UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
FACULDADE DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE BACHARELADO EM SISTEMAS DE INFORMAÇÃO

CHIP IN COW: Sistema de Gestão Compartilhada no Rateio de Compras, Realtime, Integrado a Plataforma Firebase

Wanderley das Dores Ferreira Júnior

Diamantina

2018

UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
FACULDADE DE CIÊNCIAS EXATAS E TECNOLÓGICAS

CHIP IN COW: Sistema de Gestão Compartilhada no Rateio de Compras, Realtime, Integrado a Plataforma Firebase

Wanderley das Dores Ferreira Júnior

Orientador(a):

Marcelo Ferreira Rego (Orientador)

Monografia apresentada ao Programa de Bacharelado em Sistemas de Informação do Departamento de Computação da Universidade Federal dos Vales do Jequitinhonha e Mucuri - UFVJM, como pre-requisito para obtenção do grau de Bacharel em Sistemas de Informação.

Diamantina

2018

CHIP IN COW: Sistema de Gestão Compartilhada no Rateio de Compras, Realtime, Integrado a Plataforma Firebase

Wanderley das Dores Ferreira Júnior

Orientador(a):

Marcelo Ferreira Rego (Orientador)

Monografia apresentada ao Programa de Bacharelado em Sistemas de Informação do Departamento de Computação da Universidade Federal dos Vales do Jequitinhonha e Mucuri - UFVJM, como pre-requisito para obtenção do grau de Bacharel em Sistemas de Informação.

APROVADO em 06 / 08 / 2018.

Rafael Santin

Prof. MSc. Rafael Santin – UFVJM

Luis Henrique Silva Rodrigues

Prof. MSc. Luis Henrique Silva Rodrigues – UFVJM

Marcelo Ferreira Rego

Prof. MSc. Marcelo Ferreira Rego (Orientador) – UFVJM

Aos meus pais e irmãos que, com muito carinho e amor, não mediram esforços ou deixaram de acreditar, sempre me incentivando e apoiando minhas decisões.

Aos amigos que encontrei durante esta caminhada, que de alguma forma me ajudaram e tornaram este percurso muito mais divertido e agradável de ser trilhado.

AGRADECIMENTO

Agradeço primeiramente a Deus, por permitir que eu esteja realizando mais uma grande conquista e por ter me proporcionado sabedoria durante todo o percurso traçado até este momento.

Ao meu orientador, o prof. Dr. Marcelo Ferreira Rego, por ter aceito o desafio de fazer parte deste trabalho, fazendo-se presente, disponível e sempre sincero, mesmo sabendo das dificuldades a ele apresentadas.

"Tu, Senhor, guardarás em perfeita paz aquele cujo propósito está firme, porque em ti confia."

Isaías 26:3.

RESUMO

Aplicativos móveis e mobilidade são, no atual contexto de mercado de tecnologia da informação, sinônimos de agilidade, facilidade e rentabilidade para empresas. Diversos aplicativos são desenvolvidos para solucionar problemas, propor inovações ou criar demandas específicas. O gerenciamento de despesas que necessitam ser compartilhadas por determinados grupos de pessoas, apresenta-se com um problema passível de solução através de um aplicativo. O presente trabalho pretende desenvolver um aplicativo, para a plataforma Android, que atenda a esta demanda. A aplicação será composta por listas de produtos que podem ser criadas e acessadas por qualquer usuário que seja adicionado a um dado grupo. No momento em que os produtos cadastrados em uma lista são adquiridos, eles serão retirados dela e encaminhados para um relatório de aquisições, contendo o valor unitário, a quantidade adquirida e a identificação do comprador. Isso será um comprovante para posterior ressarcimento a ser efetuado pelos demais integrantes do grupo. Sendo assim, pretende-se empregar a tecnologia para facilitar e proporcionar uma melhor gestão em compras cujas despesas sejam compartilhadas.

Palavras Chave: Android, Aplicação Android, Aplicativo, Sistemas Móveis, Mobilidade, Programação Android, Plataforma Android, Desenvolvimento de software, Smartphones.

ABSTRACT

Mobile applications and mobility are, in the current market context, synonymous of agility, ease and profitability for information technology companies. Several applications are developed to solve problems, propose innovations or create specific demands. The management of expenses that need to be shared by certain groups of people is a problem which can be solved by an application. This study aims to develop an application for the Android platform that meets this demand. The application will consist of lists of products that can be created and accessed by any user who is added to a given group. Products registered in the list, when purchased, are removed and sent to an acquisition report containing then unit value, the amount purchased and the identification of the buyer. This will be a record for further compensation by the other members of the group. Thus, we intend to use technology to facilitate and provide better management of shared expenses of specific groups.

Keywords: Android, Android Application, Application, Mobile Systems, Mobility, Android Programming, Android Plataform, Software development, Smartphones.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS

SO. - Sistemas Operacionais.

E/S. - Entrada e Saída.

PDA. - Personal Digital Assistant (Assistente Pessoal Digital).

HTC. - High-Tech Computer.

OAH. - Open Handset Alliance.

PC. - Personal Computer (Computador Pessoal).

RIM. - Research in Motion (Pesquisa em Movimento).

BBM. - Balckberrry messenger (Mensageiro blackberry).

IDC. - International Data Corporation (Corporação Internacional de Dados).

SDK. - Software Development Kit (Pacote de Desenvolvimento de Software).

GUI. - Graphic User Interface (Interface Gráfica do Usuário).

CSS. - Cascading Style Sheets (Folha de Estilos em Cascata).

SGBD. - Sistema Gerenciador de Bancos de Dados.

DVM. - Dalvik Virtual Machine (Máquina Virtual Dalvik).

VM. - Virtual Machine (Máquina Virtual)

API. - Application Programming Interface (Interface de Programação de Aplicativos).

XML. - eXtended Markup Language (Linguagem de marcação expandida).

JVM. - Java Virtual Machine (Máquina Virtual Java).

JDK. - Java Development Kit (Kit de Desenvolvimento Java).

JRE. - Java Runtime Envirorment (Ambiente de Execução Java).

IDE. - Integrated Development Enviroment (Ambiente de Desenvolvimento Integrado).

AVD. - Android Virtual Device (Dispositivo Virtual Android).

APP. - Aplicativo.

SUMÁRIO

1 INTRODUÇÃO

O presente trabalho tem como foco o projeto e desenvolvimento de um software que seja capaz de auxiliar os seus usuários no gerenciamento e controle em aquisições, cujas despesas sejam compartilhadas. Esta aplicação é destinada aos sistemas de aplicativos móveis que utilizam o Sistema Operacional Android.

1.1 Apresentação

Aplicativos móveis e mobilidade são, no atual contexto de mercado, sinônimos de agilidade, facilidade e rentabilidade para empresas de tecnologia da informação. Diversos aplicativos são desenvolvidos para solucionar problemas, propor inovações ou criar demandas específicas. A visão das empresas de tecnologia sobre estas aplicações, é dada em um trecho da matéria publicada pela Sonda, onde afirma-se que:

> "De fato, muita coisa mudou no universo corporativo com a adoção dessas tecnologias, sendo a otimização da comunicação um dos principais benefícios conquistados nesse contexto. Não tem como negar: tudo ficou mais ágil, eficaz e seguro." (SONDA, 2016).

Com isso, conectividade passou a ser uma palavra muito comum e de certa forma, indispensável para a maioria das pessoas, pois, quando algo acontece, compartilhar uma informação tornou-se algo imediato, sejam notícias, e-mails, postagens dos amigos ou informações do restaurante mais próximo, tudo tem de estar acessível, aqui e agora.

Os usuários procuram cada vez mais por meios ou artifícios que possuam recursos diversos que atendam às suas necessidades na obtenção e compartilhamento de informações, com isso, os smartphones e tablets, por toda a praticidade de locomoção, manuseio e acesso em quase qualquer área, ocupam um importante espaço em um mundo onde a palavra "mobilidade" está cada vez mais conhecida. Este pensamento é reforçado pela seguinte passagem do livro de Ricardo L. Lecheta:

> "Estamos na década da mobilidade, onde smartphones e tablets farão cada vez mais parte de nosso dia a dia, e o mercado busca incessantemente por especialistas no assunto para desenvolver aplicativos comerciais e corporativos para os mais diversos setores, como varejo, saúde, economia, jogos e muito mais." (Lecheta, 2012, pág. 16).

A busca por este tipo de tecnologia, dados os cenários atual e de um passado recente, é abordada por Oehlman em uma passagem de seu livro:

> "A medida que entramos em um mundo onde os dispositivos móveis estão tornando-se o principal mecanismo para as pessoas se conectarem com a Internet, não deveríamos surpreender-nos que a habilidade em desenvolver aplicativos para dispositivos móveis está tornando-se uma busca por funcionalidades." (Oehlman and Blanc, 2012, pág. IX).

Não apenas os usuários comuns aumentam a demanda cada vez mais de tais tecnologias, mas também, o mercado em geral tem incorporado em suas rotinas de trabalho aplicações móveis para agilizar seus negócios, integrando-as com seus sistemas, visando maior agilidade e consequentemente maior lucratividade na execução e entrega de seus produtos e serviços. As empresas que liderado esta mudança de mentalidade são a Apple e a Google, ao ofertar para o mercado seus respectivos Sistemas Operacionais Móveis.

1.2 Objetivos

1.2.1 Gerais

O presente trabalho tem o objetivo de desenvolver um software para uma plataforma móvel, que seja um facilitador no rateio simples de compras, realizadas por grupos de usuários pré-cadastrados neste mesmo sistema, onde os participantes compartilhem suas despesas de modo que seja igualitária entre todos os integrantes.

1.2.2 Específicos

A fim de alcançar o objetivo principal desse trabalho, será criado um aplicativo para dispositivos móveis que seja funcional, voltado a grupos familiares, repúblicas ou grupos de organizações de eventos que compartilhem suas despesas entre todos os integrantes, este aplicativo deverá ter as seguintes funcionalidades:

- Criação e administração de diversos grupos diferentes;
- Criação de listas de aquisições interativas, entre usuários de um dado grupo, com atualização instantânea na inserção de novos produtos;
- Criação de listas com produtos a serem adquiridos, data de criação, possibilidade de encerramento;

- Gerenciamento na aquisição coletiva de produtos e divisão das despesas;
- Geração de relatórios contendo valores, quantidades adquiridas e o identificador do integrante que realizou a compra ou pagamento;
- Possibilidade de geração de relatórios mensais, de acordo com a natureza das compras;

1.3 Justificativa

Duas vertentes da tecnologia são, nos dias de hoje, os focos principais quando trata-se do futuro e presente da tecnologia, a Inteligência Artificial e as tecnologias móveis.

O cenário atual mostra que aplicativos móveis são, de certa forma, imperativos, e gerar demandas sobre tal serviço é uma oportunidade muito boa de se alcançar grande sucesso neste mercado, com bom retorno financeiro. Um panorama entre os cenários passado e recente sobre a imersão neste mercado é dada por João B. Monteiro:

> "Antes, o mercado de desenvolvimento para celulares era praticamente restrito aos fabricantes e operadoras que controlavam a criação e inclusão dos aplicativos em seus aparelhos. A liberação, por parte dos fabricantes, de um kit de desenvolvimento de software (SDK) para suas plataformas e a criação de lojas para a distribuição de aplicativos viabilizou a abertura deste mercado para praticamente qualquer empresa ou desenvolvedor, criando assim novas oportunidades de negócio." (Monteiro, 2014, pág. 01)

A ideia de desenvolver algo em torno da tecnologia Android, envolve aspectos técnicos, tais como: a linguagem de programação utilizada e sua disseminação entre os programadores, a disponibilidade de recursos de hardware e software e o acesso a informações e pseudo-códigos de estruturas específicas; e também, aspectos de alcance, visibilidade e distribuição, uma vez que, o Android detém mais de 85% dos aparelhos vendidos em todo o planeta, como pode ser visto na **Figura 3.5** do Capítulo 3 deste trabalho.

Um problema que apresenta-se como passível de uma solução utilizando esta tecnologia, é o rateio do valor total de uma compra entre integrantes de um determinado grupo, que pode apresentar-se como integrantes de uma república estudantil, organizadores de eventos ou mesmo um grupo familiar, ou de amigos, reunidos em um bar ou restaurante, cujas despesas serão compartilhadas.

A ideia de dar maior transparência aos participantes, sobre todos os itens que estão sendo adquiridos, ou consumidos, funcionando como uma espécie de comanda iterativa, torna-se uma alternativa interessante de ser explorada.

Alguns exemplos de aplicativos de conteúdo semelhantes são, o “Conta Coletiva”, cuja definição, objetivos, público alvo e funcionalidades, estão descritos no site da Evenfy (2018), e o outro, é o “Fechando a Conta”, desenvolvido pela IBlue (2015) , ambos disponíveis para download na loja Google Play.

No app “Conta Coletiva”, cada usuário pode inserir as informações de qualquer tipo de despesas que tenha, sejam em casa, viagens, escritório ou consultório que compartilham com outros, para controle coletivo.

Já o “Fechando a Conta” segue uma ideia bastante similar ao que é proposto neste trabalho, focado em um nicho de frequentadores de bares, onde cada item é vinculado aos seus consumidores.

Tendo em vista estes nichos sociais e a oportunidade de contribuir com uma solução que seja razoável, o presente trabalho pretende desenvolver um aplicativo que atenda a esta demanda.

1.4 Estrutura da Monografia

O Capítulo 1 faz uma introdução sobre o tema abordado, uma apresentação sobre a importância e as oportunidades oferecidas no mercado de desenvolvimento de aplicações para dispositivos móveis, os objetivos que devem ser alcançados ao longo do desenvolvimento do trabalho, o porquê da escolha do tema abordado, assim como a hipótese de solução levantada para uma possível resolução do problema.

No Capítulo 2, são descritas a metodologia a ser aplicada, de acordo com a finalidade do presente trabalho.

O Capítulo 3 apresenta todo o referencial teórico utilizado na construção da presente monografia, assim como alguns fatos sobre os smartphones e tablets, aspectos, informações e definições a respeito das características inerentes ao Android.

No Capítulo 4 são apresentadas as linguagens de programação utilizadas no desenvolvimento do projeto, assim como suas singularidades e também, as ferramentas necessárias para a criação de uma aplicação e como obtê-las.

No Capítulo 5 são mostrados os passos do desenvolvimento da aplicação proposta, a estrutura dessa construção, detalhes da implementação e as principais interfaces de interação com os potenciais usuários, assim como alguns testes realizados em relação à usabilidade da ferramenta e interação com seu usuário, para que o uso possa ser o mais intuitivo possível.

Para concluir, no Capítulo 6, estão todas as considerações finais, fatos que foram observados ao longo do desenvolvimento do presente trabalho e indicações de como este projeto pode ser utilizado como base de aprendizado para uma possível evolução.

Por fim, encontra-se a Bibliografia, onde estão listadas todas as fontes de conhecimento utilizadas para fornecer uma base teórica e técnica na construção desta monografia, assim como da própria aplicação.

2 METODOLOGIA

Com base nos procedimentos técnicos utilizados, a pesquisa desenvolvida neste trabalho pode ser classificada como uma pesquisa aplicada, pois, tem como meta propor uma solução para problema, e assim, contribuir de forma prática com os seus resultados. Entretanto, pode-se dizer que, para fins mais teóricos em um âmbito acadêmico, esta pesquisa pode ser apresentada como uma metodologia aplicada, onde a ideia não é testar ou confirmar uma determinada hipótese, mas sim assumir a forma de pesquisa bibliográfica e estudo de caso, permitindo ao pesquisador definir o seu problema de pesquisa e formular a sua hipótese com mais precisão. Ela também lhe permite escolher as técnicas mais adequadas para suas pesquisas e decidir sobre as questões que mais necessitam de atenção e investigação detalhada.

Baseado na leitura dos artigos de Meirinhos and Osório (2010) e Miguel (2007), podemos afirmar que, o estudo de caso deve ser aplicado quando o pesquisador tiver o interesse em pesquisar uma situação singular, particular e específico, seguindo três fases em seu desenvolvimento: a fase de exploração, a delimitação do estudo e coleta de dados e análise sistemática desses dados.

Seus resultados fornecem geralmente dados qualitativos e objetiva gerar conhecimentos para aplicações práticas dirigidos à solução de problemas específicos.“Por ser um tipo de pesquisa muito específica, quase sempre ela assume a forma de um estudo de caso”. Gil (2010).

Configura-se também como método aplicado ou de aplicação, uma vez que em sua taxonomia, apresenta sinônimos do tipo desenvolver, esboçar, estruturar e praticar, assim como uma meta de contribuir para fins práticos, uma busca por soluções para um problema concreto e transformar em ação concreta os resultados do trabalho.

2.1 Metodologia de Desenvolvimento

Um processo de software, ou metodologia de desenvolvimento de software, é um conjunto de atividades e resultados associados que auxiliam na produção do software. Dentre as várias atividades associadas, existem, por exemplo, a análise de requisitos e a

codificação. O resultado do processo é um produto que reflete a forma como o processo foi conduzido.

Segundo Ian Sommerville, existem quatro atividades fundamentais de processo que são comuns a todos os processos de software:

> 1. "Especificação do software: clientes e engenheiros definem o software a ser produzido e as restrições para a sua operação.
> 2. Desenvolvimento de software: o software é projetado e programado.
> 3. Validação de software: na qual o software é verificado para garantir que é o que o cliente deseja.
> 4. Evolução de software: o software é modificado para se adaptar às mudanças dos requisitos do cliente e do mercado." Sommerville (2008)

Em seu livro, Roger S. Pressman apresenta o seguinte conceito:

> "[...] uma metodologia de processo genérica para engenharia de software estabelece cinco atividades metodológicas: comunicação, planejamento, modelagem, construção e entrega. Além disso, um conjunto de atividades de apoio (umbrella activities) são aplicadas ao longo do processo, como o acompanhamento e controle do projeto, a administração de riscos, a garantia da qualidade, o gerenciamento das configurações, as revisões técnicas e outros." Pressman (2009).

Neste trabalho, será utilizado o processo de software definido como prototipação evolutiva, que é um processo interativo de geração de modelos de software e a atividade de desenvolvimento de uma versão inicial do sistema baseada no atendimento dos requisitos. Mais precisamente, o aplicativo desenvolvido neste trabalho será um sistema piloto, que é usado não apenas com propósitos ilustrativos, mas deve ser instalado no ambiente de aplicação e experimentado com os usuários.

A prototipagem é um processo iterativo de geração de modelos onde há o desenvolvimento de uma versão inicial do sistema baseado no atendimento dos requisitos, permitindo a descoberta de falhas.

Este processo possui marcos de início e fim bem definidos, prezando pelo dinamismo e objetividade no desenvolvimento de um software, otimizando todos os recursos disponíveis que possam vir a gerar qualquer tipo de impedimento. Entre estes marcos, há cinco etapas sequenciais a serem seguidas:

1. **Comunicação:** Consiste na coleta e refinamento dos requisitos, sejam eles funcionais ou não-funcionais, necessários para definição do escopo de desenvolvimento;

2. **Plano Rápido:** A partir dos requisitos obtidos, traçar um cronograma bem estruturado, de acordo com a montagem do escopo de desenvolvimento;

3. **Modelagem Rápida:** Definir as estruturas que compõem a aplicação, sendo estas a modelagem de classes, objetos e métodos que darão forma, possibilitarão a navegação e usabilidade dentro do sistema;

4. **Construção do Protótipo:** Codificação à partir dos modelos definidos na etapa anterior;

5. **Implantação, Entrega e Feedback:** Disponibilização do protótipo construído, em versão alpha ou beta, para experimentação e testes a um nicho definido;

Figura 2.1: Prototipagem Evolutiva

Fonte: Kolb (2013)

O modelo de ciclo de vida apresentado na **Figura 2.1** é aplicado em processos denominados “ágeis” ou “processos leves”, baseados mais no trabalho cooperativo do que no formalismo e na documentação escrita, sendo o Scrum, Extreme Programming (XP), e Feature Driven Development (FDD) alguns destes modelos.

3 REFERENCIAL TEÓRICO

3.1 Sistemas Operacionais

Um sistema operacional é uma coleção de programas que inicializam o hardware de um computador e gerenciam as funções do processador, como o *input (ou entrada), o output (ou saída)*, o armazenamento e o controle dos dispositivos.

As principais funções de um sistema operacional (SO) são:

- Gerenciamento de Processos: Execuções simultâneas ou multitarefas;
- Gerenciamento de Memória: Alocação de recursos para acesso seguro quando requisitado;
- Gerenciamento de Recursos: Políticas de uso de recursos de hardware;
- Entrada e Saída de Dados: Controla os dispositivos que recebem e fornecem informações do e para o usuário;
- Sistema de Arquivos: Armazenamento e recuperação de informações;

Em sua concepção, os sistemas operacionais foram desenvolvidos para substituir as antigas técnicas de processamento de dados onde apenas uma tarefa era executada por vez, chamado de processamento em lotes. A definição dada por Deitel em seu livro é:

> "Os sistemas de software, chamados de sistemas operacionais, foram desenvolvidos para tornar a utilização de computadores mais conveniente. Os primeiros sistemas operacionais tornaram a transição entre trabalhos mais suave e mais rápida e, portanto, aumentaram a quantidade de trabalho, ou *throughput*, que os computadores poderiam processar." (Deitel, 2005, pág. 04).

Para Tanenbaum (2009), "Um sistema operacional é o software que executa em modo núcleo [. . .]". Pode ser classificado de acordo com dois pontos de vista, dependendo do tipo de usuário e de como ele vai lidar mais com uma função ou com outra. No primeiro, o sistema operacional como uma máquina estendida, ele é uma abstração do hardware, onde são simplificados procedimentos complexos de baixo nível. No segundo, o sistema operacional como um gerenciador de recursos, o trabalho do mesmo é fornecer

uma alocação ordenada e controlada de processadores, memórias e dispositivos de E/S entre vários programas que competem por eles.

Os sistemas operacionais modernos são capazes de executar diferentes tarefas de diferentes naturezas em paralelo. Um processo ou tarefa é uma porção de um programa em alguma fase de execução. Um programa pode consistir de várias tarefas, cada uma com funcionamento próprio ou como uma unidade, talvez comunicando entre si periodicamente.

A parte do sistema operacional responsável pelo acesso aos recursos do sistema é denominada kernel. O kernel roda diretamente no hardware do processador de uma máquina, tornando possível a realização de suas tarefas.

3.2 Smartphones e Surgimento do Android

A Apple foi a pioneira ao lançar o Newton, pode-se dizer que o dispositivo é um predecessor do iPad e iPhone. O aparelho da Apple foi anunciado e vendido como um assistente pessoal digital, os já extintos PDAs.

O primeiro PDA, acrônimo de *Personal Digital Assistant*, no formato que conhecemos *(touch screen)* foi o Newton da Apple lançado em 1993, representado na **Figura 3.1**, também conhecido como Newton Message Pad, um computador de mão com tela sensível ao toque e reconhecimento de escrita.

Figura 3.1: Apple Newton PDA.

Fonte: Lee (2011)

O T-Mobile G1, apresentado na **Figura 3.2**, desenvolvido pela HTC, foi o primeiro celular lançado com a plataforma do Android e, seus primeiros exemplares começaram a ser vendidos nos Estados Unidos em outubro de 2008.

Figura 3.2: HTC - T-Mobile G1.

Fonte: Osbourne (2009)

A história do sistema Android teve início em outubro do ano de 2003 através do surgimento da empresa Android Inc., fundada por Andy Rubinera, Rich Miner, Nick Sears e Chris White. A empresa desenvolvia tecnologia totalmente independente de outras organizações.

Dois anos mais tarde, em agosto de 2005, a Google^{TM} comprou a companhia e

colocou todo seu time de desenvolvedores, liderados por Andy Rubinera, um dos fundadores da Android Inc., que passou a integrar o corpo de membros da empresa, para trabalhar em uma plataforma móvel baseada em Linux.

Figura 3.3: Primeiro iPhone.

Fonte: nsc DC (2013)

Em 2007, um evento mudou a realidade do mercado radicalmente, o lançamento, por parte da Apple, do seu primeiro iPhone, **Figura 3.3**. Além de trazer tela sensível ao toque com tecnologia Multi Touch, o aparelho conquistou as pessoas ao integrar iPod, sincronia com o iTunes e ao trazer um teclado virtual prático. No entanto, a filosofia centralizada da Apple, onde só eles podem construir dispositivos com o seu sistema operativo, abriu uma oportunidade ao Google^{TM}. Qualquer limitação que um sistema concorrente tenha é uma vantagem para um que não tenha essas mesmas limitações.

No mesmo ano, um grupo formado por empresas de setores diversos dos ramos de software, hardware e telecomunicações, sendo algumas das principais: LG, Motorola, Samsung, Sony Ericsson, Toshiba, HTC, Huawei, Sprint Nextel, China Mobile, T-Mobile, ASUS, Intel, Acer, Dell, Garmin e muitas outras; lideradas pelo Google, ficou conhecida como Open Handset Alliance (OHA), cuja missão é "Desenvolver uma plataforma para dispositivos móveis que seja completa, aberta e gratuita." Monteiro (2014)

A Open Handset Alliance é descrita por Ableson et al. (2012) da seguinte forma:

“É uma aliança de dezenas de organizações comprometidas em trazer para o mercado um telefone celular melhor e mais aberto”.

Em relação aos seus objetivos, Oehlman and Blanc (2012) diz: “O plano era que o Android fosse utilizado por fabricantes com múltiplos dispositivos como o SO do dispositivo deles, com o qual os fabricantes pudessem customizar e operacionalizar”.

Hoje no mercado existem diversos sistemas operacionais diferentes para celulares e smartphones, o que causa falta de padronização e um enorme esforço ao tentar portar aplicações e utilização de recursos entre estes diferentes modelos e marcas.

Dentre todos os sistemas operacionais voltados para dispositivos móveis, os mais conhecidos e utilizado hoje em dia são:

- **Android:** É o sistema operacional de propriedade da e mantido pela Google. Ele é compatível com quase tudo, em se tratando de multimídia.

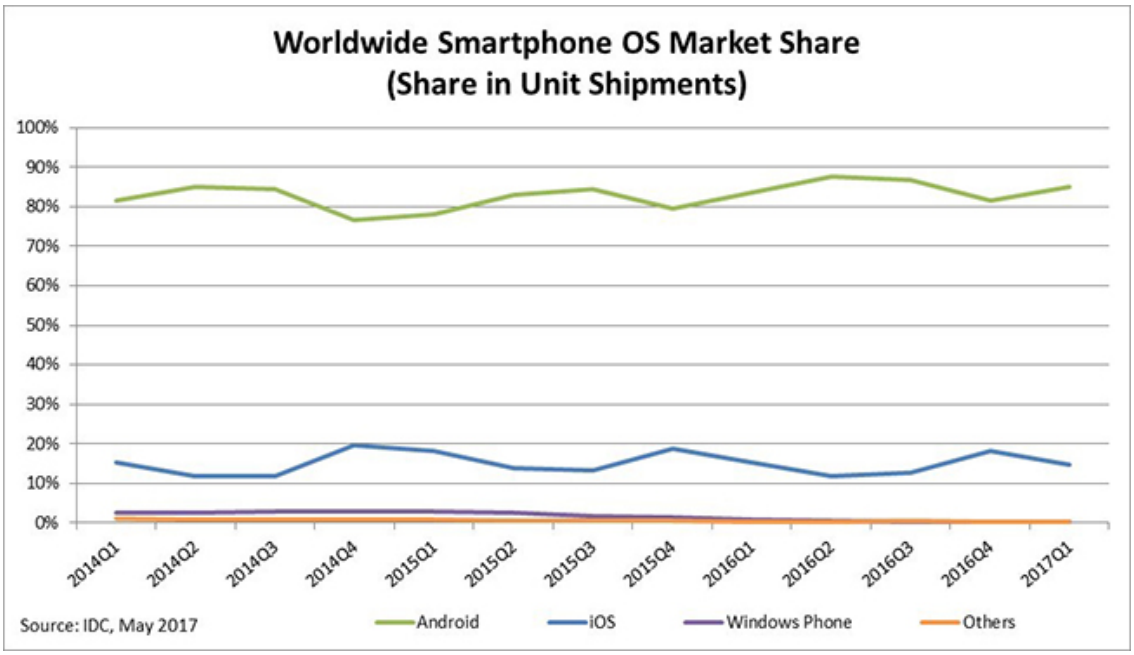
- **Windows Phone:** É um sistema operacional, para celulares, baseado no Kernel do Windows CE6. Este sistema integra e é compatível com todas as aplicações de escritório básicas da versão PC: Word, Excel, Power Point, Windows Media Player, etc.

- **iOS:** Sistema operacional que permite aos iPhones e iPads rodarem. Este foi o primeiro sistema operacional criado para celulares que oferece suporte para as tecnologias de toque múltiplas, aptas à multimídia (vídeos, imagens e músicas).

- **Symbian:** Foi concebido pela parceria de um grupo de fabricantes: Nokia, Ericsson, Motorola e Panasonic. É um sistema aberto e de baixo custo, possui recursos para gerenciar e utilizar pouca bateria e permite a instalação de software de terceiros, ao contrário do iPhone.

- **BlackBerry:** É um sistema operacional concebido pela empresa canadense RIM – Research in Motion. O que o diferencia dos demais é que ele utiliza um serviço próprio de e-mail RIM, chamado BBM (Balckberrry messenger). Ele se sobressai comparados aos demais como sistema mais adequado para uso profissional.

Figura 3.4: Participação no Mercado.



Fonte: IDC (2017)

Segundo o relatório da International Data Corporation (IDC) publicado no quarto trimestre de 2017, apesar da queda em relação à quantidade de unidades de smartphones vendidos, o Android possui uma quota de 85.0% do mercado de smartphones. O iOS aparece em segundo lugar com uma quota de 14.7% do mercado. A **Figura 3.4** acima, mostra o gráfico desde o primeiro trimestre de 2014 até o mesmo período do ano de 2017.

Os números correspondentes ao período equivalente ao ano de 2016, separado trimestralmente, e o primeiro quarto de 2017, podem ser observados através da **Figura 3.5** a seguir:

3.3 Versões do Android

As versões do Android possuem uma numeração e um nome correspondente ao seu código, sendo o primeiro, a versao de lançamento presente no aparelho da HTC, o T-Mobile G1.

- **Android 1.0 (Astro):** O Android Astro fez sua estreia no HTC G1, primeiro gadget a trazer a plataforma ao mercado consumidor norte-americano pela operadora T-

Figura 3.5: Porcentagem de Aparelhos.

Period	Android	iOS	Windows Phone	Others
2016Q1	83.4%	15.4%	0.8%	0.4%
2016Q2	87.6%	11.7%	0.4%	0.3%
2016Q3	86.8%	12.5%	0.3%	0.4%
2016Q4	81.4%	18.2%	0.2%	0.2%
2017Q1	85.0%	14.7%	0.1%	0.1%

Source: IDC, May 2017

Fonte: IDC (2017)

Mobile. A sua data de disponibilidade para o público é portanto coincidente com a comercialização efetiva do dispositivo.

- **Android 1.1 (Bender):** Esta versão foi lançada como uma atualização para o G1 e possuía suporte a diversas funcionalidades, como alarmes, calculadora, câmera, contatos, mensagem, música, entre outras configurações básicas.

A partir desta versão os nomes de código das versões de Android passaram a ser baseados em doces de culinária para além da primeira letra seguir a ordem crescente do alfabeto.

- **Android 1.5 (Cupcake):** Lançada em abril de 2009, a segunda versão do Android foi a primeira disponível comercialmente em grande escala. A atualização trouxe correções de "bugs" e muitas funcionalidades ao sistema do Google, além disso, o Cupcake chegou com o primeiro telefone touchscreen com a plataforma, o HTC Magic.

- **Android 1.6 (Donut):** A maior novidade foi a introdução do Android Market. Manteve o padrão visual e começou a oferecer suporte a telas de várias resoluções, pois os elementos da interface passaram a ser redimensionados automaticamente.

- **Android 2.0 e 2.1 (Eclair):** Com papéis de parede animados, interface repaginada e novas funcionalidades, o Eclair foi lançado em outubro de 2009 e atualizado em janeiro de 2010. Nesta versão, surgiu o suporte a sincronização de contas, o que permitiu aos usuários adicionarem múltiplas contas em um dispositivo para sincronização de e-mails e contatos.

- **Android 2.2 (Froyo):** Lançado em maio de 2010, foi responsável por acelerar consideravelmente o desempenho de smartphones equipados com o sistema. Com ele, os usuários também puderam rodar conteúdo Flash, além do compartilhamento da conexão à Internet via USB e chamadas por voz.

- **Android 2.3 (Gingerbread):** Lançado em dezembro de 2010, esta foi a mais popular versão do sistema por muito tempo. As versões 2.3.1, 2.3.2 e 2.3.5, 2.3.6 e 2.3.7 são tidas como revisões, e portanto não obtiveram uma versão de API diferente ou alterações de SDK relevantes.

- **Android 3.0, 3.1 e 3.2 (Honeycomb):** Lançado em fevereiro de 2011, esta versão de Android é exclusiva a Tablets.

- **Android 4.0 (Ice Cream Sandwich):** Lançado em outubro de 2011 e atualizado em março de 2012, trouxe a tão aguardada união de tablets com smartphones em Android, constituindo assim um patamar importante na uniformização e redução de fragmentação da plataforma.

- **Android 4.1, 4.2 e 4.3 (Jelly Bean):** Tem três edições, 4.1, 4.2 e 4.3, com opções de acessibilidades melhoradas, reorganização automática de atalhos e widgets.

- **Android 4.4 (KitKat):** Pela primeira vez com um nome de uma marca e não de um doce em si, o Android KitKat foi anunciado em setembro de 2013 pelo Google e pela Nestlé. Possui refinamentos visuais e foi otimizado para funcionar melhor com smartphones de baixo custo. Possui melhor suporte a serviços de armazenamento na nuvem de terceiros.

- **Android 5.0 (Lollipop):** Lançado em 25 de junho de 2014, durante o Google IO. Uma das mudanças mais importantes no lançamento do Lollipop é uma interface

de usuário redesenhada construída em torno de uma linguagem de design conhecido como “material design”, proporcionando uma melhor fluidez e tornando a experiência mais agradável.

- **Android 6.0 (Marshmallow):** No segundo semestre de 2015. Dessa vez com menos mudanças na aparência, a nova versão 6.0 focou em oferecer mais segurança, com permissões de privacidade para apps, e maior desempenho de bateria.

- **Android 7.0 (Nougat):** Além da melhora de desempenho recorrente a cada versão, o Nougat traz recursos como a compatibilidade com várias janelas e respostas diretas para aplicativos de comunicação.

- **Android 8.0 (Oreo):** É a versão mais atual do sistema operacional da plataforma. Com ele, vem uma enorme lista de novidades, que podem ser vistas no site da Android Developer.

O Google tem investido em desenvolver sistemas que deem suporte para aparelhos de baixo custo possibilitando assim o acesso de outras classes a estes dispositivos, explorando outro nicho de mercado, principalmente para mercados de países emergentes.

O uso de dois chips no mesmo aparelho é bastante comum nesses países, mas a implementação desse recurso é, na maior parte das etapas, fruto de esforços de desenvolvimento dos fabricantes. Em sua última versão o Google passou a dar suporte a tal funcionalidade.

Na **Figura 3.6** abaixo, podemos verificar os dados sobre o número relativo de dispositivos que executam uma determinada versão da plataforma Android.

Figura 3.6: Versões da Plataforma e Porcentagem relativa de aparelhos.

Version	Codename	API	Distribution
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.4%
4.1.x	Jelly Bean	16	1.5%
4.2.x		17	2.2%
4.3		18	0.6%
4.4	KitKat	19	10.3%
5.0	Lollipop	21	4.8%
5.1		22	17.6%
6.0	Marshmallow	23	25.5%
7.0	Nougat	24	22.9%
7.1		25	8.2%
8.0	Oreo	26	4.9%
8.1		27	0.8%

Fonte: android Developer (2018b)

Podemos observar pela tabela que, a versão Nougat, das API's 24 e 25, é a mais popular entre os usuários do Android, tendo uma distribuição de aproximadamente 31,1% de todos os aparelhos no mercado. Isto se deve ao fato de que é a plataforma moderna com melhor suporte à smartphones de baixo custo e maior fluidez em sua manipulação. Suas antecessoras Marshmallow e Lollipop aparecem logo na sequência como segunda e terceira versões mais utilizadas.

3.4 Plataforma Android

O sistema operacional Android é uma plataforma de software, projetado pela Google^{TM}, como um SO genérico para dispositivos móveis, como smartphones e tablets.

A definição por Lecheta (2013) é dada de forma: "Consiste em uma nova plataforma de desenvolvimento para aplicativos móveis, baseada em um sistema operacional Linux, com diversas aplicações já instaladas e, ainda, um ambiente de desenvolvimento bastante poderoso, ousado e flexível".

Uma outra definição é dada por Pereira and da Silva (2009): "O Android é uma

plataforma para tecnologia móvel completa envolvendo um pacote com programas para celulares, já com um sistema operacional, middleware, aplicativos e interface do usuário".

No campo da computação distribuída, middleware é um programa de computador que faz a mediação entre software e demais aplicações. É utilizado para mover ou transportar informações e dados entre programas de diferentes protocolos de comunicação, plataformas e dependências do sistema operacional.

Outros autores definem o sistema Android de forma simples, como Franz (2012) que diz: "Android é um sistema operacional baseado no núcleo Linux para dispositivos móveis [...]", e também algumas outras diversas definições tais como: O Android é um software open source criado para os celulares e outros dispositivos, ou ainda: O Android é uma plataforma de software que permite criar aplicativos para dispositivos móveis, como smartphones e tablets.

3.4.1 Código Aberto

O Android começa a ganhar forma a partir do esforço em conjunto da Open Handset Alliance, com um diferencial quando comparado ao iOS da Apple, por ser completamente livre e de código aberto (open source), pode ser sempre adaptado a fim de incorporar novas tecnologias, conforme estas forem surgindo, o que, de acordo com Lecheta (2013) "Representa uma grande vantagem para sua evolução, uma vez que diversos programadores do mundo poderão contribuir para melhorar a plataforma."

O Android é código aberto e distribuído sob licença Apache 2.0, o que quer dizer que você tem acesso aos códigos-fonte e também pode contribuir com o projeto.

Esse fato permite que, tanto aos fabricantes de aparelhos quanto aos programadores independentes, possam realizar alterações no código-fonte ou criar suas próprias aplicações, sem a necessidade de compartilhar tais alterações. O fato é que, isso contribuiu muito para seu desenvolvimento e aperfeiçoamento, uma vez que a facilidade de acesso gerou grande interesse e comunidades de desenvolvedores por todo o mundo adicionando novas funcionalidades ou simplesmente corrigindo falhas.

Acompanhado de todos os pontos até aqui apresentados, o Google libera, de forma

gratuita, um kit de desenvolvimento de software, denominado Android SDK, aos seus "colaboradores". Este fato foi observado por Monteiro, onde diz:

> "A liberação, por parte dos fabricantes de um kit de desenvolvimento de software (SDK) para suas plataformas e a criação de lojas para distribuição de aplicativos viabilizou a abertura deste mercado para praticamente qualquer empresa ou desenvolvedor, criando assim novas oportunidades de negócios." (Monteiro, 2014, pág. 01).

"O Android SDK é o kit de desenvolvimento que disponibiliza as ferramentas e API's necessárias para desenvolver aplicações para a plataforma Android, utilizando a linguagem Java". Pereira and da Silva (2009).

3.5 Arquitetura

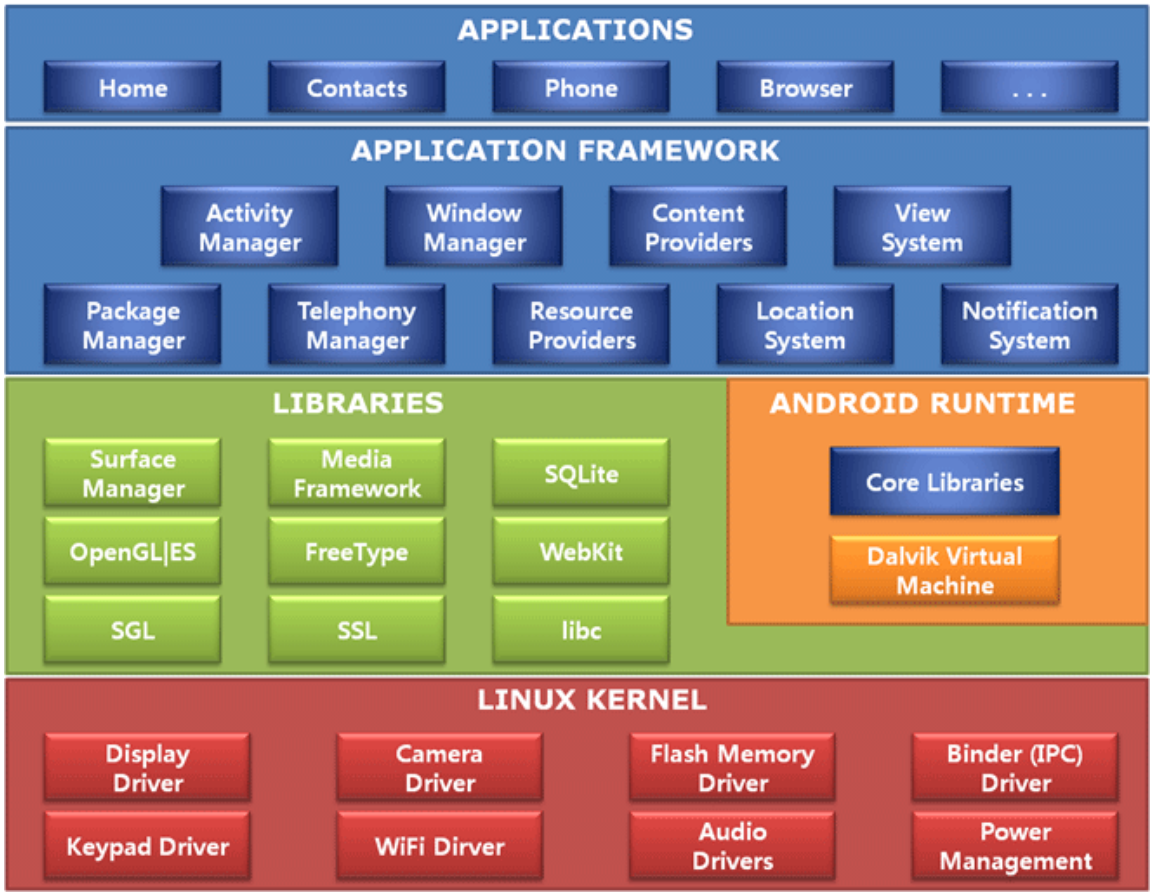
O Google geralmente se refere ao sistema operacional Android como uma pilha de softwares. Cada camada da pilha agrupa vários programas que suportam funções específicas do sistema operacional. Podemos dividir essas camadas em níveis zero, um, dois e três, conforme a **Figura 3.7** abaixo:

3.5.1 Nível 0(Zero) - Linux Kernel

Utiliza a versão 2.6 do kernel do Linux para os serviços centrais do sistema. De acordo com Lecheta (2013), "O sistema operacional do Android foi baseado no kernel 2.6 do Linux, e é responsável por gerenciar a memória, os processos, threads e a segurança dos arquivos e pastas, além de redes e drivers". Pereira and da Silva (2009) complementa ao afirmar que "Kernel também atua como uma camada de abstração entre o hardware e o resto da pilha de software". Toda a segurança do Android é baseada na segurança do Linux.

Apesar de ter sido construído com base no Linux, o Android não pode ser considerado como uma distribuição Linux, por não possuir um sistema de componentes gráficos, ou sistemas de janelas, nativos (componentes GUI – interface gráfica do usuário), não possui suporte à biblioteca glibc, em linguagem C de programação (C Library), característica em qualquer sistema operacional Unix e, alguns dos conjuntos de padrões apresentados em grande parte das distribuições Linux.

Figura 3.7: Arquitetura da Plataforma Android.



Fonte: Sampaio (2013)

3.5.2 Nível 1(Um) - Linux Kernel

No nível um, temos as camadas de bibliotecas (Libraries) e tempo de execução (Android RunTime).

A camada de biblioteca é um conjunto de instruções que dizem ao dispositivo como lidar com diferentes tipos de dados, incluindo um conjunto de biblioteca C/C++ usadas por diversos componentes do sistema e são expostas a desenvolvedores através da estrutura de aplicativo Android.

Dentre as principais bibliotecas podemos enumerar:

- **FreeType:** Para renderização de fontes, bitmaps e vector;
- **System C Library:** Consiste em uma implementação derivada da biblioteca C

padrão baseado no sistema (libc) do BSD sintonizada para dispositivos rodando Linux. Inclusive a biblioteca padrão C Bionic, que é uma biblioteca especialmente desenvolvida para o Android;

- **Webkit:** Baseado no código aberto do browser webkit, que é um renderizador de páginas para navegadores, com suporte para CSS, JavaScript, DOOM e AJAX;
- **SQLite:** Consiste no sistema gerenciador de banco de dados (SGBD) relacional disponível para todas as aplicações;
- **SGL:** Responsável pelos gráficos 2D adjacentes;
- **Surface Manager:** Responsável pelo acesso ao subsistema de exibição bem como as múltiplas camadas de aplicações 2D e 3D;
- **Media Libraries:** São as bibliotecas que suportam os mais diversos formatos de áudio e vídeo, incluindo também imagens.
- **LibWebCore:** Consiste em um web browser engine utilizado tanto no Android Browser quanto para exibições web.
- **3D Libraries:** As bibliotecas utilizam aceleração 3D via hardware (quando disponível) ou o software de renderização 3D altamente otimizado incluído no Android.

No mesmo nível da camada de bibliotecas, a camada de tempo de execução do Android (Android Runtime) é uma aplicação de software que comporta como um dispositivo independente, que inclui um conjunto de bibliotecas do núcleo Java. Nesta camada encontraremos a Máquina Virtual Dalvik (DVM). Para Pereira and da Silva (2009) o Android Runtime "[...] é uma instância da máquina virtual Dalvik, criada para cada aplicação executada no Android".

Uma máquina virtual é uma aplicação de software que age como um dispositivo independente com seu próprio sistema operacional. A Dalvik Virtual Machine é a máquina virtual desenhada e otimizada especialmente para dispositivos móveis. Esta máquina virtual é a responsável pela execução de todo o código Java que faz parte da plataforma Android.

O Android usa a máquinas virtuais Dalvik para rodar cada aplicação com seu próprio processo. Os aplicativos instalados no Android são interpretados pela máquina virtual Dalvik, e então as informações deles são enviadas até a interface gráfica. Ela funciona como um gerenciador de múltiplas máquinas virtuais, cada uma rodando um processo. O Dalvik foi escrito de forma a executar várias VM's eficientemente, isso faz com que nenhuma aplicação seja dependente de outra, o que simplifica o gerenciamento de memória, podendo ambas estarem em execução ao mesmo tempo.

A máquina virtual Dalvik foi baseada em registradores e desenvolvida de forma otimizada para requerer pouca memória e permitir que múltiplas instâncias executem ao mesmo tempo. No artigo publicado por Diogo Júnior, ele a distingue das demais VM ao afirmar que:

> "É também bastante diferente das Java VMs que são usadas normalmente noutros sistemas operativos pois esta está especialmente otimizada para que os programas possam ser interpretados duma forma rápida e eficaz e ao mesmo tempo usando pouca memória". (Júnior, 2010, pág. 06).

Em seu livro, Monteiro descreve da seguinte forma:

> "No sistema operacional Android, para cada aplicação é atribuído um usuário único de sistema e apenas este usuário recebe permissões para acessar seus arquivos. Além disso, por padrão, cada aplicação é executada em um processo próprio, que possui também sua própria instância da máquina virtual Dalvik". (Monteiro, 2014, pág. 34).

Recentemente, a Google™ está testando uma nova máquina virtual, a ART, visando que a Dalvik pode ser um problema para aplicativos que exijam maior poder de processamento e recursos utilizados.

Em vez de trabalhar com o sistema "Just-in-Time" do Dalvik, é realizada uma compilação "Ahead-of-Time", que pode ser traduzida como "à frente do tempo". Os códigos são pré-compilados na linguagem de execução já durante a instalação dos aplicativos. Isso implica em uma demanda maior de tempo na instalação dos softwares, porém, maior rapidez na execução, justamente pelo fato de a virtualização já ter acontecido e ter sido gravada na memória dos dispositivos.

3.5.3 Nível 2(Dois - Framework de Aplicação

No nível dois, temos a camada de framework de aplicação (Application Framework). Isso inclui programas que gerenciam as aplicações básicas do telefone, como alocação de recursos, mudanças entre processos ou programas. Os desenvolvedores têm acesso total ao framework como um conjunto de ferramentas básicas com o qual poderá construir ferramentas mais complexas.

Um Framework é um conjunto de módulos integrados que visam o reaproveitamento de código, aumentando a produtividade na hora do desenvolvimento. A ideia principal de um framework é proporcionar a uma determinada linguagem uma série de recursos para elevar o nível de abstração dessa linguagem.

3.5.4 Nível 3(Três) - Camada de Aplicações

No último nível estão as aplicações em si, ou seja, as funcionalidades ou funções básicas do dispositivo, as quais o usuário comum tem acesso, como por exemplo, fazer chamadas, acessar o navegador Web ou acessar seus contatos. Esta é a camada de interação entre o usuário e o dispositivo móvel e você faz isso com a interface de usuário. Apenas os programadores do Google^{TM}, os desenvolvedores de aplicação e os fabricantes de hardware acessam outras camadas mais baixas da pilha.

3.6 Componentes de uma Aplicação

As componentes de uma aplicação são os blocos diferentes de construção, sendo que através de cada um destes o sistema consegue interagir com sua aplicação. Não são todos eles que têm um caráter de interação direta do utilizador. Uma aplicação pode ser constituída por vários destes blocos, e estes blocos são comunicáveis entre si na maioria dos casos.

Uma aplicação sempre será formada por, pelo menos, um desses componentes e não necessariamente por todos eles. Em complemento a essa ideia, Monteiro (2014) faz seguinte afirmação: “Não é necessário que uma aplicação Android tenha todos estes

componentes mas e importante conhecê-los para que, no momento de projetar a aplicação, possamos selecionar o componente adequado para atender as necessidades".

A seguir estão definidos os quatro tipos de componentes, cada qual com um propósito e ciclo de vida bem definidos.

3.6.1 Activity

Uma activity representa uma tela com interface gráfica capaz de promover algum tipo de interação com o usuário, o análogo a uma Janela no Windows, onde se produz e elabora a componente de interação visual de uma aplicação. Uma aplicação Android pode ser composta de diversas para fornecer um conjunto de funcionalidades para o usuário.

É o mais utilizado de todos os componentes. Possui uma interface de usuário composta por views, consistindo de várias telas que respondem a eventos previamente programados.

3.6.2 Services

Os serviços são componentes que não contem um front-end, ou seja, um elemento visual como a activity, sendo estes Services códigos sem interface de usuário, executados em segundo plano.

Seu objetivo principal é realizar tarefas potencialmente demoradas, em plano de fundo, sem comprometer a interação do usuário com alguma activity. Os Services mantêm o serviço ativo até que seja recebida outra ordem. Tocar uma música ou fazer o download de um arquivo são exemplos de funcionalidades que podem ser implementadas utilizando um Service.

3.6.3 Broadcast Receivers

Os provedores de conteúdo são componentes que permitem o acesso e modificação de dados armazenados em um banco de dados SQLite local, ou mesmo dados armazenados na web.

Quando da necessidade de compartilhamento dos dados de sua aplicação, os provedores de conteúdo fornecerão, de forma abstrata, informações da nossa aplicação a outras aplicações instaladas no sistema, que é basicamente a forma de compartilhar dados entre os aplicativos.

Um exemplo de uma aplicação com uma componente deste gênero é a aplicação Contatos, ou lista de contatos, presente na generalidade de terminais Android.

3.6.4 Content Provides

São componentes capazes de responder a eventos propagados pelo sistema operacional ou, quando uma aplicação recebe um evento/sinal externo através de uma intenção, é este componente que trata a reação deste evento.

Podem ser vistos, de forma genérica, como os sinais de alerta que o próprio sistema nos oferece para um determinado serviço, como por exemplo o nível baixo da bateria, o recebimento de uma nova mensagem de texto, o toque de uma chamada de voz, alteração do estado da bateria, alteração das condições de conectividade.

São bastante úteis, por exemplo, no caso de uma aplicação que necessite de atualizações constantes, dependente da conectividade de rede. Uma função que pause a atualização no caso de perda da conexão sem corromper os dados o cancelar o download, faz uma gestão inteligente de quando é que a aplicação vai executar código relativo à tentativa de obtenção de atualizações periódicas.

3.7 Intents

As intents do Android são mecanismos estruturados responsáveis por passar informações, como se fossem mensagens, de forma assíncrona, para os principais componentes da API do Android, como as Activities, Services e BroadCast Receivers, que permitem que os componentes de um aplicativo solicitem a funcionalidade de outros componentes do Android.

As Intents permitem que você interaja com componentes do mesmo aplicativo, bem como com componentes contribuídos por outras aplicações. Quando usado em

conjunto com Intent Filters podemos até iniciar uma Activity de outros aplicativos, ou o inverso, deixar que um outro aplicativo inicie uma das nossas Activities. Este é um recurso-chave no Android, pois, é através dele que podemos fazer com que as aplicações colaborem entre si, disponibilizando funcionalidades que podem ser reutilizadas, sem a necessidade de importar códigos ou dependências para dentro da sua aplicação.

Embora os intents facilitem a comunicação entre componentes de diversos modos, há três casos de uso fundamentais:

- Para iniciar uma atividade (Activity);
- Para iniciar um serviço (Service);
- Para fornecer uma transmissão (Content Provides);

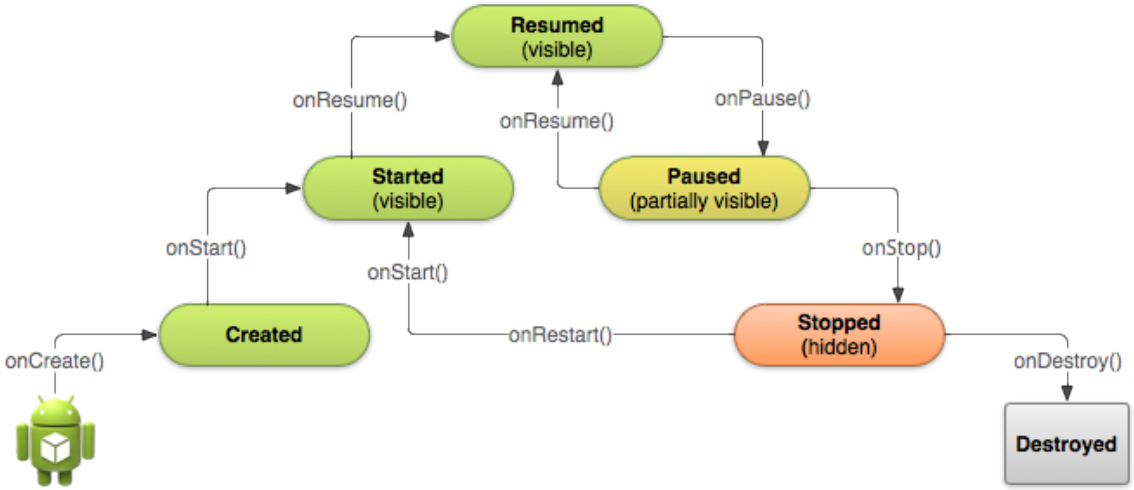
3.8 Ciclo de Vida de Uma Atividade

O Android é um sistema operativo multitarefa, e nesse sentido, desde o início do seu desenvolvimento consta de uma gestão inteligente e eficaz de processos. Por essa característica, uma aplicação pode ser interrompida de forma inesperada ou porque o usuário abriu uma outra activity. Assim que uma atividade é iniciada, esta é colocada no topo da "pilha de atividades" e se torna a atividade em execução.

Diferente de outros paradigmas de programação em que os aplicativos são lançados com um método main(), o sistema Android inicia o código em uma instância Activity. As activities contam com um conjunto de métodos de callback. Esses métodos são invocados no início de uma activity e identificam, a mudança na orientação do dispositivo, a recepção de uma chamada, mudança para uma outra aplicação, dentre outros cenários que podem surgir. Os "callback methods" são chamados em uma certa sequência criando um ciclo que é denominado o ciclo de vida de uma activity.

Conforme o usuário navega, sai e retorna para o seu aplicativo, as instâncias activity no aplicativo transitam entre diferentes estados no ciclo de vida. Durante a vida de uma atividade, o sistema chama um núcleo principal de métodos do ciclo de vida em uma sequência parecida com uma pirâmide em degraus. Na **Figura 3.8**, podemos identificar todos os métodos que fazem parte do ciclo de vida de uma activity.

Figura 3.8: Ciclo de Vida de uma Activity.



Fonte: android Developer (2018a)

O topo da pirâmide é o ponto em que a atividade funciona em primeiro plano e o usuário pode interagir com ela.

É importante saber como e quando é invocado cada um deles de forma que possamos sempre prever algumas situações e tomar as medidas necessárias para garantir a continuidade e uma boa experiência ao utilizador. Cada método possui suas características e funcionalidades:

- **onCreate():** É a primeira função a ser executada, obrigatoriamente, e uma única vez. Geralmente é o responsável por carregar os layouts e outras operações de inicialização.
- **onStart():** Chamado imediatamente após a criação da atividade ou reiniciando-a após ter sido parada, voltando a ter foco.
- **onResume():** É chamado na criação e toda vez que uma atividade volta a ter foco.
- **onPause():** Chamado quando uma atividade é colocada em background, ou segundo plano, salvando o estado da aplicação e liberando algum recurso que seja necessário à nova atividade.
- **onStop():** Invocado quando encerra-se uma atividade ou muda de aplicação e esta deixa de estar visível.

- **onRestart():** Quando uma atividade volta a ter foco após estar em background, este método é chamado. Ele faz a chamada de execução do método onStart() instanciando a atctivity em questão.

- **onDestroy():** A última função a ser executada, finaliza a aplicação liberando a memória. Pode ser invocado também pelo sistema operacional, caso ele deseje desalocar recursos.

A implementação adequada dos métodos do ciclo de vida da atividade garante que seu aplicativo tenha um bom desempenho em vários sentidos, possibilitando manter uma experiência boa para o utilizador de forma contínua, independentemente da mudança de configurações ou outro incidente que faça com que a activity seja recriada.

4 FERRAMENTAS UTILIZADAS

Para o desenvolvimento de uma aplicação, seja ela destinada à web, sistemas desktop ou dispositivos móveis, são necessárias algumas ferramentas específicas para cada uma destas áreas. Da mesma forma, o programador deve possuir conhecimento sobre qual ou quais linguagens serão utilizadas para a construção da aplicação.

Para um aplicativo desenvolvido para ser utilizado na plataforma Android, cada linguagem e as ferramentas utilizadas são descritas a seguir.

4.1 Linguagens de Programação

Uma linguagem de programação é um método padronizado para comunicar instruções para um computador, ou seja, assim como nas linguagens naturais, existe um conjunto de regras sintáticas e semânticas usadas para definir um programa de computador.

O desenvolvimento de aplicativos para Android é feito utilizando a linguagem Java, com a utilização de arquivos XML para a criação das interfaces. Apesar de parecer complexo, é relativamente simples criar seus aplicativos. Além disso, é bem fácil ter acesso a diversos recursos geralmente disponíveis em dispositivos móveis, tais como câmera, GPS, Bluetooth, etc.

4.1.1 Java

A linguagem Java é considerada simples porque permite o desenvolvimento de sistemas em diferentes sistemas operacionais e arquiteturas de hardware, sem que o programador tenha que se preocupar com detalhes de infraestrutura.

"O Java é uma poderosa linguagem de programação de computador divertida para iniciantes e adequada para programadores experientes utilizarem na construção de sistemas de informações importantes." Deitel (2005).

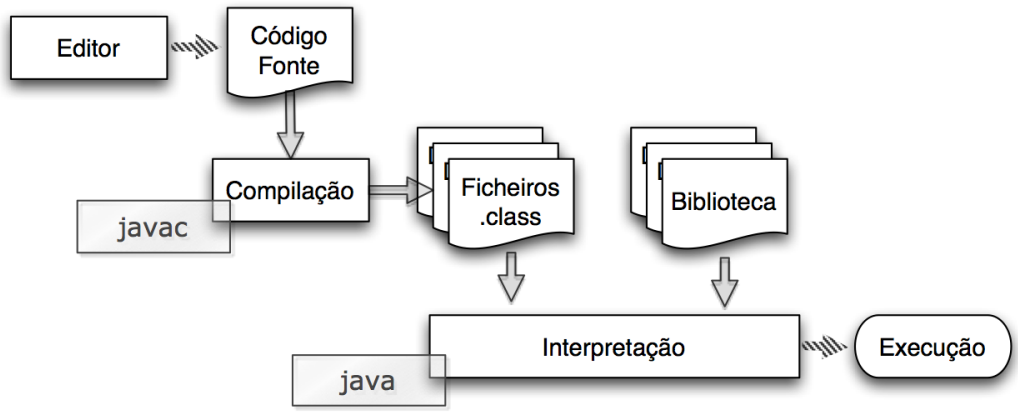
Um programa em Java, nada mais é que um conjunto de instruções enviadas ao processador para que ele possa executá-las. Para isso, o processador utilizará um compilador para converter os comandos em Java para uma linguagem que ele possa entender,

denominada Linguagem de Máquina (utiliza apenas os dígitos 0 e 1 – Sistema Binário). O compilador Java é chamado de Java compilador (Javac) e tem como objetivo transformar o código fonte em bytecodes. Os bytecodes são os arquivos convertidos pelo compilador que serão interpretados pela JVM.

"A linguagem de programação Java representa uma linguagem simples, orientada a objetos, multithread, interpretada, neutra de arquitetura, portável, robusta, segura e que oferece alto desempenho." Mendes (2009).

Orientada a objetos porque manipula as informações como entidades representativas denominadas objetos, construídas através de uma classe; Multithread é uma técnica de programação concorrente, ou seja, várias aplicações trabalham em paralelo de forma eficiente; Interpretada e neutra de arquitetura significa que, após compilado, é gerado um arquivo intermediário no formato bytecode, que poderá ser executado em qualquer arquitetura (Windows, Linux, Mac e Unix) que tenha uma máquina virtual Java instalada. A **Figura 4.1** ilustra os processos aos quais um programa em Java são submetidos para serem executados.

Figura 4.1: Processo de execução de um programa em Java.



Fonte: Lopes

4.1.2 XML

O XML, ou eXtended Markup Language, é um padrão para a formatação de dados, ou seja, uma maneira de organizar informações. Com a XML definem-se facilmente linguagens de marcação (Markup Languages).

Linguagem de marcação é um sistema para anotação de um texto de modo que ele seja sintaticamente distinguível, ou seja, trabalha com o armazenamento e manipulação de textos estruturados.

Ela é uma metalinguagem que define as regras para criar as linguagens de ”markup” para codificar exemplos de documentos particulares ou tipos de mensagens. Seu padrão de armazenamento de dados é em um formato de texto simples, o que significa que ele também pode ser aberto em qualquer computador.

Todas as informações contidas no XML estão dentro de tags, ou rótulos, e cada tipo de documento possui tipos diferentes destas, pois, elas são definidas pelo programador, ou seja, você pode inventar suas próprias tags. Um parâmetro é um atributo da TAG, ele serve para fornecer alguma informação extra a que contribua na construção, apresentação ou funcionalidade dela. O formato de uma tag em XML dentro do desenvolvimento de uma aplicação para Android é dada pela **Figura 4.2** a seguir:

Figura 4.2: Tag XML no Android.

```
<TextView
    android:layout_width="wrap_content"
    android:layout_height="wrap_content"
    android:text="Hello World!"
    android:id="@+id/textView" />
```

O Android faz uso da XML principalmente na criação de suas telas e instanciação de botões, caixas de texto e demais componentes visuais.

4.1.3 Kotlin

O Kotlin é uma nova linguagem de programação, estaticamente tipada, cuja sintaxe é mais expressiva e concisa do que a do Java. Esta linguagem foi recentemente adotada como uma linguagem de programação oficial no Android e é descrita da seguinte forma no próprio site da Android Developers: “É expressivo, conciso e poderoso. O melhor de tudo, é interoperável com os nossos idiomas e tempo de execução Android existentes.”.

Assim como o Java, Kotlin é convertida em bytecodes, através do compilador e interpretada pela própria JVM, fazendo com que esta interoperabilidade entre as duas

linguagens de programação possa ocorrer até mesmo dentro de um único aplicativo.

4.2 Banco de Dados e Firebase

Bancos de dados, ou bases de dados, em inglês databases, são coleções de informações estruturadas, ou não estruturadas (NoSql), que se relacionam de forma a evitar redundância e criar sentido quando necessário a abstração desses dados.

Uma base de dados é um componente crucial para qualquer sistema de informação, pois, todos os dados brutos dos quais ele faz uso e transforma em informações válidas, encontram-se armazenados nessas estruturas. Caso não haja um banco de dados interagindo com o sistema, os dados utilizados estarão disponíveis apenas em tempo de execução, ou seja, perderíamos todas as informações ao fecharmos o sistema.

Para controlar estes dados são utilizados SGBD's (Sistemas de Gerenciamento de Bancos de Dados), que consistem em um conjunto de programas responsáveis por este gerenciamento.

Figura 4.3: Exemplos de SGBD's.

Fonte: Magestore (2016)

Para aplicativos desenvolvidos na plataforma Android, uma opção nativa é a biblioteca SQLite, que atua como um "mini-SGBD", capaz de controlar diversas tabelas de diversos bancos de dados. Estas bases de dados são geradas localmente em um determinado diretório no aparelho e ficam disponíveis apenas à aplicação que o criou.

Em função desta limitação, para a aplicação desenvolvida será utilizada uma opção que permite o armazenamento de informações em um servidor acessível a todos os usuários do aplicativo de forma interativa, o Firebase.

4.2.1 Firebase

O Firebase é uma plataforma de desenvolvimento web e mobile focada em ser um back-end completo de fácil usabilidade, disponibilizando diversos serviços, inclusive de armazenamento de dados de forma não estruturada (NoSql), sistemas de autenticação, controle de qualidade, análise de desempenho e usabilidade e, o que mais nos interessa, o sistema de real time database, ou seja, manipulação de dados de forma concorrente em tempo real.

Para o desenvolvimento mobile, o Firebase disponibiliza uma série de dependências que, devem ser importadas para que possamos trabalhar de forma a utilizar os serviços que estas disponibilizam. A seguir, temos a lista de dependências disponibilizadas para o desenvolvimento com Android, das quais, utilizaremos apenas três:

- Realtime Database: Manipulação concorrente a dados em tempo real.
- Authentication: Autenticação de acesso de usuários.
- Storage: Armazenamento de arquivos.

O Firebase oferece serviços muito poderosos e de fácil implementação, e a maioria deles pode ser usada de forma gratuita.

4.3 Programas e Aplicações

Para iniciarmos com a codificação de um software, são necessárias as instalações de alguns programas, disponíveis gratuitamente, por algumas empresas que oferecem ferramentas de assistência a desenvolvedores em diversas plataformas.

No caso de uma aplicação Android, os serão necessários os programas listados

Figura 4.4: Dependências do Firebase.

Linha de dependência do Gradle	Serviço
com.google.firebase:firebase-core:15.0.0	Analytics
com.google.firebase:firebase-database:15.0.0	Realtime Database
com.google.firebase:firebase-firestore:15.0.0	Cloud Firestore
com.google.firebase:firebase-storage:15.0.0	Storage
com.google.firebase:firebase-crash:15.0.0	Crash reporting
com.google.firebase:firebase-auth:15.0.0	Authentication
com.google.firebase:firebase-messaging:15.0.0	Cloud Messaging
com.google.firebase:firebase-config:15.0.0	Configuração remota
com.google.firebase:firebase-invites:15.0.0	Invites e Dynamic Links
com.google.firebase:firebase-ads:15.0.0	AdMob
com.google.firebase:firebase-appindexing:15.0.0	Indexação de apps
com.google.firebase:firebase-perf:15.0.0	Monitoramento de Desempenho
com.google.firebase:firebase-functions:15.0.0	SDK de cliente do Cloud Functions para Firebase

Fonte: Firebase (2018)

nesta seção. Serão apresentadas quais suas funcionalidades, onde obtê-las e como instalá-las.

4.3.1 JDK e JRE

Abreviação para Java Development Kit (Kit de Desenvolvimento Java), é um conjunto de utilitários composto pelo compilador e pelas bibliotecas (API's) necessárias para criação de programas em Java e ferramentas úteis para o desenvolvimento e para testes dos programas escritos por esta linguagem de programação.

Este pacote é disponibilizado pela Oracle™ e já possui todo o ambiente necessário para desenvolver e executar aplicativos em java. No livro "Programando Android" de Mendnieks é citado este pacote:

> "O JDK oferece ferramentas como o compilador Java, utilizado por IDEs e SDKs para desenvolvimento de programas em Java. O JDK também contém um Java Runtime Environment (JRE), que permite a execução de programas Java, como o Eclipse, em seu sistema". (Mendnieks et al., 2012, pág. 18).

O JRE (Java Runtime Envirorment ou Ambiente de Execução Java) é o mínimo que você precisa ter instalado para poder rodar um aplicativo Java. É composto pela

JVM e pela biblioteca de classes Java, utilizadas para execução de aplicações. A máquina virtual Java é adicionada ao sistema operacional automaticamente, no caso de ainda não ter uma instalada no computador.

Figura 4.5: Download do JDK.

Fonte: Oracle (2018)

A instalação é bem simples, basta fazer o download do JDK na página da Oracle clicando sobre a imagem em destaque mostrada na **Figura 4.5**, aceitar o contrato de licença fornecido pela empresa, apresentado na **Figura 4.6** e por fim, selecionar para qual sistema operacional deseja obter a ferramenta à ser utilizado pelo desenvolvedor.

Figura 4.6: JDK - Seleção do JDK para o Sistema Operacional.

Java SE Development Kit 8u172

You must accept the Oracle Binary Code License Agreement for Java SE to download this software.

Accept License Agreement Decline License Agreement

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.99 MB	jdk-8u172-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.9 MB	jdk-8u172-linux-arm64-vfp-hflt.tar.gz
Linux x86	170.07 MB	jdk-8u172-linux-i586.rpm
Linux x86	184.91 MB	jdk-8u172-linux-i586.tar.gz
Linux x64	167.15 MB	jdk-8u172-linux-x64.rpm
Linux x64	182.08 MB	jdk-8u172-linux-x64.tar.gz
Mac OS X x64	247.87 MB	jdk-8u172-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	140.05 MB	jdk-8u172-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	99.35 MB	jdk-8u172-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.63 MB	jdk-8u172-solaris-x64.tar.Z
Solaris x64	97.06 MB	jdk-8u172-solaris-x64.tar.gz
Windows x86	199.11 MB	jdk-8u172-windows-i586.exe
Windows x64	207.3 MB	jdk-8u172-windows-x64.exe

Fonte: Oracle (2018)

4.3.2 Integrated Development Enviroment – IDE

O IDE, Ambiente de Desenvolvimento Integrado, é um software que ajuda todo processo de desenvolvimento de software e íntegra as várias ferramentas necessárias para

o desenvolvimento de softwares, ajudando todo o processo ficar mais fácil.

Sua principal função é ajudar o programador a editar o código que será usado para criar seu programa. Outra coisa muito importante que todos IDE's fazem é chamar o compilador, ou interpretador, dentro de certos parâmetros para gerar e/ou executar o programa criado, se ele não contiver erros.

Exitem diversas opções de IDE's, cada uma com suas características e voltadas para uma linguagem específica ou grupo de linguagens.

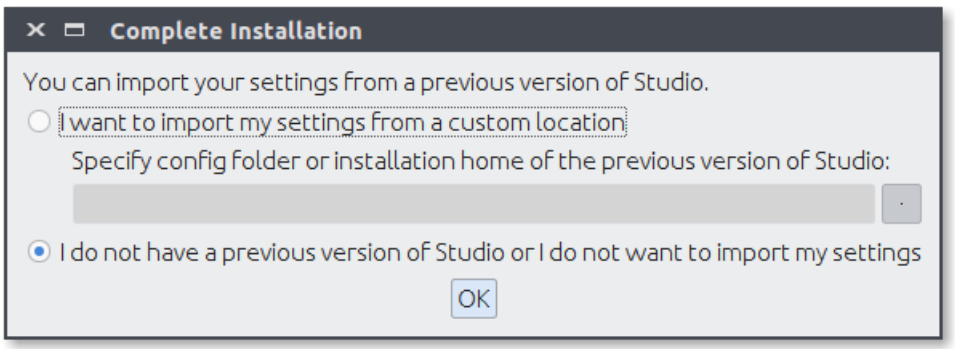
Quando se trabalha com a linguagem Java as principais IDE's utilizadas são o Eclipse, o Intellij e o Netbeans. A escolha de qualquer um deles depende da preferência do programador. Para o propósito deste trabalho, foi utilizada a ferramenta disponibilizada pela própria Google™ para desenvolvimento para Android, o Android Studio.

4.3.3 Instalando o Android Studio

Para instalar o Android Studio, é necessário que já possua em sua máquina o JDK, já mencionado neste mesmo trabalho, realizar o download da versão que deseja para o Sistema Operacional que possua em seu computador.

Após baixar os arquivos necessários, basta dar início ao processo e seguir os passos descritos a seguir, onde, na primeira tela, o sistema perguntará se deseja importar configurações previamente instaladas de versões anteriores da IDE:

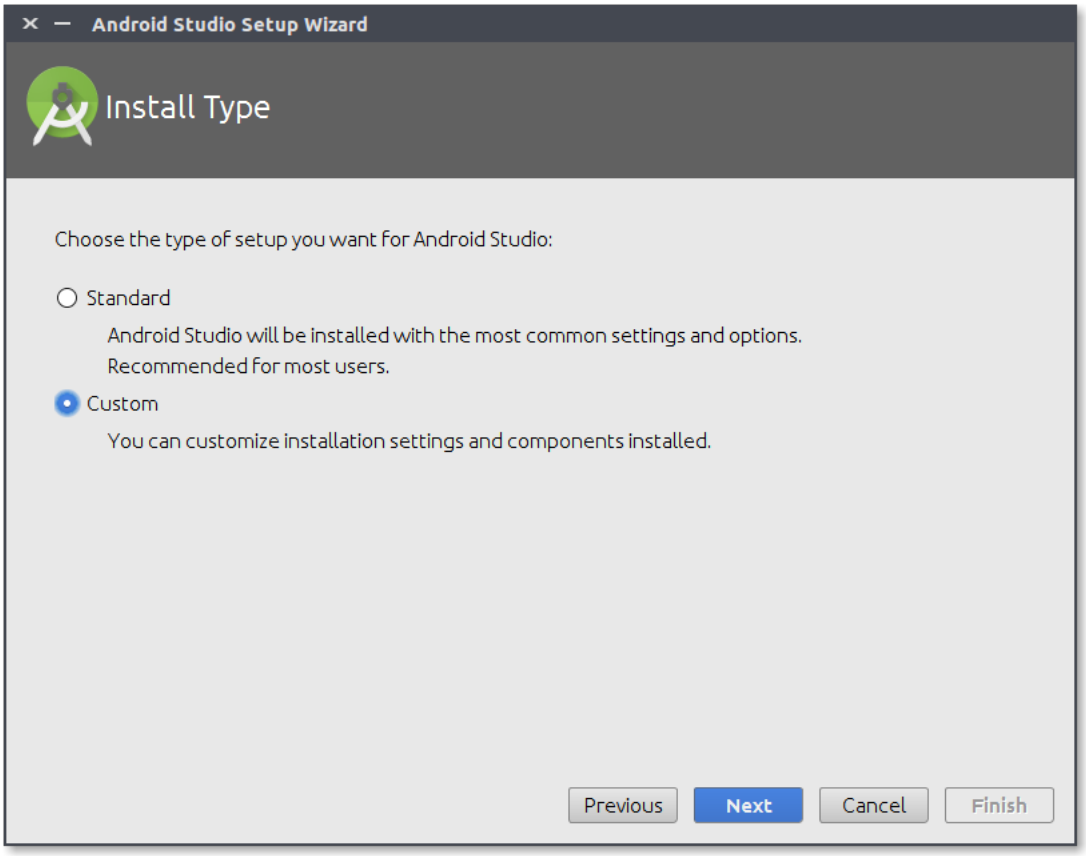
Figura 4.7: Instalação - Importação de Configurações previamente instaladas.



Em seguida, na tela de "Boas Vindas", basta avançar o processo. Logo na sequência, será solicitado o tipo de instalação que deseja, sendo as opções Standard ou Custom, que

significam Padrão ou Customizado.

Figura 4.8: Seleção de tipo de instalação.



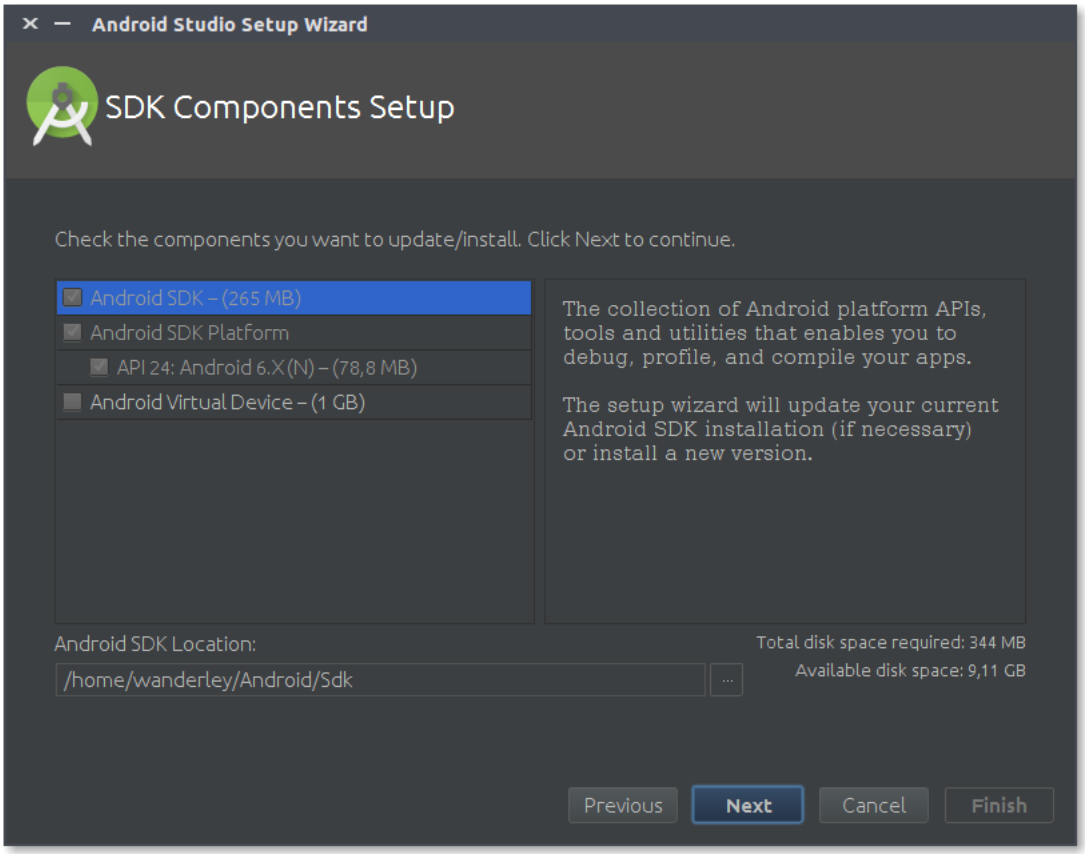
Ao escolhermos a opção Customizada, podemos selecionar as cores de estilo para a exibição da IDE e o local de destino da pasta SDK, assim como os componentes básicos para o SDK, SDK Plataform, API de desenvolvimento e Android Virtual Device – AVD, que é o emulador de ambiente mobile para testes das aplicações nos computadores.

O passo a passo da instalação da IDE pode ser visto no próprio site da Android Studio Developers, onde encontram-se uma lista de instruções a serem seguidas, assim como vídeos demonstrativos para cada um dos Sistemas Operacionais mais populares: Windows, MAC e Linux.

4.3.4 Android SDK e Android SDK Tools

Assim como o Java, o Android também possui um SDK, ou kit de desenvolvimento de software próprio que, é um pacote de ferramentas que auxiliam os desenvolve-

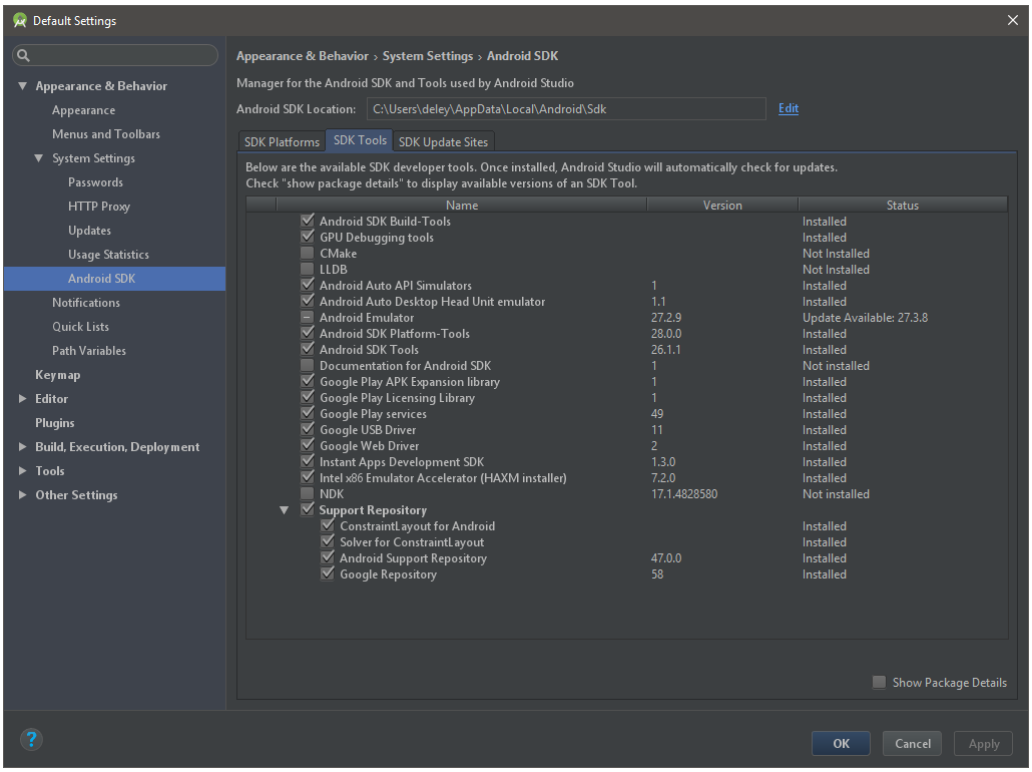
Figura 4.9: Definição do diretório para SDK.



dores fornecendo ferramentas de desenvolvimento, exemplos de códigos-fonte pré programados, emuladores e bibliotecas.

As ferramentas do SDK Tools Android já vêm configuradas por padrão na instalação do Android Studio. Através do SDK Manager, podemos baixar todas as atualizações mais recentes para as API's, ou Plataformas do Android e também, as versões mais atuais das ferramentas que auxiliam nesse desenvolvimento, nas abas SDK Plataforms e SDK Tools, como mostra a **Figura 4.10**:

Figura 4.10: SDK Manager - Android Studio.



5 SISTEMA DESENVOLVIDO

A aplicação será composta por listas de produtos que podem ser criadas e acessadas por qualquer usuário que seja adicionado a um dado grupo. Essas listas ficarão armazenadas em uma base de dados simples. Os produtos cadastrados em uma lista, quando já adquiridos, serão retirados dela e encaminhados para um relatório de aquisições, contendo o valor unitário, a quantidade adquirida e a identificação do comprador. Isso será um comprovante para posterior ressarcimento a ser efetuado pelos demais integrantes do grupo.

5.1 Levantamento de Requisitos

Segundo os processos da prototipação evolutiva, foi realizada uma coleta de informações e realizado um refinamento de requisitos sobre as principais necessidades relacionadas ao desenvolvimento do software.

Como este projeto baseia-se na criação, ou concepção, de uma ideia advinda de um *brainstorming*, entre o autor deste trabalho, juntamente a alguns colegas de graduação e um dos docentes do curso de Sistemas de Informação, os requisitos funcionais são parte do escopo definido nos objetivos deste mesmo documento.

Em seguida, foram traçadas estratégias focadas na agilidade do processo de desenvolvimento, com uma modelagem baseada no padrão de arquitetura e desenvolvimento MVC – Model View Controller – que divide a as funcionalidades do sistema em camadas, estruturando o projeto em três partes interconectadas de acordo com o tipo de interação cada um dos componentes exercerá dentro do softwares:

- Dados da Aplicação;
- Regras de Negócio;
- Lógica e Funções;

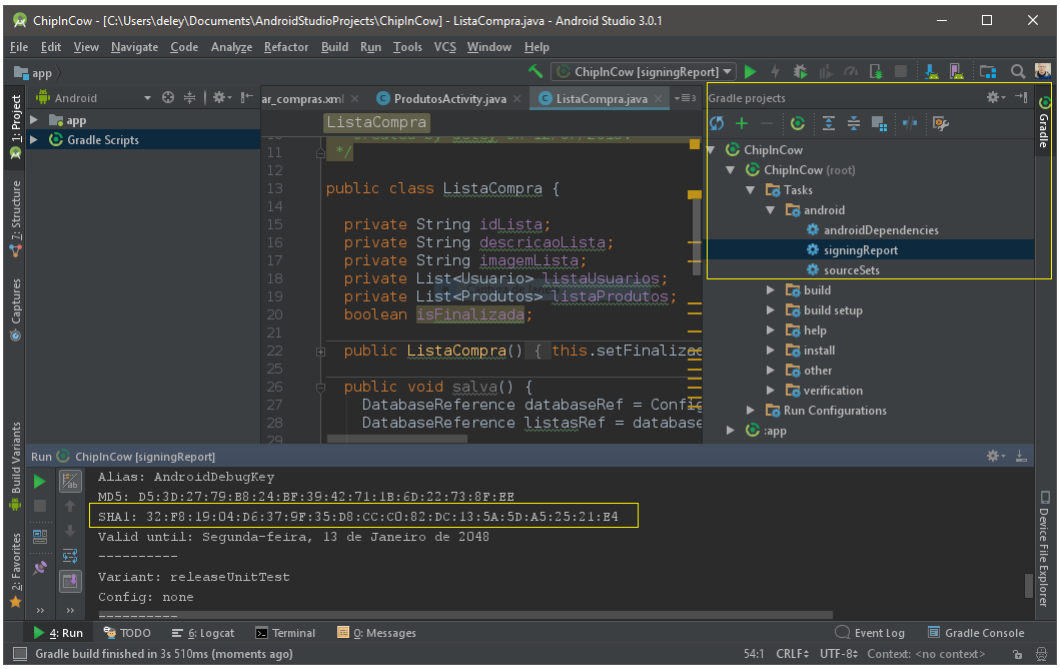
A partir deste ponto, passou-se a efetivamente ser implementada, ou codificada, a aplicação.

5.2 Implementação

Para dar início à codificação do aplicativo, temos antes que criar um projeto em branco, definindo o nome e o caminho referente ao pacote padrão deste, o caminho onde ele será salvo e sincronizá-lo com a entidade do Firebase que integrará com nosso software.

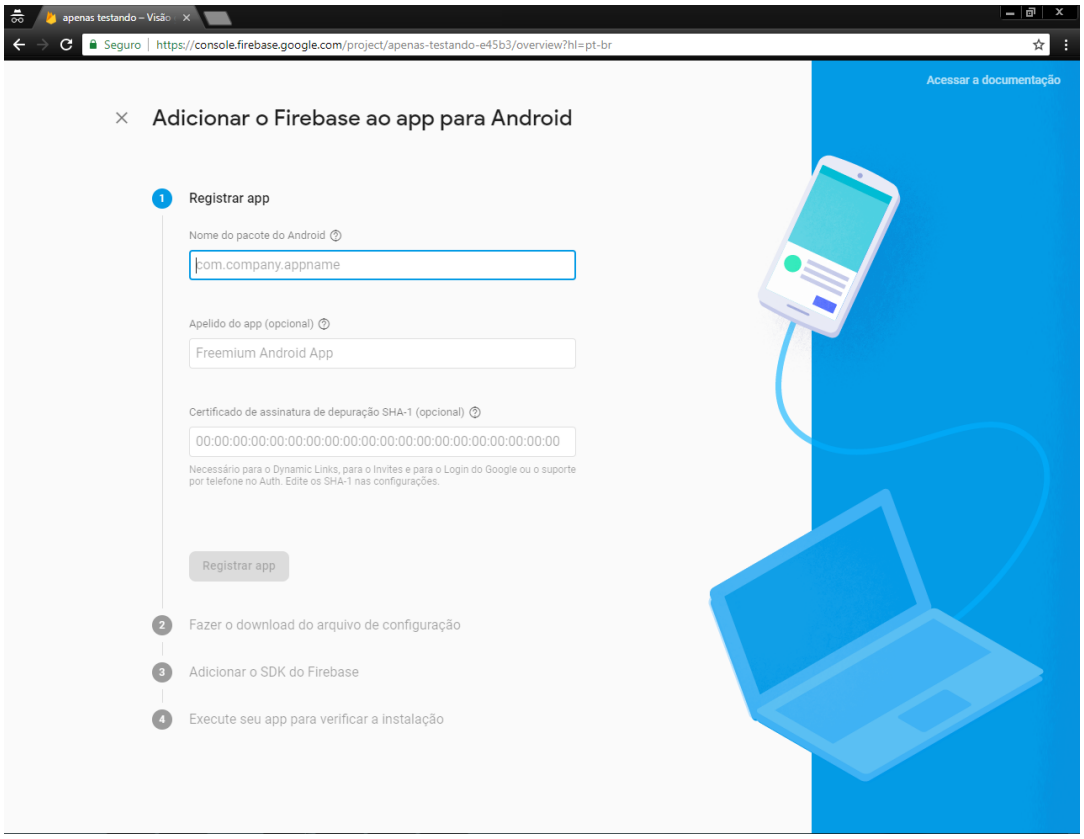
Após a criação, devemos gerar o certificado de assinatura de depuração. Isso é possível ao acessar a interface inicial da IDE, no canto superior direito, em Gradle, identificamos o root, ou administrador, do seu aplicativo e acessar o caminho: Tasks – android – signingReport; executando esta instância através de um duplo clique. Com isso, podemos recuperar as informações do SHA1 necessárias na sincronização com o Firebase no console localizado na parte inferior do Android Studio.

Figura 5.1: Obtenção do Certificado de Assinatura de Depuração.



A partir deste ponto, basta seguirmos o passo a passo das etapas descritas na própria plataforma, inicialmente fornecendo o pacote principal definido na aplicação, um apelido e a sequência de caracteres gerada como identificador do certificado, como mostrado na **Figura 5.2** abaixo:

Figura 5.2: Sincronização com a plataforma Firebase.

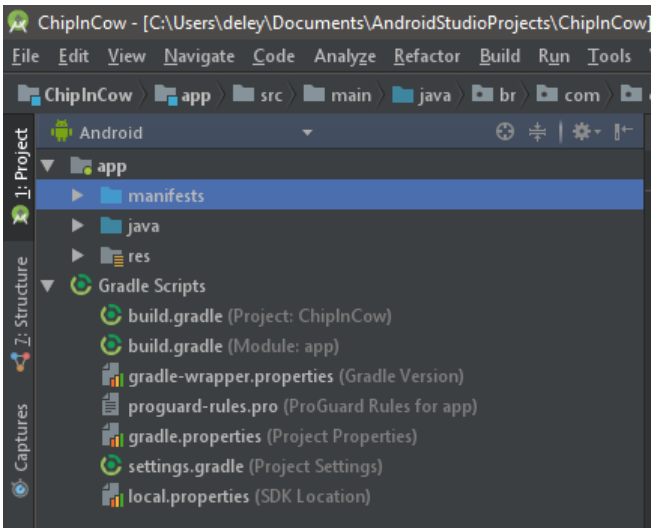


Fonte: Firebase

5.2.1 Hierarquia de Pastas

A própria IDE de programação – Android Studio – na criação de um novo projeto, já proporciona uma estrutura básica para manuseio e uma melhor disposição dos arquivos de código fonte, sendo estes divididos de acordo com a **Figura 5.3** abaixo, onde, na visualização Android, podemos ver as duas raízes principais desta disposição: app e Gradle Scripts.

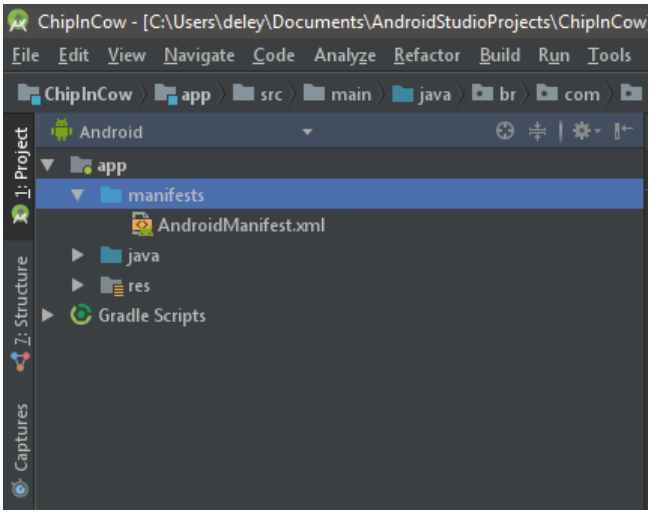
Figura 5.3: Hierarquia Android.



Dentro da raiz principal app, são identificadas outras três subpastas, cada uma contendo seus arquivos e outras subpastas.

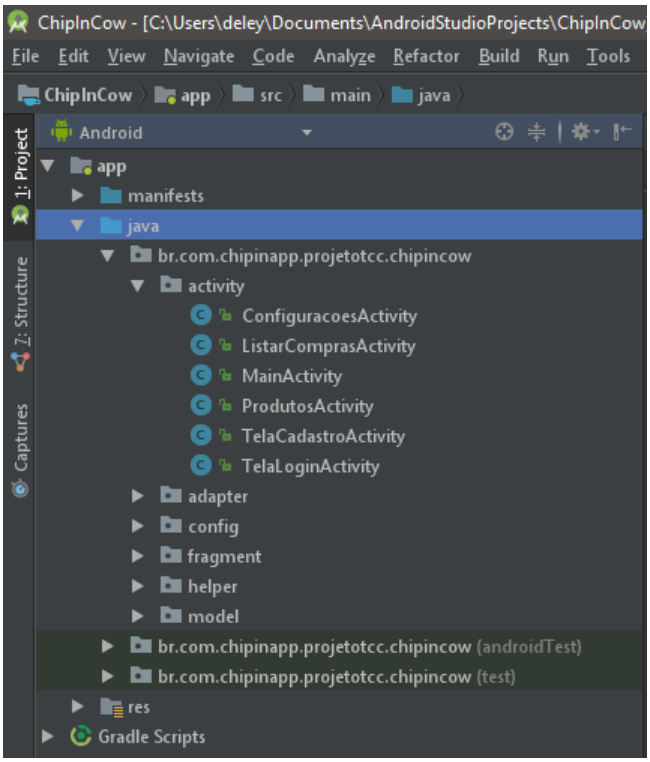
- manifest: Deve conter única e exclusivamente o arquivo AndroidManifest.xml.

Figura 5.4: Hierarquia de Pastas – Manifest.



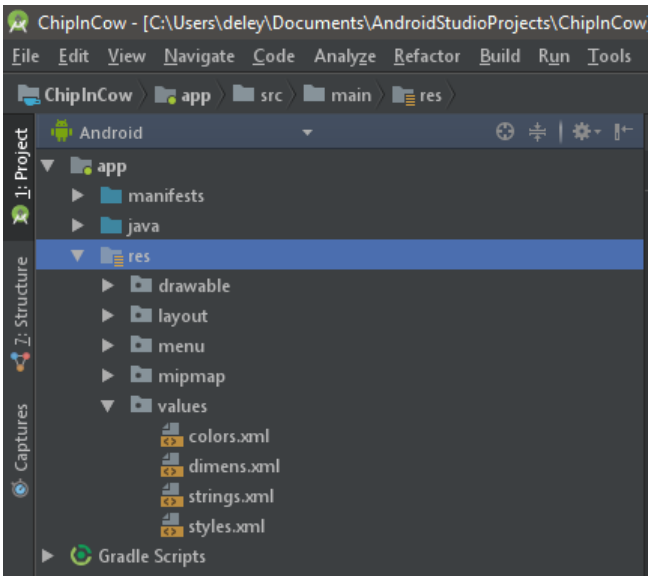
- java: Na geração de um projeto, esta pasta será composta por três entidades com o mesmo nome definido para o pacote da nossa aplicação, sendo duas delas de testes, que abrigarão as classes de testes unitários, caso desenvolvidos e uma onde adicionaremos nossos próprios pacotes e classes contendo as regras de negócio e lógica e funções.

Figura 5.5: Hierarquia de Pastas – Java.



- res: Abriga os arquivos em xml que podem ser a parte visual da aplicação, ou seja, suas telas e componentes que formam suas estruturas e recursos utilizados no aplicativo como imagens, sons, cores, entre outros.

Figura 5.6: Hierarquia de Pastas – Res.



5.2.2 AndroidManifest

O AndroidManifest, que é traduzido como Manifesto Android, é um arquivo, que leva precisamente este mesmo nome, obrigatório em todas as aplicações desenvolvidas para esta plataforma.

O arquivo de manifesto apresenta informações essenciais sobre o aplicativo ao sistema Android, necessárias para o sistema antes que ele possa executar o código do aplicativo.

Entre outras coisas, o arquivo do manifesto serve para:

- Definir o nome do pacote principal para a aplicação – primordial no momento da criação de uma instância de sincronização da ferramenta Firebase.
- Definir as instâncias das Activity's, como se fosse um cabeçalho, listando todas as activity's presentes no app, com suas hierarquias e demais dependências.
- Declarar todas as permissões, definindo quais aplicações externas poderão ser utilizadas pelo app.
- Declarar o nível mínimo da Android API que o aplicativo exige.
- Listar as bibliotecas às quais o aplicativo deve se vincular.

5.2.3 Gradle

Quando se trabalha em projetos complexos, com grande carga de dependências e regras de compilação, o que é o caso do Android, faz-se necessário a existência de um meio que simplifique esta tarefa. Quando o desenvolvimento Android era realizado na IDE de programação Eclipse, era preciso instalar uma série de pacotes auxiliares ou, criar um script de execução manualmente para que rodasse na linha de comando.

Para facilitar esta tarefa fazendo com que isso tudo ocorra de forma automática, a Google™ adotou oficialmente o Gradle, que é um sistema de automatização de build. Este sistema organiza a sequência de compilação de todos os arquivos inerentes ao projeto.

Além disso, são em suas classes de módulo e projetos que instanciamos todas as dependências externas necessárias para o correto funcionamento de nossa aplicação, a definição SDK mínimo em que o aplicativo poderá ser instalado e diversas outras funcionalidades.

5.3 Telas Principais, Funcionalidades e Testes

A seguir, encontram-se as principais interfaces de interação com o usuário final, uma descrição sucinta de suas funcionalidades e quais as ações que foram testadas para validação daquilo que foi planejado.

As telas de acesso à aplicação são as de Login e Cadastro, mostradas nas **Figuras 5.7 e 5.8**, onde o usuário deve entrar com as informações exigidas. Foram criadas validações para consistência de senhas com seus respectivos e-mails cadastrados e exceções caso algum campo deixe de ser devidamente preenchido.

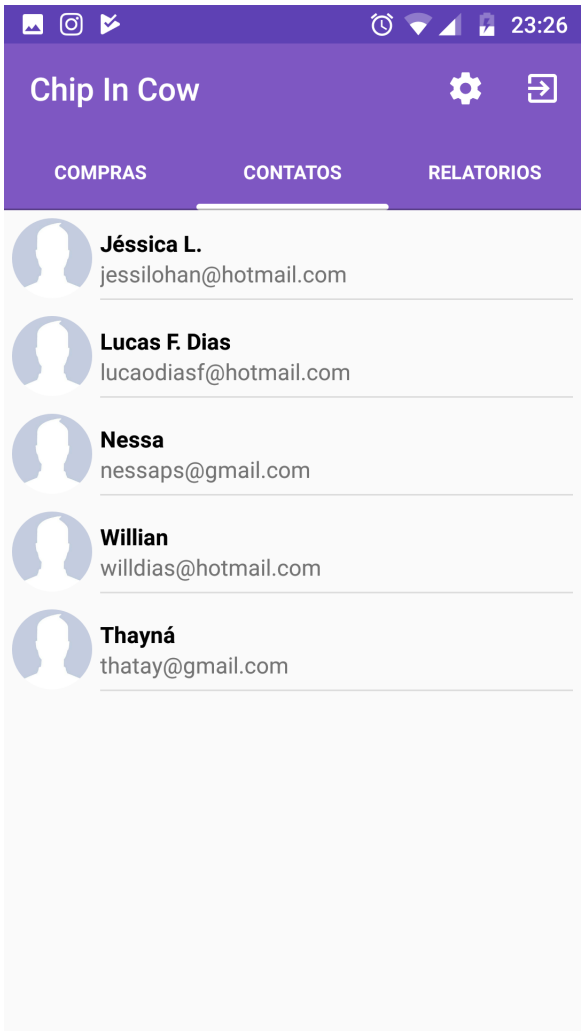
Figura 5.7: Tela de Login.

Figura 5.8: Tela de Cadastro de Usuário.

Ao acessar o aplicativo utilizando seu e-mail e senha cadastrados, é apresentada a tela principal do aplicativo. Esta possui três abas que listam os principais elementos do sistema: Compras, Contatos e Relatórios.

Nas abas Contatos e Relatórios, estarão listados todos os elementos referentes a estas entidades.

Figura 5.9: Aba de Contatos.



No canto superior direito da aplicação, existe uma figura com a forma de engrenagem, onde o usuário pode definir o nome e imagem, esta pode ser uma foto instantânea usando a câmera do aparelho ou uma imagem da galeria, que serão apresentadas para os seus contatos.

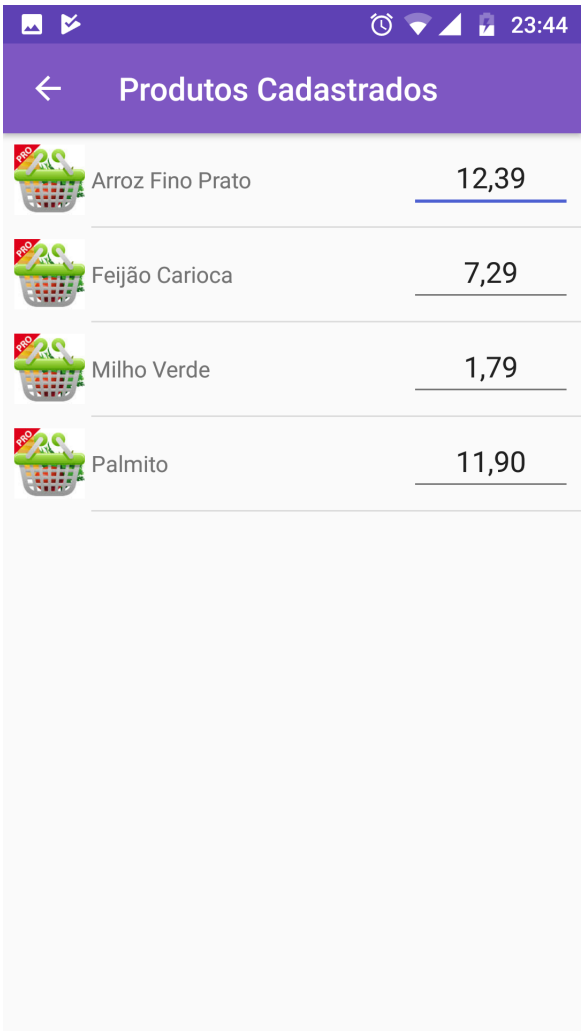
Figura 5.10: Tela de Configurações.

Na aba de Compras, são exibidos dois elementos separados, que também trazem uma listagem dos itens, identificados através de seus próprios títulos, assim como nas abas anteriores.

Figura 5.11: Aba de Compras.

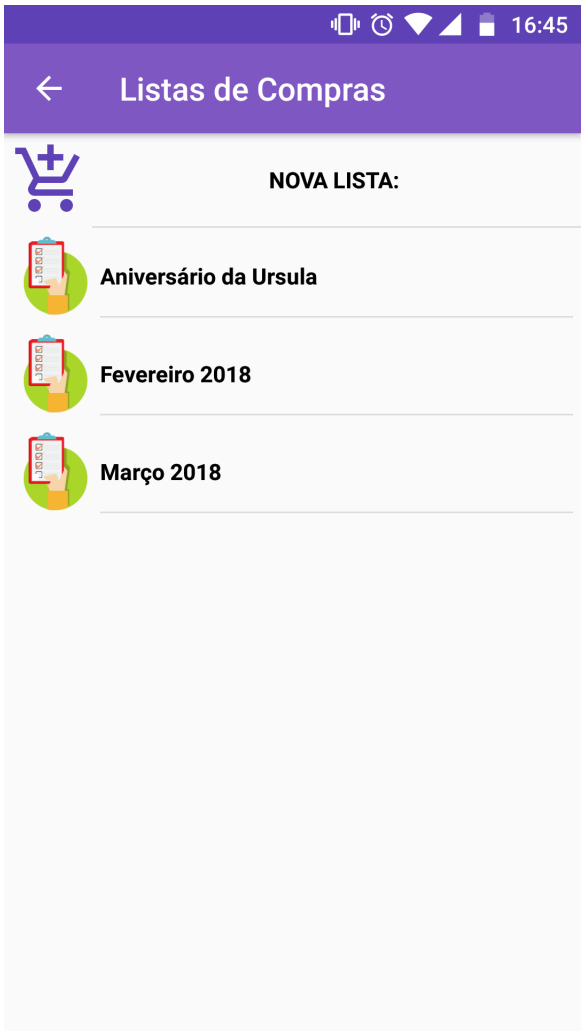
- Lista de Produtos:

Figura 5.12: Listagem de Produtos.



- Listagem de Listas de Compras:

Figura 5.13: Exibição de Listas Cadastradas.



Ao pressionar o item estático “Nova Lista” na parte superior da visão, iniciaremos o processo de inserção de uma nova instância de Lista de Compras, onde, devem ser selecionados os integrantes que farão parte desse grupo de compras, em seguida, definidos o nome, ou descrição, que identifique o novo elemento e por fim, selecionados todos os produtos que desejam adquirir.

Figura 5.14: Seleção de Integrantes.

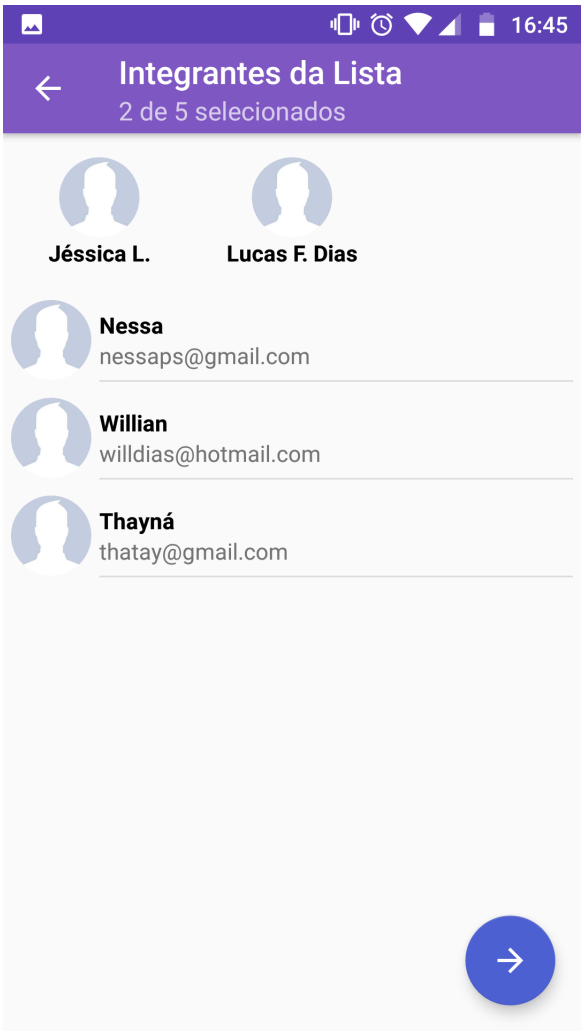


Figura 5.15: Definições sobre o Grupo.

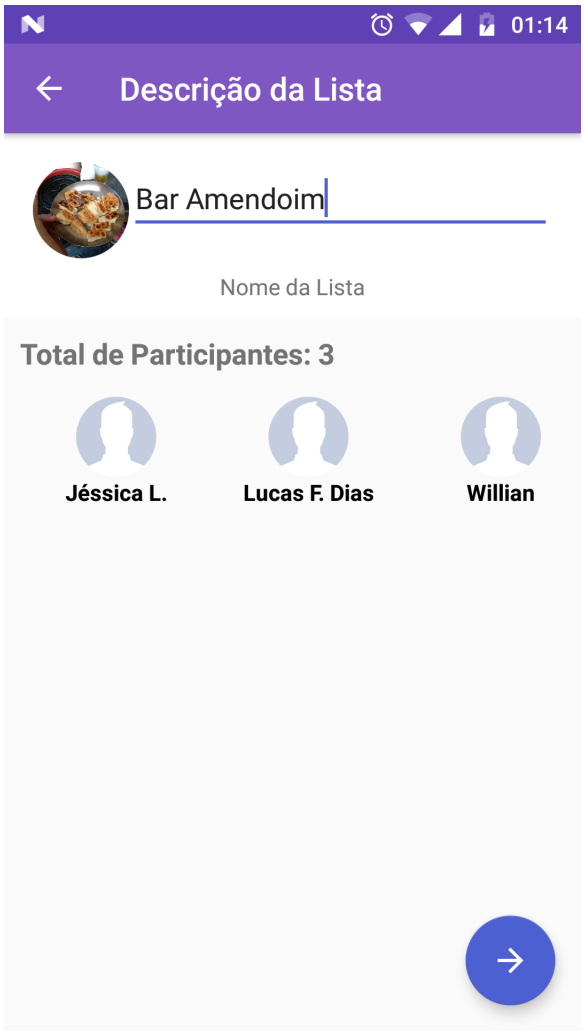
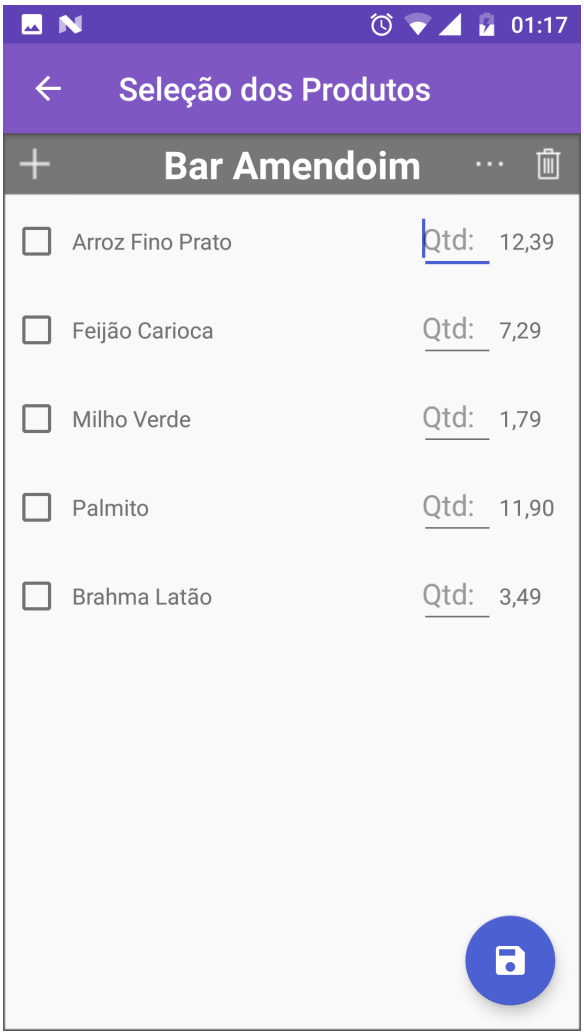


Figura 5.16: Seleção de Produtos.



Após realizar estas etapas, visando a inclusão de uma nova lista de compras para um determinado grupo de pessoas, as informações salvas poderão ser acessadas ao clicar sobre o elemento na listagem principal da aba Compras, item Listas de Compras. A interface traz todos os produtos adicionados pela ação de inclusão efetuada anteriormente, um valor total de fatura e a opção de encerrar, alterando seu status para paga, enviando-a para os Relatórios.

Figura 5.17: Modelo de Lista em Aberto.

Ao encerrar uma lista em aberto, clicando-se no botão mostrado na **Figura 5.17** acima, será gerado um comprovante, ou comanda finalizada, onde é possível visualizar a descrição e valor de cada produto, multiplicado pela quantidade adquirida, o valor total e o nome do integrante e valor rateado, como mostra a ilustração na **Figura 5.18** abaixo:

Figura 5.18: Modelo de Comprovante ou Comanda.

5.4 Publicando uma Aplicação

Uma aplicação é distribuída através de um arquivo com o sufixo .apk que empacota seu código compilado junto aos dados e recursos utilizados, sendo este posteriormente exportado para uma loja virtual.

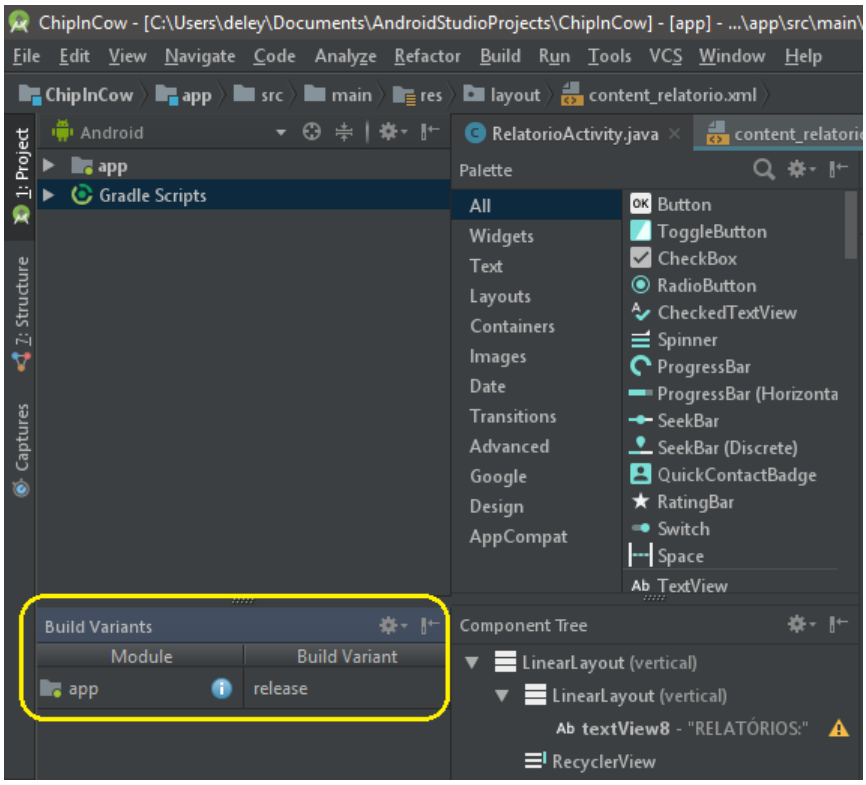
Dentre as lojas virtuais que podemos acessar para publicarmos nosso aplicativo, encontram algumas como a Samsung Galaxy Apps, a Amazon AppStore, a Opera Mobile Store e por fim, a loja oficial da mantenedora do sistema operacional Android, a Google Play, que diferentemente das demais citadas, é a única que não exige a utilização de um

framework proprietário para que possamos publicar o aplicativo.

Para publicar uma aplicação é necessário que ela esteja assinada digitalmente, sendo necessária a criação de uma chave do desenvolvedor para identificá-lo como responsável pela aplicação. Para gerar esta chave acompanhado de o arquivo apk, são utilizadas funcionalidades da própria IDE de programação.

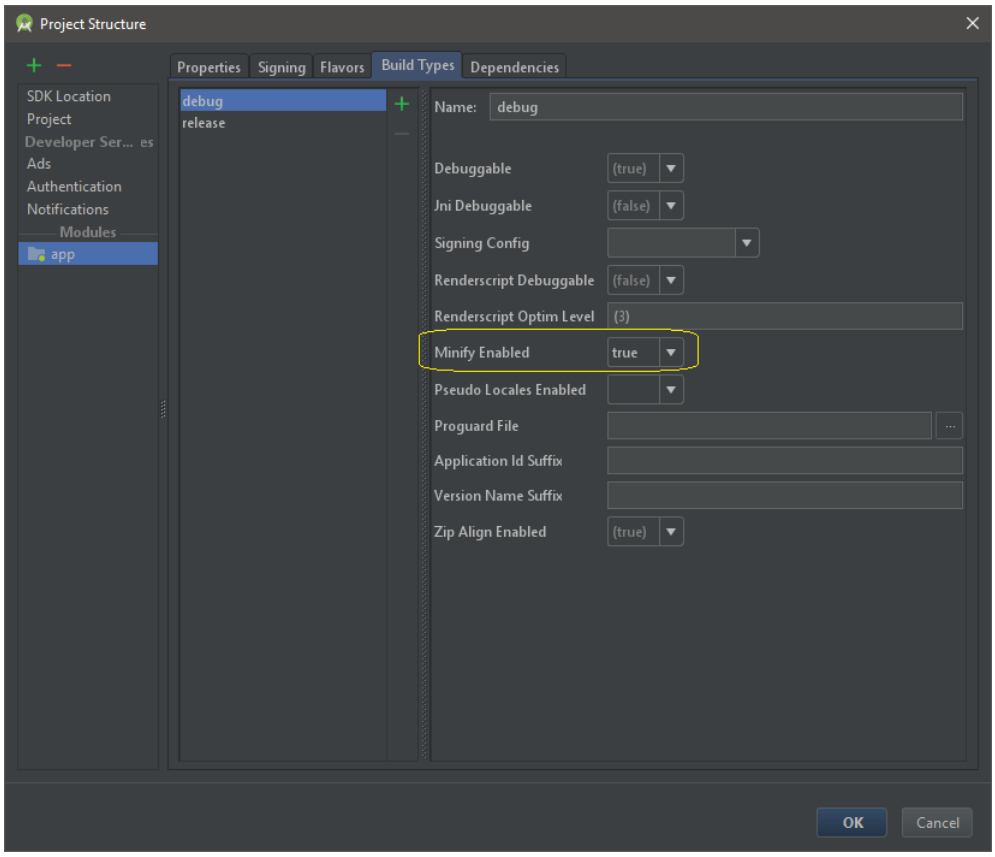
Primeiramente, deveríamos indicar à IDE que desejamos que o módulo de compilação seja identificado como uma versão acabada, não apenas de testes, alterando a variante de build de debug para release, através do menu Build – Select Build Variant.

Figura 5.19: Indicação de Tipo de Build - Para Publicação.



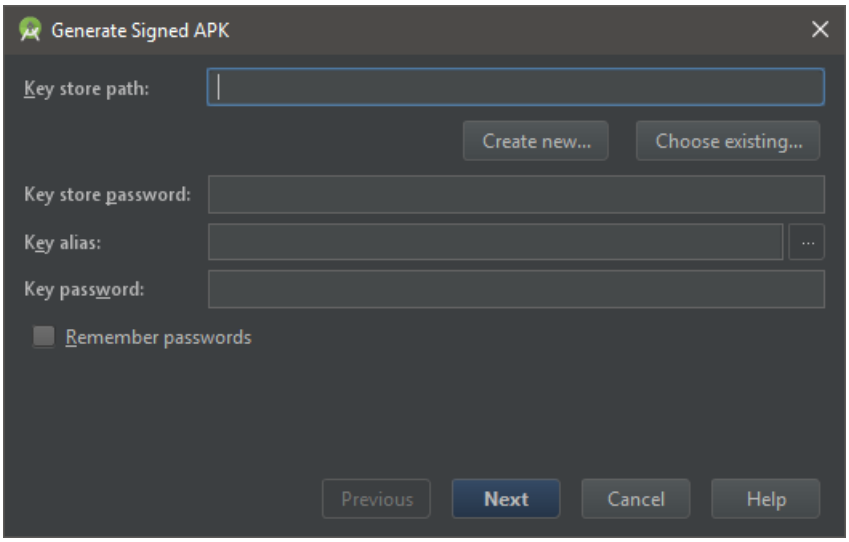
Para que nosso aplicativo tenha uma maior segurança, dificultando com que outras pessoas possam realizar a engenharia reversa, ofuscaremos nosso código fonte. Para isso, basta alterar a opção Minify Enable para “true”, através do caminho: File – Project Structure – app – Build Types.

Figura 5.20: Ofuscar Código Fonte.



Para criarmos a chave de assinatura do desenvolvedor, juntamente ao arquivo apk, basta preenchermos todas as informações exigidas e seguir o passo a passo das instruções contidas no menu Build – Generate Signed APK, mostrada na **Figura 5.21** abaixo:

Figura 5.21: Gerar Chave de Assinatura e APK.



Depois de assinada, ela está pronta para ser distribuída para usuários com dispositivos equipados com a plataforma Android.

Por fim, utilizaremos a principal opção de distribuição, Google Play, por ser a mais simples e eficiente quando se deseja atingir um grande número de usuários, visto que pode ser acessada a partir da maioria dos dispositivos. Deve-se possuir uma conta Google^{TM}, cadastrar-se como desenvolvedor na Google Play Developer Console, criar um projeto inserindo todas as informações exigidas, em seguida, acessar o menu lateral "Versões de apps", selecionar o tipo de publicação e seguir o passo a passo, respondendo à classificação de conteúdo e preencher as informações de Preço e Distribuição.

6 CONSIDERAÇÕES FINAIS

O trabalho realizado mostra o desenvolvimento de uma aplicação mobile, idealizada e desenvolvida pelo autor deste trabalho, como um simples protótipo minimamente funcional, que possa auxiliar integrantes de um nicho de mercado, ou público-alvo, com uma abordagem simples.

Os objetivos gerais e específicos foram atingidos com a criação do aplicativo Chip In Cow, onde, durante todo o período de evolução e implementação, a metodologia de prototipagem e o embasamento em modelos aplicados à gestão ágil, foram devidamente demonstrados e citados, sendo de suma importância para a conclusão deste trabalho.

As maiores dificuldades encontradas durante todo o percurso foram a busca por informações técnicas atualizadas, uma vez que, a plataforma Android evolui numa taxa de velocidade muito alta se comparada com outros sistemas, o que reflete diretamente nas formas de se construir certas estruturas mais complexas e particulares da plataforma.

Como este trabalho foi construído como uma Prototipagem Evolutiva, com o objetivo de estender os conceitos abordados, uma proposta é que esta aplicação seja integrada a uma API de um estabelecimento comercial, para que este estabelecimento possa disponibilizar seus produtos diretamente no celular do cliente.

Referências Bibliográficas

aaaa.

W. Frank Ableson, Chris King, and Robi Sen. *Android em Ação*. Elsevier Brasil, São Paulo, SP, 3ª edition, 2012. ISBN 978-85-3524-841-8.

android Developer. Como iniciar uma atividade, 2018a. URL ⟨https://developer.android.com/training/basics/activity-lifecycle/starting⟩.

android Developer. Versões da plataforma, 2018b. URL ⟨https://developer.android.com/about/dashboards/⟩.

H M Deitel. *Java: Como Programar*. Pearson Prentice Hall, 6ª edition, 2005. ISBN 978-85-7605-019-3.

Evenfy. Conta Coletiva, 2018. URL ⟨https://www.evenfy.com/pt-br/⟩.

Firebase. Configuração do Projeto. URL ⟨https://console.firebase.google.com/?hl=pt-br⟩.

Firebase. Adicionar o Firebase ao seu projeto do Android, 2018. URL ⟨https://firebase.google.com/docs/android/setup?authuser=0⟩.

G Franz. *Apostila de Programação Android - Programação Android Descomplicada*. Agbook, 2012. URL ⟨https://books.google.com.br/books?id=-KER2EyXClkC⟩.

Antônio Carlos Gil. *Como Elaborar Projetos de Pesquisa*. Atlas, São Paulo, SP, 5ª edition, 2010. ISBN 9788522458233.

IBlue. Fechando a Conta, 2015. URL ⟨http://www.iblue.com.br/site/?page{_}id=35⟩.

IDC. Smartphone OS, 2017. URL ⟨https://www.idc.com/promo/smartphone-market-share/os⟩.

Diogo Júnior. Introdução à Programação de Android. *Revista Programar*, page 21, mar 2010. ISSN 1647-0710. URL ⟨http://www.revista-programar.info⟩.

Juliana Jenny Kolb. Modelo Prototipagem, 2013. URL ⟨http://jkolb.com.br/prototipagem/⟩.

Ricardo R. Lecheta. *Google Android para Tablets - Aprenda a desenvolver aplicações para o Android - De smartphones a tablets*. NOVATEC, São Paulo, SP, 2012. ISBN 978-85-7522-292-8.

Ricardo R. Lecheta. *Google Android - Aprenda a criar aplicações para dispositivos móveis com o Android SDK*. NOVATEC, São Paulo, SP, 3ª edition, 2013. ISBN 978-85-7522-344-4.

Adriana Lee. Using the Apple Newton For Twitter, 2011. URL ⟨https://www.technobuffalo.com/2011/06/10/using-the-apple-newton-for-twitter/⟩.

Rui Pedro Lopes. Java: Uma introdução à Linguagem de Programação. URL ⟨http://www.ipb.pt/{~}ellobo/introj/intro.php⟩.

Magestore. My SQL – Magento 2 requirement, 2016. URL ⟨https://www.magestore.com/magento-2-tutorial/magento-2-system-requirements/my-sql-magento-2-requirement/⟩.

Manuel Meirinhos and Antônio Osório. O estudo de caso como estratégia de investigação em educação. *Instituto Politécnico de Bragança*, page 17, 2010. URL ⟨https://www.eduser.ipb.pt/index.php/eduser/article/view/24⟩.

D R Mendes. *Programação Java com Ênfase em Orientação a Objetos:*. NOVATEC, 2009. ISBN 978-85-7522-176-1. URL ⟨http://books.google.com.br/books?id=tNw9J-UwtvsC⟩.

Zigurd Mendnieks, Laird Dornin, G. Blake Meike, and Masumi Nakamura. *Programando o Android*. NOVATEC, São Paulo, SP, 2012. ISBN 978-85-7522-284-3. URL ⟨https://books.google.com.br/books?id=5skjkwEACAAJ⟩.

P. A. CAUCHICK Miguel. Estudo de caso na engenharia de produção: estruturação e recomendações para sua condução. *POLI-USP*, page 14, 2007. URL ⟨http://www.scielo.br/pdf/{\%}0D/prod/v17n1/14.pdf⟩.

J B Monteiro. *GOOGLE ANDROID - Crie aplicações para celulares e tablets*. CASA DO CODIGO, 2014. ISBN 978-85-66250-02-2.

nsc DC. Apple coloca primeiro iPhone no grupo de equipamentos obsoletos da marca, 2013. URL ⟨http://dc.clicrbs.com.br/sc/noticias/noticia/2013/05/apple-coloca-primeiro-iphone-no-grupo-de-equipamentos-obsoletos-da-marca-4129153.html⟩.

Damon Oehlman and Sébastien Blanc. *Aplicativos Web Pro Android - Desenvolvimento Pro Android Usando HTML5, CSS3 & JavaScript*. Ciência Moderna, Rio de Janeiro, RJ, 2012. ISBN 978-85-399-0295-5.

Oracle. Java SE Downloads, 2018. URL ⟨http://www.oracle.com/technetwork/pt/java/javase/downloads/index.html⟩.

Carlson Osbourne. HTC T-Mobile G1 Mobile Phone Review - Android and 3.5G Combined, 2009. URL ⟨http://gadget-village.blogspot.com/2009/09/htc-t-mobile-g1-mobile-phone-review.html⟩.

Lúcio C.O. Pereira and Michel L. da Silva. *Android para Desenvolvedores*. BRASPORT, Rio de Janeiro, RJ, 2ª edition, 2009. ISBN 978-85-7452-405-4. URL ⟨http://books.google.com.br/books?id=8u9wJowXfdUC⟩.

Roger S Pressman. Engenharia de Software - Uma Abordagem Profissional, 2009.

Cleuton Sampaio. Características de dispositivos móveis, 2013. URL ⟨http://www.obomprogramador.com/2013/10/curso-de-criacao-de-apps-moveis-com{_}22.html⟩.

Ian Sommerville. *Engenharia de Software*. Addison Wesley, São Paulo, SP, 8ª edition, 2008. ISBN 9788588639287. URL ⟨http://tmv.edu.in/pdf/DiplomaSyllabus/Computer/TY{_}fifth{_}sem/FifthSemesterCurriculum.pdf⟩.

SONDA. Mobilidade tecnológica: agilizando a comunicação dentro do negócio, 2016. URL ⟨https://blog.sonda.com/mobilidade-tecnologica-agilizando-a-comunicacao-dentro-do-negocio/⟩.

A S Tanenbaum. *Sistemas Operacionais Modernos*. Pearson Prentice Hall, São Paulo, SP, 3ª edition, 2009. ISBN 978-85-7605-237-1.

AUTORIZAÇÃO

Autorizo a reprodução e/ou divulgação total ou parcial do presente trabalho, por qualquer meio convencional ou eletrônico, desde que citada a fonte.

Diamantina, 06 / 08 / 2018.

Wanderley das Dores Ferreira Júnior

Wanderley das Dores Ferreira Júnior

ferreirajr.wanderley@gmail.com

Universidade Federal dos Vales do Jequitinhonha e Mucuri

Campus JK - Rodovia MGT 367 - Km 583, nº 5000 - Diamantina - CEP 39100-000.