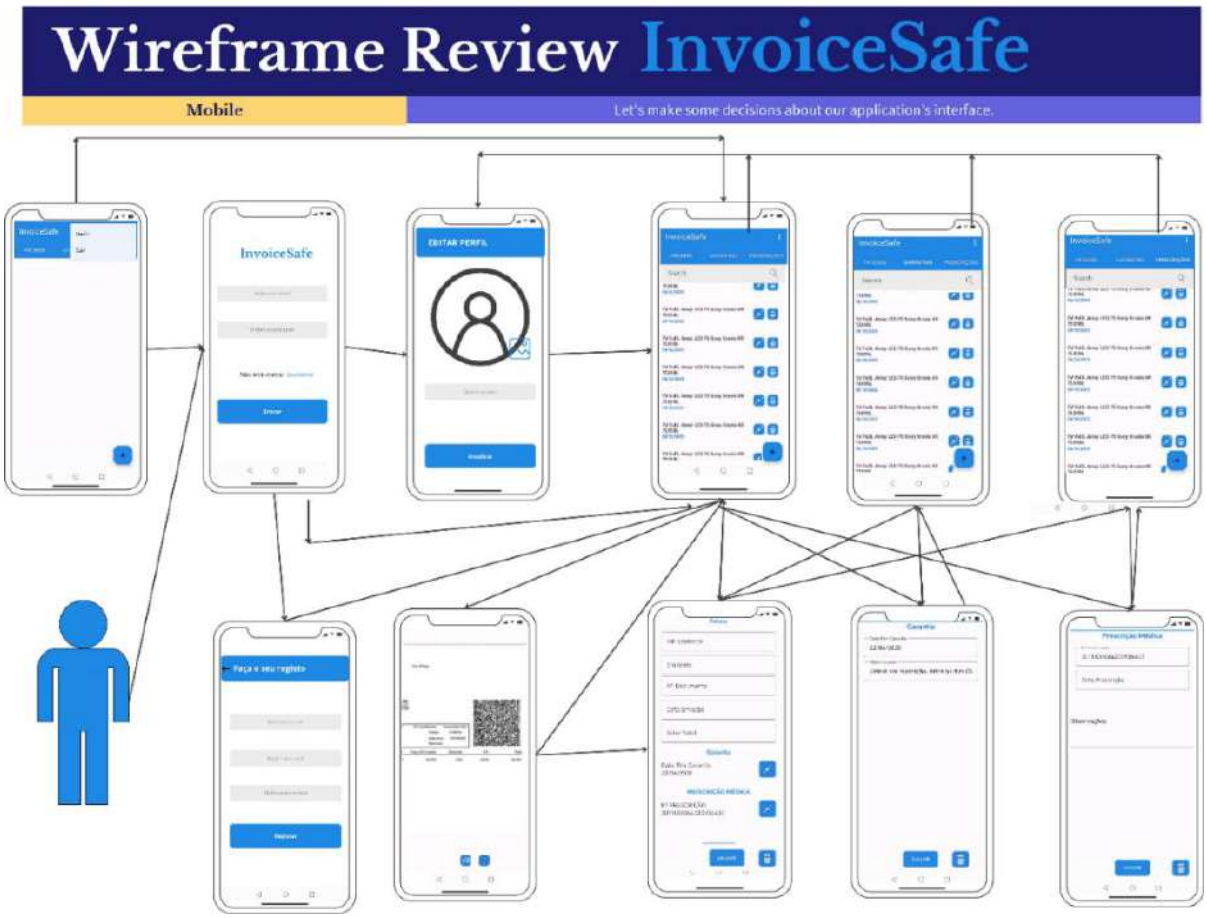


Figura 34: Wireframe review InvoiceSafe (Brito, 2024, p. 25)

Após a conclusão da definição dos ecrãs, iniciou-se a sua implementação e criação dos ids para usar nas lógicas a implementar.

De seguida, iniciou-se, em paralelo, as implementações da lógica para inserir/editar faturas e da lógica para capturar o QR Code, ler o conteúdo e extrair apenas a informação necessária para popular os parâmetros para inserir a fatura na base de dados (BD).

Aqui apareceu o primeiro grande desafio, foi necessário criar 'inteligência' para não duplicar registos de faturas, essa validação ficou na função saveInvoice() da classe AddInvoiceActivity, que verifica se existe algum registo da Identificação única do documento, caso não exista grava os dados na BD, caso exista recupera os dados e coloca no modo de edição da fatura.

A **classe AddInvoiceActivity**, localizada no pacote com.rfsab.invoicesafe, é uma atividade Android que administra a adição e edição de dados de faturas. A seguir, detalham-se as principais funções desta classe:

onCreate(Bundle savedInstanceState): Este método inicializa a atividade, configurando a interface do utilizador a partir de um arquivo de layout, estabelecendo um lançador de resultados de atividades, recuperando e utilizando o ID da fatura fornecido por uma Intent, se disponível, e preparando os campos para o registo de uma nova fatura.

prepareForNewInvoice(): Prepara a interface para a adição de uma nova fatura, preenchendo os campos da UI com dados fornecidos através da Intent.

configureActivityResultLauncher(): Configura um ActivityResultLauncher para gerir os resultados de outras atividades iniciadas para captura ou seleção de dados, atualizando a interface com os dados retornados.

loadInvoiceData(Long invoiceId): Carrega os dados de uma fatura existente usando o ID fornecido, preenchendo os campos da interface com esses dados.

setupButtonListeners(): Estabelece ouvintes de eventos para os botões na interface, incluindo ações para salvar e excluir faturas, capturar documentos usando a câmera, e configurar um seletor de datas.

saveInvoice(): Salva os dados da fatura no banco de dados. Este método determina se a entrada é nova ou uma atualização de uma fatura existente, constrói um objeto ModelInvoice, e utiliza o método adequado do InvoiceDAO para inserir ou atualizar a fatura no banco de dados.

deleteInvoiceById(Long id): Exclui uma fatura específica usando seu ID. Este método verifica o sucesso da exclusão e atualiza a interface de acordo.

setupDatePicker(): Configura um seletor de datas (DatePicker) para um campo de entrada de texto, facilitando a seleção de datas de forma visual e intuitiva pelo utilizador.

Cada função desempenha um papel essencial na gestão do ciclo de vida e nas interações do utilizador nesta atividade, contribuindo para uma interface de utilizador funcional e responsiva no gerenciamento de dados de faturas.

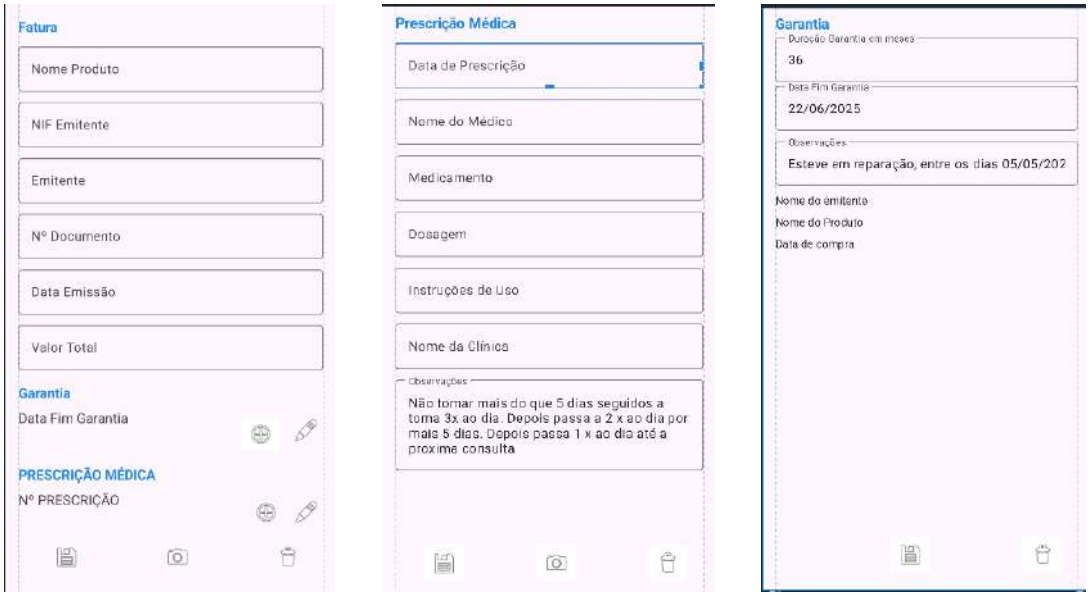


Figura 35: Ecrãs adicionar/Editar: Fatura/Garantia/Prescrição médica (Brito, 2024, p.25).

A **classe CaptureDocumentActivity** do pacote com.rfsab.invoicesafe estende AppCompatActivity e é projetada para capturar documentos utilizando a câmera do dispositivo, processar códigos QR e gerenciar dados de faturas. Abaixo, detalham-se as funções principais desta classe:

setupImageReader(): Configura um ImageReader para a captura de imagens estáticas no formato JPEG, com dimensões específicas. Este método também estabelece um listener que, ao detetar uma nova imagem, a converte em um Bitmap e a envia para o método processQRCode() para processamento.

onCreate(): Método de inicialização da atividade que define o layout, inicializa componentes da UI como o SurfaceView e o botão de captura, configura o BarcodeDetector para detetar códigos QR, e adiciona callbacks ao SurfaceHolder para gerenciar eventos da superfície de visualização.

startBackgroundThread() e stopBackgroundThread(): Controlam um thread de fundo destinado a operações que não devem interromper a UI, como a captura de imagens pela câmera.

openCamera(): Solicita acesso e abre a câmera do dispositivo de forma assíncrona, configurando um StateCallback para gerenciar eventos de estado da câmera.

createCameraPreviewSession(): Estabelece uma sessão de captura para a pré-visualização da câmera, definindo como as imagens capturadas serão processadas e exibidas na superfície fornecida pelo SurfaceHolder.

captureImage(): Inicia uma captura de imagem estática utilizando o ImageReader configurado para salvar a imagem capturada para processamento subsequente.

processQRCode(): Processa um Bitmap em busca de códigos QR com o BarcodeDetector. Caso um código QR seja encontrado, os dados são extraídos, salvos em um banco de dados e uma nova atividade é aberta para exibir os dados da fatura.

openAddInvoiceActivityWithInvoiceData() e saveInvoiceDataToDB(): Métodos utilizados para persistir os dados da fatura extraídos e para navegar para outra atividade que permite a exibição ou edição desses dados.

parseQRCode(): Analisa o conteúdo de um QR Code para extrair campos específicos usados na criação de um modelo de fatura (ModelInvoice).

onRequestPermissionsResult(): Gerencia os resultados de solicitações de permissão, como a permissão para usar a câmara, fundamental para a funcionalidade desta atividade.

Cada função é meticulosamente projetada para lidar com aspetos específicos do processo de captura de imagens e processamento de documentos, abrangendo desde a interação com o utilizador até operações de fundo para o processamento de imagens e gestão de dados.

Figura 36: Ecrã Captura de QR Code (Brito, 2024, p.19).

Os **Fragments** representam uma parte reutilizável da interface do utilizador numa aplicação Android. Utilizados para criar interfaces dinâmicas e adaptativas, os fragments ajustam-se a diferentes tamanhos de ecrã e orientações sem necessidade de recarregar toda a

atividade. Um fragment pode ser usado em múltiplas atividades, facilitando a reutilização de componentes comuns e o gerenciamento de interfaces complexas com múltiplas seções.

Os fragments possuem seu próprio ciclo de vida, embora estejam sempre incorporados numa atividade que hospeda e gerência seus estados. São especialmente úteis em dispositivos com ecrãs grandes, como tablets, onde múltiplos fragments podem ser mostrados simultaneamente.

Na InvoiceSafe, foram implementados três fragments InvoicesFragment, WarrantiesFragment e PrescriptionsFragment.

A classe InvoicesFragment é um exemplo de como um fragment pode ser utilizado para gerenciar uma seção específica de uma aplicação - neste caso, a visualização e gerenciamento de faturas. Segue uma exploração das funções desta classe:

onCreateView(LayoutInflater, ViewGroup?, Bundle?): View: Este método é responsável por criar a vista do Fragment. O layout é inflado usando um FragmentInvoicesBinding, e os métodos setupFab() e setupRecyclerView() são chamados para configurar a interface. Retorna a vista raiz inflada.

setupFab(): Configura o Floating Action Button (FAB) para adicionar novas faturas. Ao clicar no FAB, a atividade AddInvoiceActivity é iniciada, permitindo ao utilizador adicionar uma nova fatura.

setupRecyclerView(): Configura o RecyclerView, utilizado para listar as faturas. Inicializa o InvoiceAdapter com um listener de clique que abre a atividade de edição da fatura quando um item é clicado. Define o layout manager como LinearLayoutManager, adiciona decorações ao RecyclerView e atualiza a lista de faturas chamando updateInvoicesList().

editInvoice(ModelInvoice): Este método é acionado quando um item na lista é clicado. Inicia a AddInvoiceActivity, passando o ID da fatura como extra no Intent, permitindo editar a fatura existente.

updateInvoicesList(): Busca todas as faturas do banco de dados usando invoiceDAO.getAllInvoices() e atualiza o RecyclerView passando a lista de faturas para o adapter através de submitList().

onResume(): Chamado quando o fragment volta a ficar ativo. Aqui, a lista de faturas é atualizada para refletir quaisquer mudanças que possam ter ocorrido enquanto o Fragment estava pausado ou parado.

onDestroyView(): Chamado quando a vista associada ao fragment está prestes a ser destruída. Neste método, a referência ao binding é limpa (_binding = null), evitando fugas de memória.

Cada função desempenha um papel específico no gerenciamento da interface do fragment e na interação com o utilizador, facilitando a navegação, a visualização e a edição das faturas numa aplicação que utiliza uma arquitetura baseada em fragments para uma experiência de utilizador mais modular e adaptável.

Figura 37: Fragment Lista de faturas (Brito, 2024, p.18).

4.2. Interface Amigável e Intuitiva

Foi também desenvolvido um Logótipo, pois ao criar uma imagem de marca, associada ao desenvolvimento da aplicação, leva à rápida identificação dos utilizadores com as funcionalidades pretendidas.

InvoiceSafe

Figura 38 - Logótipo

Planeamento e Arquitetura: A aplicação 'InvoiceSafe' foi projetada para operar exclusivamente em dispositivos Android, com os seguintes ecrãs principais definidos para garantir uma navegação intuitiva e funcional.

Ecrãs de Interface:

- Ecrã de Login.
- Registo de perfil.
- Listagem de faturas.
- Edição de perfil, garantias e prescrições.
- Leitura de QR Codes com reconhecimento inteligente do tipo de documento.
- Funcionalidades de Gestão:
- Cada ecrã foi projetado para facilitar a interação com o utilizador e otimizar a gestão de documentos.

Modelagem e Design da Arquitetura:

- O design da arquitetura da aplicação foi modelado para apoiar uma base de dados robusta e segura, utilizando SQLite para armazenar dados encriptados. A arquitetura foi planeada para suportar múltiplas funções interativas, conforme descrito nas especificações de funcionalidades.

Desenvolvimento de Funcionalidades

Segurança:

- Autenticação: Implementação de um sistema de autenticação através de email e senha.
- Recuperação de Senha e Login Biométrico: Introdução de funcionalidades para a recuperação de senha e uma opção de autenticação biométrica, aumentando a segurança e a facilidade de acesso à aplicação.

Gestão de Dados:

- Armazenamento Local com SQLite: Utilização do SQLite para o armazenamento de dados em tabelas estruturadas, destinadas a suportar informações detalhadas sobre faturas, garantias, prescrições médicas e dados dos emitentes.

Funcionalidades Operacionais:

- Interface de Utilizador: Desenvolvimento de diversos ecrãs para o registo e gestão de faturas, garantias e prescrições. Inclusão de funcionalidades para

adicionar e editar esses itens, assim como para capturar informações das faturas através da leitura de códigos QR.

4.3. Integração de Funcionalidades e Testes

A fase de integração foi essencial para assegurar a operacionalidade coesa de todos os componentes da aplicação 'InvoiceSafe'. Utilizou-se o GitHub como ferramenta de gestão de versões, o que permitiu um controlo eficaz das alterações do código-fonte e facilitou a gestão do desenvolvimento.

Utilidade do GitHub: O uso desta plataforma de gestão de versões proporcionou um registo detalhado de todas as alterações efetuadas, preparando o projeto para uma eventual transição para trabalho em equipa, caso necessário para a evolução da aplicação.

A realização de testes rigorosos revelou-se fundamental para assegurar a funcionalidade e segurança da aplicação.

Testes Funcionais:

- **Objetivo:** Assegurar a operacionalidade integral de todas as funcionalidades, desde a gestão de finanças pessoais até à leitura de códigos QR, de acordo com os requisitos estabelecidos.
- **Metodologia:** Implementação de procedimentos de teste que verifiquem a funcionalidade dos componentes individuais e do sistema como um todo, assegurando a conformidade com os parâmetros de desempenho predefinidos.

Testes de Usabilidade:

Confirmar que a interface do utilizador é intuitiva e acessível.

- **Objetivo:** Verificar se a interface do utilizador é intuitiva e de fácil acesso.
- **Metodologia:** Realização de ensaios de usabilidade, empregando o feedback direto dos utilizadores para identificar e remediar pontos de fricção, com o propósito de aprimorar a experiência global de interação com a aplicação.

A metodologia estruturada aplicada no desenvolvimento, integração e ensaios da aplicação 'InvoiceSafe' não apenas incrementou a fiabilidade do software, mas também estabeleceu uma base sólida para futuras ampliações e aperfeiçoamentos no sistema. Este processo está alinhado com as normas académicas e as práticas profissionais recomendadas na engenharia de software.

Os resultados mostraram uma preferência significativa pelo sistema operativo Android, embora também tenha sido notada uma utilização considerável de dispositivos iOS. Este panorama sugere que um desenvolvimento cross-platform seria ideal para alcançar um público mais amplo. No entanto, considerando as limitações de recursos e know-how técnico disponíveis no momento, optou-se por concentrar os esforços no desenvolvimento para o sistema operativo Android. Esta decisão foi apoiada pela maior quota de mercado do Android no conjunto de utilizadores potenciais identificados, bem como pela familiaridade da equipa de desenvolvimento com esta plataforma.

Esta escolha estratégica permitiu alocar recursos de maneira mais eficaz, assegurando que a aplicação possa ser desenvolvida dentro das capacidades técnicas da equipe e lançada de maneira oportuna. A possibilidade de expansão para outras plataformas, como iOS, permanece uma meta a longo prazo à medida que a capacidade técnica e os recursos do projeto se expandirem.

Preferências e Preocupações Funcionais: As funcionalidades mais valorizadas incluem a segurança de dados, a integração com outros serviços digitais e a capacidade de leitura automática de códigos QR. Estes resultados destacam a prioridade que os utilizadores dão à segurança e à conveniência na gestão de informações sensíveis.

A disponibilidade para pagar por tais serviços depende largamente do preço, com muitos utilizadores não dispostos a assumir custos adicionais sem garantias de valor acrescentado significativo. Este aspeto sugere que uma estratégia de monetização baseada em modelos de subscrição flexíveis ou pagamentos por funcionalidades premium poderia ser mais atraente para o mercado alvo.

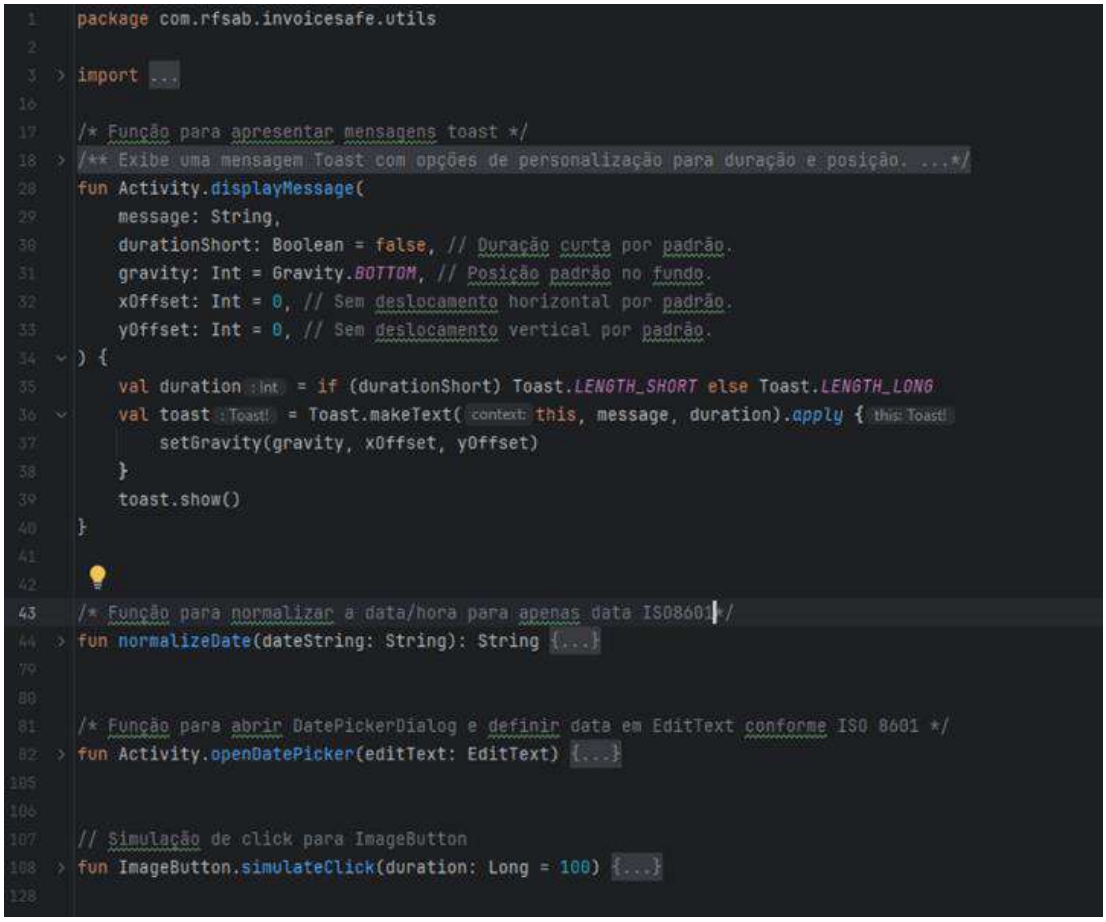
Frequência de Uso e Problemas de Conectividade: A maioria dos participantes sincroniza dados diariamente, indicando um uso frequente e dependente de conectividade robusta. No entanto, problemas de conectividade são comuns, com muitos utilizadores a relatarem velocidades de internet moderadas a rápidas. Isto sublinha a necessidade de a aplicação funcionar eficazmente sob várias condições de rede.

Preferências de Sincronização: Existe uma clara divisão entre preferências por sincronização automática de dados e a opção por controle manual, sugerindo que a personalização nas configurações de sincronização poderia ser um fator atrativo para os utilizadores. Este recurso permitiria que os utilizadores adaptassem a funcionalidade da aplicação às suas necessidades específicas de segurança e gestão de dados.

Implementação: Após os resultados do inquérito terem suscitado algumas alterações foi efetuada uma versão revista e aperfeiçoada sobre a estrutura da aplicação, reajustada conforme solicitado:

- A estrutura da aplicação foi sustentada por cinco pilares fundamentais: Experiência do **Utilizador (UX)**, **Base de Dados (BD)**, **Performance**, Automatização e **Boas Práticas**.
- **Experiência do Utilizador (UX):** Foi dada prioridade a uma interface intuitiva e responsiva, projetada para garantir facilidade de navegação e acessibilidade. Este pilar foi crucial para melhorar significativamente a interação dos utilizadores com a aplicação.
- **Base de Dados (BD):** Adotou-se uma arquitetura robusta que assegurou uma gestão eficiente dos dados. Este aspeto foi fundamental para proporcionar respostas rápidas às consultas e para manter a manutenção segura e escalável do sistema.
- **Performance:** Enfatizou-se a otimização contínua dos processos internos da aplicação para reduzir os tempos de carregamento e melhorar a reatividade, proporcionando assim uma experiência de utilizador mais ágil e satisfatória.
- **Automatização:** A automatização de tarefas reduziu o número de interações manuais necessárias, facilitando a transição dos utilizadores entre as funcionalidades. Utilizaram-se, por exemplo, QR codes para a leitura automática de dados, como faturas e prescrições médicas, agilizando o carregamento automático destes dados.
- **Boas Práticas:** Desde o início do desenvolvimento, comprometeu-se com a incorporação das melhores práticas de desenvolvimento de software. Isto incluiu a aderência ao princípio de "Não se Repita" (Don't Repeat Yourself - DRY), conforme sugerido por Sylor-Miller (2017, pp. 87-88), a implementação de padrões de design e a realização de testes frequentes para assegurar a qualidade e sustentabilidade do código ao longo do tempo.

Neste contexto, desenvolveu-se um pacote específico denominado 'utils' para agrupar funções comuns, facilitando a sua reutilização ao longo do código, conforme ilustrado na Figura 39.



```
package com.rfsab.invoicesafe.utils

import ...

/* Função para apresentar mensagens toast */
/** Exibe uma mensagem Toast com opções de personalização para duração e posição. ...*/
fun Activity.displayMessage(
    message: String,
    durationShort: Boolean = false, // Duração curta por padrão.
    gravity: Int = Gravity.BOTTOM, // Posição padrão no fundo.
    xOffset: Int = 0, // Sem deslocamento horizontal por padrão.
    yOffset: Int = 0, // Sem deslocamento vertical por padrão.
) {
    val duration : Int  = if (durationShort) Toast.LENGTH_SHORT else Toast.LENGTH_LONG
    val toast : Toast!  = Toast.makeText( context: this, message, duration).apply { this: Toast!
        setGravity(gravity, xOffset, yOffset)
    }
    toast.show()
}

/* Função para normalizar a data/hora para apenas data ISO8601 */
fun normalizeDate(dateString: String): String {...}


/* Função para abrir DatePickerDialog e definir data em EditText conforme ISO 8601 */
fun Activity.openDatePicker(editText: EditText) {...}


// Simulação de click para ImageButton
fun ImageButton.simulateClick(duration: Long = 100) {...}
```

Figura 39: Package utils (Brito, 2024, p. 10).

Inicialmente, planeava-se desenvolver de raiz o sistema de gestão de perfil e login. Contudo, após uma análise pormenorizada, optou-se pela utilização do sistema de autenticação do Firebase. Esta decisão fundamentou-se na análise de Moroney (2017, pp. 1-8) acerca do custo-benefício de empregar tecnologias de ponta, justificando assim a adoção do Firebase e permitindo que os recursos fossem inteiramente dedicados ao desenvolvimento da aplicação. Para a gestão do perfil e do processo de login, recorreu-se a várias bibliotecas: a de autenticação do Firebase, a de Storage do Firebase e a do Firestore, seguindo as orientações da documentação do Firebase.

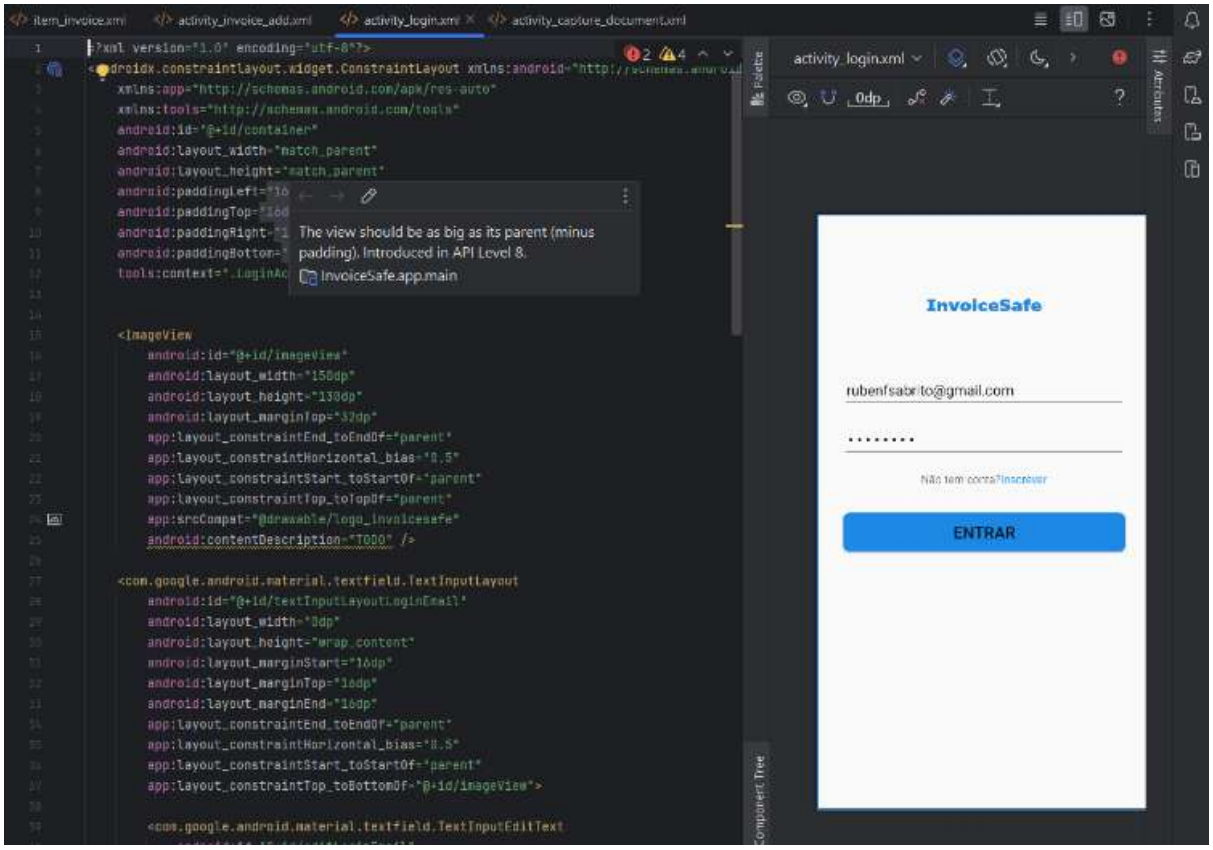


Figura 40: Ecrã Login (Brito, 2024, p. 14)

4.4. Desafios e Soluções

Um dos maiores desafios do projeto consistiu em assegurar a usabilidade. sendo fundamental proporcionar uma experiência ao utilizador que fosse natural, intuitiva e equilibrada, oferecendo um feedback claro após cada ação. A primeira refatoração implicou a eliminação de mensagens de feedback não essenciais, que foram substituídas por uma mudança subtil de cor na área de interação, simulando a pressão de um botão, bem como a remoção do excesso de botões.

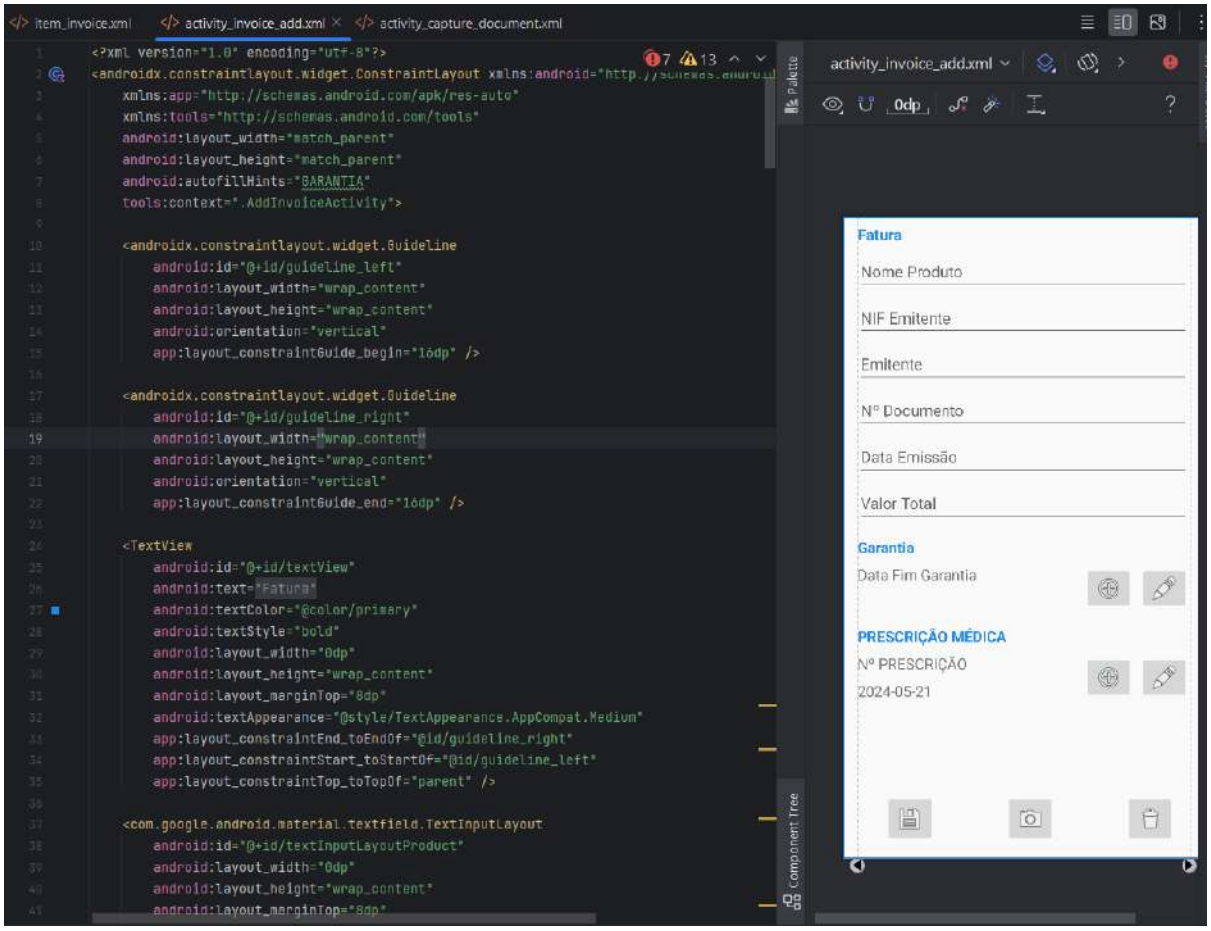


Figura 41: Ecrã adicionar fatura (Brito, 2024, p. 14)

A construção da base de dados também representou um consumo significativo de recursos. Optou-se por utilizar modelos e interfaces para separar a lógica do negócio das lógicas de interação com a base de dados, proporcionando a flexibilidade necessária para facilitar futuras mudanças na tecnologia de armazenamento de dados.

A implementação da base de dados SQLite revelou-se crucial para uma gestão eficaz e segura dos dados. A estruturação da base foi pensada para suportar a complexidade das operações requeridas pela aplicação. A classe DBHelper centralizou a gestão dos comandos SQL, assegurando eficiência e segurança nas operações, o que foi vital para a integridade e performance da base.

Interfaces como IInvoiceDAO, IWarrantyDAO e IPrescriptionDAO foram fundamentais para facilitar a interação entre o código da aplicação e a base de dados, garantindo a separação das lógicas e a facilitação de alterações tecnológicas. A classe InvoiceDAO, WarrantyDAO e PrescriptionDAO implementou as operações CRUD, e o InvoiceContract, WarrantyContract, PrescriptionContract centralizou os nomes dos parâmetros e tabelas, aumentando a manutenibilidade do código. A classe ModelInvoice, ModelWarranty e

ModelPrescription simplificou a manipulação dos dados, funcionando como uma camada de abstração entre a lógica de negócios e a base de dados.

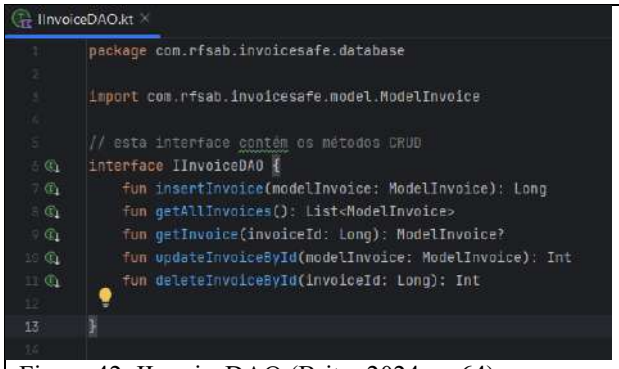


Figura 42: IInvoiceDAO (Brito, 2024, p. 64)

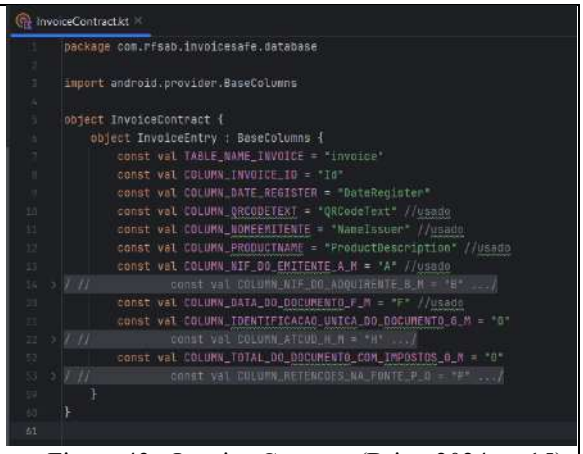


Figura 43 - Invoice Contract (Brito, 2024, p. 15)

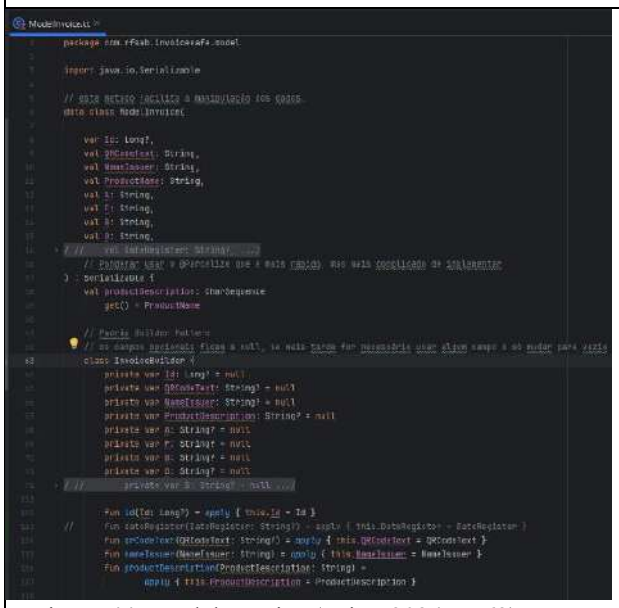


Figura 44: Model Invoice (Brito, 2024, p. 63)

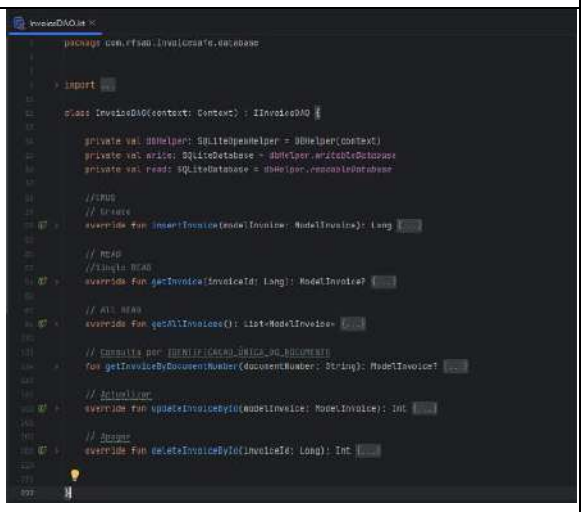


Figura 45: InvoiceDAO (Brito, 2024, p. 64)

Durante o desenvolvimento, a base de dados foi refatorada várias vezes para melhorar o desempenho e a escalabilidade. A implementação do método onUpgrade no DBHelper permitiu atualizações do esquema da base de dados com mínimo impacto nos utilizadores finais, envolvendo a remoção e recriação de tabelas de forma cuidadosa.

A funcionalidade de carregamento automático de faturas destaca-se como o elemento mais distintivo do projeto, aproveitando a tecnologia de leitura de QR Code através da câmara do smartphone.

Foi particularmente desafiante desenvolver um algoritmo que efetuasse o carregamento das faturas de forma eficaz e com alta performance, especialmente pela necessidade de detetar a existência de faturas previamente carregadas.

Não foi possível incluir todas as funcionalidades pensadas na primeira fase e ficou a faltar os seguintes pontos:

- Tirar fotos e associar ao registo da fatura selecionada;
- Efetuar pesquisas através de filtros.
- Implementar SQLCipher de modo a cifrar a BD local, na versão final de entrega
- Gerar um PDF com as faturas guardadas e respetivas garantias/ Prescrições

5. Conclusão

A aplicação InvoiceSafe que foi desenvolvida neste estudo foi concebida para integrar a gestão de finanças pessoais, documentação médica e interações com informações fiscais através da leitura de códigos QR em faturas. Implementou-se igualmente um sistema para gerir garantias de produtos adquiridos. A plataforma centralizou o armazenamento de recibos, faturas e prescrições médicas, aplicando rigorosos princípios de segurança da informação e usabilidade. A privacidade dos utilizadores foi salvaguardada através da criptografia de dados sensíveis, acessíveis apenas com autorização explícita, login de utilizador.

Um dos atributos principais da aplicação foi a sua capacidade de capturar e interpretar códigos QR em faturas comerciais, facilitando a compreensão das informações fiscais. A aplicação também proporcionou ferramentas para monitorizar e organizar garantias de produtos, melhorando a gestão e acessibilidade das mesmas.

A escolha do SQLite e a arquitetura adotada para a base de dados revelaram-se decisões acertadas, evidenciadas pela robustez e adaptabilidade do sistema face às exigências operacionais e ao aumento de volume de dados. A funcionalidade de backups e a facilidade de migração tecnológica sublinharam a adequação do SQLite para aplicações móveis em contextos dinâmicos que exigem desenvolvimento ágil.

Os testes realizados confirmaram a segurança no armazenamento de dados sensíveis, a precisão na interpretação dos códigos QR e a eficiência na gestão de garantias. A interface do utilizador foi otimizada para ser intuitiva, melhorando significativamente a experiência de utilização.

Contudo, identificaram-se áreas que necessitam de melhorias, tais como o refinamento da pesquisa para aumentar a eficiência e reduzir o impacto na base de dados, a implementação de um robusto mecanismo de backups, um fator decisivo na escolha dos serviços do Firebase, a possibilidade de captura direta de ficheiros em PDF e de imagens (JPEG, PNG ou BMP), e o desenvolvimento de uma funcionalidade de impressão para PDF, que facilite a impressão de faturas, garantias, prescrições médicas e extração de relatórios.

Foi também expresso o interesse pela funcionalidade que permite a associação de imagens aos registos, sugerindo uma valia adicional para a integração desta capacidade. Este projeto contribuiu significativamente para o campo da gestão pessoal ao desenvolver uma aplicação que facilita a administração de finanças pessoais, documentação médica e informações fiscais. A aplicação, reunindo múltiplas funcionalidades num único recurso móvel,

apresentou-se como uma solução organizada e segura de que atendeu a diversas necessidades de gestão, destacando-se como uma inovação útil no mercado de aplicações móveis.

Os resultados obtidos através do inquérito revelaram um interesse significativo por uma aplicação que ofereça funcionalidades avançadas de segurança e integração, complementadas por uma interface amigável e ajustável às necessidades de sincronização individuais dos utilizadores. Recomenda-se, portanto, que o desenvolvimento futuro da aplicação 'InvoiceSafe' considere estas preferências, enfocando a criação de uma solução segura, intuitiva e personalizável. Esta aplicação deverá operar de forma eficiente em diversas plataformas e condições de rede, assegurando assim uma experiência otimizada para todos os utilizadores. Adicionalmente, a implementação de estratégias de preços flexíveis poderia fomentar uma maior adoção e satisfação dos utilizadores, contribuindo decisivamente para o sucesso comercial da aplicação.

6. Bibliografia

Amadeo, R. (2013). *Google's iron grip on Android: Controlling open source by any means necessary*. Ars Technica. https://arstechnica.com/gadgets/2013/10/googles-iron-grip-on-android-controlling-open-source-by-any-means-necessary/

Android.stackexchange.com. (2015). *What are the names of the various versions of the Android OS, and how are these names chosen?* http://android.stackexchange.com/questions/11400/what-are-the-names-of-the-various-versions-of-the-android-os-and-how-are-these

Autoridade Tributária e Aduaneira. (2020, October). Especificações técnicas. Portaria n.º 195/2020. Versão 1.1. Info.portaldasfinancas.gov.pt. https://info.portaldasfinancas.gov.pt/pt/apoio_contribuinte/Novas_regras_faturacao/Paginas/Especificacoes_tecnicas.aspx

Bradshaw, K. (2023, January 18). *Android 13 is running on 5.2% of all devices five months after launch*. 9to5Google. https://9to5google.com/2023/01/18/android-13-device-distribution/

Brito, R. (2024). Diário de Investigação. Observação de Campo. (Não impresso). Pp. 37.

Callaham, J. (2023, 12 1). Android Authority. Retrieved from www.androidauthority.com: https://www.androidauthority.com/history-android-os-name-789433/

Dix, A., Finlay, J., Abowd, G. D., & Beale, R. (2004). *Human-computer interaction* (3rd ed.). Prentice-Hall.

Google LLC. (2023, Agosto 8). Android open source project. https://source.android.com/static/images/android_stack_720.png?hl=pt

Google LLC. (2023b, September 25). *Google Analytics*. Firebase. https://firebase.google.com/docs/analytics

Health. (2009). *Research-based web design & usability guidelines*. Pearson. https://www.usability.gov/what-and-why/guidelines.html

Moroney, L. (Ed.). (2017). *The definitive guide to Firebase: Build Android apps on Google's mobile platform*. Springer Science+Business Media New York.

Open Handset Alliance. (2007). *Industry leaders announce open platforms for mobile devices*. https://www.openhandsetalliance.com/press_110507.html

Shneiderman, B., & Plaisant, C. (2009). *Designing the user interface: Strategies for effective human-computer interaction* (5th ed.). Addison-Wesley Longman.

Simione, A. A. (2020). *Pesquisa Qualitativa e Estudos de Caso: Significado e Linhas Práticas de Orientação. Revista Estudo & Debate, 27(4).* https://doi.org/10.22410/issn.1983-036x.v27i4a2020.2580

SQLite. (2019). *About sqlite*. Sqlite.org. https://www.sqlite.org/about.html

Zetetic, LLC. (2024). SQLCipher Documentation - How-to Guides, Articles, and Documentation. Www.zetetic.net. https://www.zetetic.net/sqlcipher/documentation/

Anexo 1 - Evolução Cronológica do Android por ordem decrescente

Tabela 2 - Versões da plataforma Android (socialcompare.com, 2023).

Android 13 (Tiramisu)
Data de lançamento: 15 de agosto de 2022
Principais recursos do utilizador adicionados: Notificações precisam ser permitidas antes do primeiro envio, painel de notificação mostra número de aplicações ativas, suporte de áudio Bluetooth LE, codec de áudio LC3, MIDI 2.0 sobre suporte USB.
Principais recursos do programador adicionados: Android Runtime (ART) melhora a gestão de memória.
Nível da API: 33
Nome da Versão: Tiramisu
Android 12L (Snow Cone v2)
Data de lançamento: 7 de abril de 2022
Principais recursos do utilizador adicionados: Suporte de ecrã dobrável, ajustes da interface do utilizador para ecrãs grandes.
Nível da API: 32
Android 12 (Snow Cone)
Data de lançamento: 4 de outubro de 2021
Principais recursos do utilizador adicionados: IU mais personalizável, melhorias de segurança e privacidade, suporte ao formato de arquivo de imagem AV1 (AVIF).
Principais recursos do programador adicionados: Linguagem de design "Material You", Núcleo de computação privada do Android.
Nível da API: 31
Nome da Versão: Snow Cone
Android 11 (Red Velvet Cake)
Data de lançamento: 8 de setembro de 2020
Principais recursos do utilizador adicionados: Gravação de ecrã, melhores configurações de permissões, balões de mensagem, tethering Ethernet.
Principais recursos do programador adicionados: Suporte de aplicação 5G, API do portal cativo, conjuntos de dados partilhados, correção de erros no Wireless.
Nível da API: 30
Nome da Versão: Red Velvet Cake
Android 10 (Queen Cake)

Data de lançamento: 3 de setembro de 2019
Principais recursos do utilizador adicionados: Melhor suporte de privacidade, acesso às configurações do sistema diretamente das aplicações, novo suporte a codecs de áudio/vídeo, suporte WIFI WPA3, suporte para telefones dobráveis, tema escuro, navegação por gestos.
Principais recursos do programador adicionados: API MIDI, balões de Notificação, autenticação biométrica em aplicações.
Nível da API: 29
Nome da Versão: Queen Cake
Android 9 (Pie)
Data de lançamento: 9 de agosto de 2018
Principais recursos do utilizador adicionados: Atualizações da interface do utilizador, controle deslizante de volume atualizado, notificações de mensagens mais ricas.
Nível da API: 28
Nome da Versão: Pie
Android 8.1 (Oreo)
Data de lançamento: 5 de dezembro de 2017
Principais recursos do utilizador adicionados: Mostrar o nível da bateria em "Configurações rápidas", temas claros e escuros automáticos.
Principais recursos do programador adicionados: API de redes neurais, API de memória partilhada, ações programáticas de navegação segura.
Nível da API: 27
Nome da Versão: Oreo
Android 8.0 (Oreo)
Data de lançamento: 21 de agosto de 2017
Principais recursos do utilizador adicionados: PIP, aplicações instantâneas, sistema de notificações melhorado.
Nível da API: 26
Nome da Versão: Oreo
Android 7.1 (Nougat)
Data de lançamento: 4 de outubro de 2016
Principais recursos do utilizador adicionados: Modo de realidade virtual Daydream, luz noturna, opção para ativar o gesto de deslizar para baixo da impressão digital.

Principais recursos do programador adicionados: API's do gestor de atalhos, inserção de imagem de teclado, suporte a chamadas multi-endpoint.
Nível da API: 25
Nome da Versão: Nougat
Android 7.0 (Nougat)
Data de lançamento: 22 de agosto de 2016
Principais recursos do utilizador adicionados: Emoji Unicode 9.0, melhor multitarefa, modo de várias janelas.
Principais recursos do programador adicionados: API de modo de desempenho sustentado, API de renderização 3D Vulkan, plataforma de realidade virtual Daydream.
Nível da API: 24
Nome da Versão: Nougat
Android 6.0 (Marshmallow)
Data de lançamento: 5 de outubro de 2015
Principais recursos do utilizador adicionados: Suporte a USB tipo C, suporte para autenticação de impressão digital, melhor duração da bateria com "sono profundo".
Principais recursos do programador adicionados: Guias personalizadas do Chrome, atualização de gestão de permissões de aplicações.
Nível da API: 23
Nome da Versão: Marshmallow
Android 5.1 (Lollipop)
Data de lançamento: 9 de março de 2015
Principais recursos do utilizador adicionados: Suporte a vários cartões SIM, proteção de bloqueio em caso de perda ou roubo, chamada de voz de alta-definição.
Nível da API: 22
Nome da Versão: Lollipop
Android 5.0 (Lollipop)
Data de lançamento: 17 de outubro de 2014
Principais recursos do utilizador adicionados: Novo design (design de material), melhoria de velocidade, melhoria no consumo de bateria.
Principais recursos do programador adicionados: Várias API's novas, aplicação de rastreamento de consumo de bateria.
Nível da API: 21

Nome da Versão: Lollipop
Android 4.4 (KitKat)
Data de lançamento: 31 de outubro de 2013
Principais recursos do utilizador adicionados: Gravação de ecrã, nova IU do sistema translúcido.
Principais recursos do programador adicionados: API pública para gerir os SMS, uso de memória aprimorado, melhorias de segurança.
Nível da API: 19
Nome da Versão: KitKat
Android 4.3 (Jelly Bean)
Data de lançamento: 24 de julho de 2013
Principais recursos do utilizador adicionados: Preenchimento automático do teclado de discagem, IU da aplicação da câmara atualizada.
Principais recursos do programador adicionados: OpenGL ES 3.0 para suporte a gráficos, melhorias de registo e análise.
Nível da API: 18
Nome da Versão: Jelly Bean
Android 4.2 (Jelly Bean)
Data de lançamento: 13 de novembro de 2012
Principais recursos do utilizador adicionados: Widgets do ecrã de bloqueio, imagens em 360 graus com Photo Sphere.
Principais recursos do programador adicionados: Sincronismo vsync, buffer triplo, latência de toque reduzida.
Nível da API: 17
Nome da Versão: Jelly Bean
Android 4.1 (Jelly Bean)
Data de lançamento: 9 de julho de 2012
Principais recursos do utilizador adicionados: Google Now, pesquisa por voz, melhorias de velocidade.
Principais recursos do programador adicionados: Navegação de pilha de aplicações, classe MediaActionSound, NFC suporta grandes cargas por bluetooth.
Nível da API: 16

Nome da Versão: Jelly Bean
Android 4.0 (Ice Cream Sandwich)
Data de lançamento: 18 de outubro de 2011
Principais recursos do utilizador adicionados: Novas ações do ecrã de bloqueio, entrada de texto e verificação ortográfica aprimoradas.
Principais recursos do programador adicionados: Multimédia streaming de baixo nível, esquema de Grade, serviço de verificação ortográfica.
Nível da API: 14
Nome da Versão: Ice Cream Sandwich
Android 3.0 (Honeycomb)
Data de lançamento: 2 de fevereiro de 2011
Principais recursos do utilizador adicionados: Suporte a vários núcleos, melhor suporte para tablets, IU 3D atualizada.
Principais recursos do programador adicionados: Barra de ação contextual, fragmentos, gráficos 2D acelerados por hardware.
Nível da API: 11
Nome da Versão: Honeycomb
Android 2.3 (Gingerbread)
Data de lançamento: 6 de dezembro de 2010
Principais recursos do utilizador adicionados: IU atualizada, maior facilidade de uso do teclado, gestão de energia aprimorado.
Principais recursos do programador adicionados: Desempenho, NDK, API de efeitos de áudio.
Nível da API: 9
Nome da Versão: Gingerbread
Android 2.2 (Froyo)
Data de lançamento: 20 de maio de 2010
Principais recursos do utilizador adicionados: Melhorias na velocidade, implementação JIT, tethering USB.
Nível da API: 8
Nome da Versão: Froyo
Android 2.1 (Eclair)

Data de lançamento: 12 de janeiro de 2010
Nível da API: 7
Nome da Versão: Eclair
Android 2.0 (Eclair)
Data de lançamento: 26 de outubro de 2009
Principais recursos do utilizador adicionados: HTML, zoom digital, suporte do Microsoft Exchange.
Nível da API: 5
Nome da Versão: Eclair
Android 1.6 (Donut)
Data de lançamento: 15 de setembro de 2009
Principais recursos do utilizador adicionados: Estrutura de gesto, navegação curva a curva.
Nível da API: 4
Nome da Versão: Donut
Android 1.5 (Cupcake)
Data de lançamento: 30 de abril de 2009
Principais recursos do utilizador adicionados: Bluetooth A2DP, suporte AVRCP, teclado virtual com previsão de texto.
Nível da API: 3
Nome da Versão: Cupcake
Android 1.1 (Banana Bread)
Data de lançamento: 9 de fevereiro de 2009
Principais recursos do utilizador adicionados: Teclado numérico "Mostrar" e "Ocultar" na aplicação de chamada, capacidade de salvar anexos MMS.
Nível da API: 2
Nome da Versão: Banana Bread
Android 1.0 (Apple Pie)
Data de lançamento: 23 de setembro de 2008
Principais recursos do utilizador adicionados: Lançamento público do Android, download e atualizações via Android Market.
Nível da API: 1

Nome da Versão: Apple Pie
Android 0.9 (Beta)
Data de lançamento: 22 de agosto de 2008
Nome da Versão: Beta

Estes são os detalhes das principais versões do Android por ordem decrescente, começando com a versão 13 até à versão Beta inicial.

Anexo 2 - Especificações Técnicas do Código de Barras Bidimensional - QR Code

Tabela 3 - Especificações técnicas do código 2D (QR) (Autoridade Tributária e Aduaneira, 2023).

Campo	Tamanho Máximo	Descrição
A	9	NIF do emitente
B	30	NIF do adquirente
C	12	País do adquirente
D	2	Tipo de documento
E	1	Estado do documento
F	8	Data do documento
G	60	Identificação única do documento
H	70	ATCUD
I1	5	Espaço fiscal
I2	16	Base tributável isenta de IVA
I3	16	Base tributável de IVA à taxa reduzida
I4	16	Total de IVA à taxa reduzida
I5	16	Base tributável de IVA à taxa intermédia
I6	16	Total de IVA à taxa intermédia
I7	16	Base tributável de IVA à taxa normal
I8	16	Total de IVA à taxa normal
J1	5	Espaço fiscal
J2	16	Base tributável isenta de IVA
J3	16	Base tributável de IVA à taxa reduzida
J4	16	Total de IVA à taxa reduzida
J5	16	Base tributável de IVA à taxa intermédia
J6	16	Total de IVA à taxa intermédia
J7	16	Base tributável de IVA à taxa normal
J8	16	Total de IVA à taxa normal
K1	5	Espaço fiscal
K2	16	Base tributável isenta de IVA
K3	16	Base tributável de IVA à taxa reduzida
K4	16	Total de IVA à taxa reduzida
K5	16	Base tributável de IVA à taxa intermédia
K6	16	Total de IVA à taxa intermédia
K7	16	Base tributável de IVA à taxa normal
K8	16	Total de IVA à taxa normal
L	16	Não sujeito / não tributável em IVA / outras situações
M	16	Imposto do Selo
N	16	Total de impostos
O	16	Total do documento com impostos

P	16	Retenções na fonte
Q	4	4 carateres do Hash
R	4	Nº do certificado
S	65	Outras informações

Apêndice A – Questionário de investigação sobre Gestão Digital de Documentos.

Questionário de investigação sobre Gestão Digital de Documentos

Caro(a) participante,

Obrigado por participar neste inquérito, parte de um projeto académico do curso de Engenharia Multimédia no Instituto Superior de Tecnologias Avançadas de Lisboa. A sua participação é crucial para o sucesso deste estudo, que explora a aceitação de uma nova aplicação móvel para a gestão de documentos importantes, facilitando a organização pessoal e contribuindo para uma melhor gestão financeira e de saúde.

Este inquérito permitirá compreender melhor as preferências e necessidades do nosso público-alvo, ajudando-nos a desenvolver uma aplicação que seja tanto segura quanto intuitiva.

Por favor, responda honestamente às questões seguintes, asseguramos a confidencialidade de todas as suas respostas. Agradecemos imensamente o seu tempo e colaboração.

* Required

Parte 1: Informação Demográfica

1. Qual é a sua faixa etária? *

- ◯ Menos de 18 anos
- ◯ 18-24 anos
- ◯ 25-34 anos
- ◯ 35-44 anos
- ◯ 45-54 anos
- ◯ 55-64 anos
- ◯ 65 anos ou mais

2. Qual é o seu género? *

- ◯ Masculino
- ◯ Feminino
- ◯ Prefiro não dizer
- ◯ Other

Parte 2: Comportamento e Preferências Tecnológicas

3. Qual é o seu nível de escolaridade? *

- ◯ Ensino Básico Incompleto
- ◯ Ensino Básico Completo
- ◯ Ensino Secundário Incompleto
- ◯ Ensino Secundário Completo
- ◯ Ensino superior incompleto
- ◯ Ensino superior completo
- ◯ Pós-graduação

4. Qual é a sua ocupação atual? *

- ◯ Estudante
- ◯ Empregado(a)
- ◯ Trabalhador(a) Independente
- ◯ Desempregado(a)
- ◯ Reformado(a)
- ◯ Other

5. Com que frequência utiliza dispositivos móveis como smartphones ou tablets? *

- ◯ Nunca
- ◯ Raramente
- ◯ Mensalmente
- ◯ Semanalmente
- ◯ Diariamente
- ◯ Constantemente ao longo do dia

6. Que sistema operativo utiliza no seu smartphone ou tablet? *

- ◯ Android
- ◯ iOS (Apple)
- ◯ Windows Mobile
- ◯ Other

7. Já utiliza alguma aplicação para gerir suas finanças pessoais (ex: faturas, bancos, investimentos)? *

- ◯ Sim
- ◯ Não

8. Quais funcionalidades considera essenciais numa aplicação de gestão de documentos financeiros e médicos? (Marque todas as que se aplicarem) *

- ☐ Leitura automática de códigos QR
- ☐ Inserção manual de dados
- ☐ Associação de fotos aos documentos
- ☐ Armazenamento seguro de dados
- ☐ Pesquisa avançada de documentos
- ☐ Lembretes de pagamentos ou vencimentos de garantias
- ☐ Integração com outros serviços digitais (e-fatura, bancos, seguros)

9. Estaria disposto(a) a pagar por uma aplicação que oferecesse estes serviços? *

- ◯ Sim
- ◯ Não
- ◯ Depende do preço

10. Quais são as suas principais preocupações ao utilizar uma aplicação para gestão de documentos pessoais? *

- ◯ Segurança dos dados
- ◯ Facilidade de uso
- ◯ Custo da aplicação
- ◯ Fiabilidade do serviço
- ◯ Other

Parte 3: Uso de Dados e Conectividade

11. Com que frequência sincroniza ou atualiza dados em aplicações moveis que utiliza (como e-mail, apps de mensagens, apps financeiras, etc.)? *

- ◯ Nunca
- ◯ Raramente
- ◯ Mensalmente
- ◯ Semanalmente
- ◯ Diariamente
- ◯ Várias vezes ao dia

12. Qual o volume de dados que estima usar mensalmente no dispositivo móvel (incluindo navegação na web, uso de apps, streaming de vídeos, etc.)? *

- ◯ Menos de 500 MB
- ◯ 500 MB a 1 GB
- ◯ 1 GB a 3 GB
- ◯ 3 GB a 5 GB
- ◯ Mais de 5 GB

13. Qual a velocidade média de sua conexão à internet no dispositivo móvel? *

- ◯ Muito lenta (menos de 3 Mbps)
- ◯ Lenta (3 a 10 Mbps)
- ◯ Moderada (10 a 25 Mbps)
- ◯ Rápida (25 a 50 Mbps)
- ◯ Muito rápida (mais de 50 Mbps)

14. Com que frequência encontra problemas de conectividade ou velocidade lenta de internet ao usar aplicativos no seu smartphone ou tablet? *

- ◯ Nunca
- ◯ Raramente
- ◯ Às vezes
- ◯ Frequentemente
- ◯ Sempre

Parte 4: Feedback e Sugestões

15. Preferiria uma aplicação que sincronize os dados automaticamente sempre que alterados ou uma que permita escolher quando sincronizar? *

- ◯ Sincronização automática sempre que houver alterações
- ◯ Preferiria escolher manualmente quando sincronizar
- ◯ Indiferente

16. Tem alguma sugestão ou comentário que gostaria de adicionar sobre o tipo de aplicação que estamos desenvolvendo?

This content is neither created nor endorsed by Microsoft. The data you submit will be sent to the form owner.

Microsoft Forms

Apêndice B – Formulário de testes para a aplicação InvoiceSafe.V01

Formulário de Testes para a Aplicação "InvoiceSafe" – V01

*Required

Informações Gerais do Teste

1. Data do Teste: *

2. Versão da Aplicação: *

3. Responsável pelo Teste: *

4. Plataforma (Android/iOS): *

- Android
- iOS

1. Testes de Funcionalidade

F1. Início de Sessão e Autenticação

5. Inicia sessão com sucesso. * (1 Point)

- OK
- NOK
- N/A

6. Tenta aceder com credenciais inválidas. * (1 Point)

- OK
- NOK
- N/A

7. Utiliza a recuperação da senha. * (1 Point)

- OK
- NOK
- N/A

8. Testa o início de sessão novamente após a privação. * (1 Point)

- OK
- NOK
- N/A

F2. Gestão de Documentos

9. Adiciona uma fatura manualmente. * (1 Point)

- OK
- NOK
- N/A

10. Adiciona uma fatura automaticamente via leitura QR Code. * (1 Point)

- OK
- NOK
- N/A

11. Edita uma fatura existente. * (1 Point)

- OK
- NOK
- N/A

12. Elimina uma fatura e confirma a sua remoção. * (1 Point)

- OK
- NOK
- N/A

13. Procura faturas utilizando filtros específicos * (1 Point)

- OK
- NOK
- N/A

14. Adiciona uma Garantia manualmente. * (1 Point)

- OK
- NOK
- N/A

15. Edita uma Garantia existente. * (1 Point)

- OK
- NOK
- N/A

16. Elimina uma garantia e confirma a sua remoção. * (1 Point)

- OK
- NOK
- N/A

17. Procura garantias utilizando filtros específicos * (1 Point)

- OK
- NOK
- N/A

18. Adiciona uma prescrição médica manualmente. * (1 Point)

- OK
- NOK
- N/A

19. Edita uma prescrição médica existente. * (1 Point)

- OK
- NOK
- N/A

20. Elimina uma prescrição médica e confirma a sua remoção. * (1 Point)

- OK
- NOK
- N/A

21. Procura prescrições médicas utilizando filtros específicos * (1 Point)

- OK
- NOK
- N/A

2. Testes de Usabilidade

U1. Navegação na Aplicação

22. Avalia a intuitividade da navegação entre diferentes secções. * (1 Point)

1. Muito mau 2. Mau 3. Satisfaz 4. Bom 5. Muito bom

1	2	3	4	5

23. Confirma se os ícones e menus são facilmente compreensíveis. * (1 Point)

1. Muito mau
2. Mau
3. Satisfaz
4. Bom
5. Muito bom

1	2	3	4	5

24. Verifica a responsividade dos botões e ligações. * (1 Point)

- OK
- NOK
- N/A

U2. Clareza e Legibilidade

25. Assegura que todos os textos são legíveis * (1 Point)

- OK
- NOK
- N/A

26. Assegura que todos os textos são legíveis * (1 Point)

- OK
- NOK
- N/A

27. Verifica se as instruções são claras e concisas. * (1 Point)

- OK
- NOK
- N/A

28. Avalia a adequação dos gráficos e ícones informativos. * (1 Point)

1. Muito mau
2. Mau
3. Satisfaz
4. Bom
5. Muito bom

1	2	3	4	5

U3. Acessibilidade

29. Testa elementos de acessibilidade para utilizadores com deficiências. * (1 Point)

- OK
- NOK
- N/A

30. Verifica opções de ajuste de texto e contraste. * (1 Point)

- OK
- NOK
- N/A

31. Avalia a compatibilidade com leitores de ecrã. * (1 Point)

1. Muito mau
2. Mau
3. Satisfaz
4. Bom
5. Muito bom

1	2	3	4	5

3. Testes de Segurança

S1. Segurança dos Dados

32. Simula ataques de injeção SQL nos campos de entrada. * (1 Point)

- OK
- NOK
- N/A

33. Verifica a encriptação de dados sensíveis. * (1 Point)

- OK
- NOK
- N/A

34. Testa a segurança na transmissão de dados (use HTTPS). * (1 Point)

- OK
- NOK
- N/A

S2. Testes de Penetração

35. Realiza testes de penetração para identificar vulnerabilidades. * (1 Point)

- OK
- NOK
- N/A

36. Avalia a robustez contra ataques de negação de serviço. * (1 Point)

1. Muito mau
2. Mau
3. Satisfaz
4. Bom
5. Muito bom

1	2	3	4	5

37. Testa a resistência a ataques de força bruta. * (1 Point)

- OK
- NOK
- N/A

4. Feedback Geral

Comentários

38. Que funcionalidades adicionais sugere? * (1 Point)

39. Encontrou alguma dificuldade particular durante os testes? * (1 Point)

40. Outras observações ou recomendações. * (1 Point)

This content is neither created nor endorsed by Microsoft. The data you submit will be sent to the form owner.

Microsoft Forms

Apêndice C - Acesso ao artefacto digital:

https://drive.google.com/drive/folders/1HT7oshoiGAulJWQ3KLL_mc53YUZMU63O?usp=sharing