CENTRO UNIVERSITÁRIO DE ANÁPOLIS – UNIEVANGÉLICA
CURSOS SUPERIORES DE COMPUTAÇÃO

APPVIS – APLICATIVO DE ACESSIBILIDADE E INCLUSÃO PARA PESSOAS COM DEFICIÊNCIA VISUAL

DIOGO MENDES DE SOUZA
LUIZ FELIPE FERREIRA DE MORAIS

ANÁPOLIS - GO
2016

DIOGO MENDES DE SOUZA
LUIZ FELIPE FERREIRA DE MORAIS

APPVIS – APLICATIVO DE ACESSIBILIDADE E INCLUSÃO PARA PESSOAS COM DEFICIÊNCIA VISUAL

Trabalho de conclusão de curso apresentado como requisito parcial à obtenção do grau de Bacharel em Engenharia de Computação do programa de Graduação dos cursos superiores de Computação, Centro Universitário de Anápolis – UniEVANGÉLICA.

Orientador: Prof. McGill Evaristo Dias

ANÁPOLIS - GO
2016

DIOGO MENDES DE SOUZA
LUIZ FELIPE FERREIRA DE MORAIS

APPVIS – APLICATIVO DE ACESSIBILIDADE E INCLUSÃO PARA PESSOAS COM DEFICIÊNCIA VISUAL

Banca Examinadora

.....

Prof. McGill Evaristo Dias.

Orientador

.....

Prof.....

Convidado

.....

Prof.....

Convidado

Nota:.....

Anápolis, ... de ..... de 2016.

DEDICATÓRIA

Dedicamos esse trabalho aos nossos familiares, em especial aos nossos pais e irmãos.

Aos amigos que sempre nos ajudaram e incentivaram durante esse período acadêmico.

Aos colegas de classe que nos ajudaram e retiraram diversas dúvidas.

AGRADECIMENTOS

Agradecemos primeiramente a Deus, por te nos proporcionado saúde, sabedoria e fé para levantarmos a cada dia e lutar pela realização de mais um sonho.

As nossas famílias, especialmente aos nossos pais e irmãos, pelas orações, estímulos e grande amor dado em todos os momentos das nossas vidas.

Aos amigos por todo o incentivo e pelos momentos felizes e inesquecíveis de lazer.

A todos os meus professores, desde a escola até a universidade, que contribuíram para nossa formação acadêmica, pessoal e profissional.

De maneira muito especial, agradecemos ao professor McGill Evaristo Dias pela paciência, dedicação e apoio durante a realização deste trabalho, estando sempre desposto a esclarecer nossos dúvidas.

Ao senho Massani, por ter nos esclarecido diversas dúvidas que facilitou e motivou a criação do aplicativo, resultado dessas pesquisa.

RESUMO

O objetivo deste trabalho é desenvolver um aplicativo para auxiliar pessoas deficientes visuais na realização de compras durante o cotidiano. Foram utilizadas algumas técnicas de desenvolvimento, que foram aplicadas durante o período de graduação. Este trabalho apresentará passo a passo o desenvolvimento do aplicativo desde o início até a sua conclusão, especificando as técnicas, frameworks utilizados dentro do ambiente de desenvolvimento.

Palavra-chave: Deficiente visual, *Android*, *Frameworks*, *API*

Abstract

The objective of this work is to develop an application to assist visually impaired people in making purchases for the daily life, some development techniques that have been applied during the graduation period will be used. This paper presents step by step, the development of the application from the start to its completion, specifying techniques, frameworks used inside the development environment.

Keyword: Visually impaired, Android, frameworks, API

Lista de ilustrações

Lista de abreviaturas e siglas

API	*Application Programming Interface*
APPVIS	Aplicativo de Acessibilidade e Inclusão para Pessoas com Deficiencia Visual
CMMI	*Capability Maturity Model – Integration*
DSDM	*Dynamic Systems Development Method*
FDD	*Feature Driven Development*
FTT	Fábrica de Tecnologias Turing
HTTP	Hypertext Transfer Protocol
IBGE	Instituto Brasileiro de Geografia e Estatística
IEEE	*Institute of Electrical and Electronic Engineers*
PNS	Pesquisa Nacional de Saúde
UML	Unified Modeling Language
XML	eXtensible Markup Language
XP	*Extreme Programming*

Sumário

Introdução

Um dos maiores problemas enfrentados hoje por deficientes visuais são as dificuldades nos atendimentos específicos em estabelecimentos que talvez não estejam preparados para atendê-los de uma forma qualitativa. O deficiente em muitos casos precisa da compreensão dos lojistas ou funcionários para conseguir realizar a compra de um produto qualquer.

De acordo com Villela (2015)

> Dados do Instituto Brasileiro de Geografia e Estatística (IBGE) revelam que 6,2% da população brasileira tem algum tipo de deficiência. A Pesquisa Nacional de Saúde (PNS) considerou quatro tipos de deficiência: auditiva, visual, física e intelectual. Dentre os tipos deficiência pesquisados, a visual é a mais representativa e atinge 3,6% dos brasileiros, sendo mais comum entre as pessoas com mais de 60 anos (11,5%). O grau intenso ou muito intenso da limitação impossibilita 16% dos deficientes visuais de realizarem atividades habituais.

Segundo Aparecida (2004) 0 uso da tecnologia facilita aos portadores de necessidades especiais o desenvolvimento de aprendizagem com os recursos de escrita, leitura e pesquisa de informação. No Brasil, o que permitiu o acesso do deficiente visual ao mundo da informática foi o lançamento dos programas leitores de tela, como o DOSVOX, o VIRTUAL VISION e JAWS. Os programas interpretam toda a parte visual da tela do computador e, após essa interpretação, cada ferramenta, de forma diferente, pronuncia para o deficiente visual o que está sendo executado na tela, de acordo com a tarefa que está sendo processada.

Com o avanço da tecnologia da informação, podemos observar o tamanho do impacto positivo causado na sociedade, onde a computação resolveu diversos problemas e vem modificando cada dia mais a vida das pessoas.

Durante o dia-a-dia pode-se notar que grande partes das lojas, empresas, escolas e até mesmo cidades não são inclusivas para pessoas com deficiência visual. Na cidade de Anápolis, pode-se perceber que a mesma não possuem recursos nas ruas e nas sinalizações de forma a facilitar o acesso e garantir a segurança dos mesmos.

Desta forma atividades, que são consideradas normais por grande porcentagem da população, muitas das vezes se tornam atividades complexas para pessoas com algum tipo de deficiência visual.

Com isso, existe alguma forma de facilitar a vida destas pessoas em situações que vivenciam em seus cotidianos? De proporcionar as mesmas maior autonomia e acessibilidade? De tornar as redes de lojas inclusivas para pessoas com deficiência visual?

Desta forma, este trabalho tem como problematização: se a criação de um aplicativo (AppVis) pode ajudar os deficientes visuais a realizarem suas compras em lojas como farmácias, supermercados entre outras lojas?

1 Referencial Teórico

1.1 Deficiência Visual

Segundo MONTEIRO (2012, apud MAIOLA; SILVEIRA, 2009):

> A deficiência visual engloba o universo de pessoas cegas e com baixa visão (ou visão subnormal) e é decorrente de problemas de diferentes ordens, a saber: congênita, adquirida, genética ou degenerativa. Dentre as principais causas da deficiência visual, encontram-se: a retinopatia da prematuridade, que consiste na imaturidade da retina decorrente de partos prematuros ou de excesso de oxigênio na incubadora; a catarata congênita, causada pela ocorrência de rubéola ou outras infecções intrauterinas; o glaucoma congênito, doença que pode ser de ordem genética, ou fruto de infecções, dentre outros fatores.

Para que a pessoa seja considerada deficiente visual, a acuidade[1] deve ser menor ou igual a 0,05 no olho que possui melhor correção óptica. Já aqueles que possuem baixa visão, a acuidade visual precisa ter entre 0,03 e 0,05 no olho que possui a melhor correção óptica (BRASIL, 2016).

De acordo com a concepção educacional está deficiência se define a partir da eficiência da visão, desta maneira, uma pessoa é considerada cega quando apresenta desde a ausência total da visão (amaurose[2]) até a percepção de luz. A cegueira parcial é a condição onde os indivíduos percebem vultos e distâncias, necessitam do Braille e dos outros sentidos para perceberem o mundo. Pessoas com baixa visão utilizam sua visão residual para a situação educacional, incluindo leitura e escrita, com ou sem recursos ópticos para as situações do seu cotidiano (ORMELEZI, 2000).

No ano de 2000, foi contemplada a Lei Federal 10.098 que tem como objetivo a promoção do acesso de pessoas deficientes ou com mobilidade reduzida a locais públicos. A lei inclui todos os tipos de deficientes, pois eles necessitam da eliminação de barreiras que lhes assegure acesso aos bens culturais e sociais, como também segurança na locomoção.

---

[1] Capacidade mínima de percepção de um órgão do sentido.

[2] Enfraquecimento ou perda completa da vista, sem lesão alguma do aparelho visual, nem obstáculo nenhum à passagem dos raios luminosos.

1.1.1 Tecnologia Assistiva

Segundo Avila (2011), quando se fala sobre o termo tecnologia, logo se pensas em equipamentos como computador, smartphones, *tablets*, rádios, televisões e etc. No entanto, a tecnologia assistiva é a aplicação do conhecimento do homem sobre os artefatos da natureza. Os equipamentos citados anteriormente, são considerados como equipamentos de ponto, desta forma, existe equipamentos simples como caneta mesa também fazem parte do universo da tecnologia. Desta forma existe uma forma de tecnologia que vem sendo utilizada como forma de inclusão social, conhecida como tecnologia assistiva.

O conceito de tecnologia assistiva é ainda muito novo, sendo uma área interdisciplinar, que engloba recursos, metodologias, estratégias entre outas, que tem como objetivo melhorar a qualidade de vida e inclusão social das pessoas. De acordo com Manzini (2005) a tecnologia assistiva é qualquer instrumento utilizado para proporcionar maior facilidade e segurança para uma pessoa, como por exemplo, os óculos utilizados por pessoas com deficiências visuais que tem como objetivo melhorar sua visão.

Com isso, a tecnologia assistiva sendo utilizada como ferramenta de inclusão social proporciona aos indivíduos maior autonomia e independência. E isto se verifica quando Pelosi (2003) relata que a tecnologia assistiva disponibiliza a criação de equipamentos para auxílio na vida cotidiana de pessoas com deficiências visuais (com baixa visão ou quase cegas).

Com este pretexto, teve-se a ideia de criar um aplicativo para facilitar as compras dos indivíduos com deficiência visual, visto que algumas embalagens fornecem informações com letras que até mesmos para pessoas sem deficiência é difícil de se ler.

1.1.2 Comunicação Aumentativa/Alternativa

A comunicação aumentativa/alternativa é uma subárea da tecnologia assistiva, sendo considerada como a integração de símbolos, recursos, técnicas e estratégias com intuito de promover a comunicação. Pode-se dividi-la em dois grupos: as de baixa tecnologia, que tem símbolos gráficos que constituem os estilos de comunicação em papel e as de alta tecnologia, as comunicações com artefatos

tecnológicos computadorizados. A comunicação aumentativa/alternativa tem como intuito proporcionar aso indivíduos diferentes meios de forma a reinventar a comunicação, de forma a atender a necessidade de cada indivíduo. Tal área auxilia três grupos principais de pessoas:

> **Grupo com necessidade de um meio de expressão:** apresentam uma grande diferença na capacidade de compreender e produzir a fala.
>
> **Grupo com necessidade de uma linguagem de apoio:** este grupo subdivide-se em duas partes, sendo a primeira constituída por indivíduos que se utilizam da Comunicação Aumentativa/Alternativa como um passo para a aquisição da linguagem num processo transitório. O outro subgrupo inclui crianças e adultos que, embora tenham aprendido a falar, em determinados momentos não conseguem se fazer compreender.
>
> **Grupo com necessidade de uma linguagem alternativa:** neste caso a Comunicação Aumentativa/Alternativa é vista como um grupo permanente, substituidora definitiva da linguagem falada. As pessoas deste grupo caracterizam-se por usar muito pouco ou por não utilizar a fala no meio de comunicação. (AVILA. 2011, P. 52, 53)

1.1.3 Virtual Vision

O Virtual Vision foi criado a partir da necessidade do Banco Bradesco de tornar seus terminais eletrônicos acessíveis às pessoas com Deficiência Visual. O Bradesco contatou a *Micropower* que o desenvolveu e o tornou o mais eficaz dos leitores de tela existentes no Brasil.

Segundo seu fabricante, “O Virtual *Vision* é a solução definitiva para que pessoas com deficiência visual possam utilizar com autonomia o *Windows*, o *Office*, o *Internet Explorer* e outros aplicativos, através da leitura dos menus e telas desses programas por sintetizador de voz”.

Atualmente o Virtual *Vision* é compatível com todos os sistemas operacionais da *Microsoft*. Desde o *Windows XP* até o *Windows* 10.

1.2 Engenharia de software

Segundo PRESSMAN (2016):

> A engenharia de software abrange um processo, um conjunto de métodos (práticas) e um leque de ferramentas que possibilitam aos profissionais desenvolverem software de altíssima qualidade. A engenharia de software é importante porque nos capacita para o desenvolvimento de sistemas complexos dentro do prazo e com alta qualidade. Ela impõe disciplina a um trabalho que pode se tonar caótico, mas também permite que as pessoas produzam software adaptado à sua abordagem, da maneira mais conveniente

> às suas necessidades. Cria-se software para computadores da mesma forma que qualquer produto bem-sucedido; aplicando-se um processo adaptável e ágil que conduza a um resultado de alta qualidade.

Segundo Padua (2008) um processo, segundo o IEEE, é uma “sequência de passos executados com um determinado objetivo”; segundo o CMMI, é “um conjunto de ações e atividades inter-relacionadas realizadas para obter um conjunto especificado de produtos, resultados ou serviços”.

De acordo com Pressman (2016) processo é um conjunto de atividades, ações e tarefas realizadas na criação de algum artefato. Uma atividade se esforça para atingir um objetivo amplo e é utilizada independentemente do campo de aplicação, do tamanho do projeto, da complexidade dos esforços ou do grau de rigor com que a engenharia de software será aplicada. “Processos ágeis constituem um grupo de metodologias diferentes entre si, mas caracterizadas por princípios comuns, mais baseados no trabalho cooperativo do que no formalismo e na documentação escrita”. (PADUA, 2008, p.102)

1.2.1 Protótipo

Padua (2008) afirma que o protótipo de requisitos é um protótipo visual de baixa fidelidade, que tem por objetivo explorar aspectos críticos dos requisitos de um produto, simulando de forma rápida um pequeno subconjunto da sua funcionalidade. Esse subconjunto não tem por objeto representar fielmente interfaces de usuário real: elas podem ter aparência bastante diferente, desde que sejam funcionalmente equivalentes. Podem até variar quando ao número e nomes dos elementos das interfaces, como telas, painéis e outros agrupamentos, campos e comandos. Aspectos de usabilidade não são tratados no protótipo dos requisitos, mas apenas questões de funcionalidade.

1.2.2 Engenharia de Requisitos

Segundo Sommerville (2013) podemos considerar os requisitos de um sistema como sendo as definições das tarefas que o sistema deve realizar, os serviços oferecem e as limitações de seu funcionamento. Os mesmos refletem as necessidades que os clientes possuem em relação ao sistema, como por exemplo realizar o controle de um estoque de supermercado. O processo de gerenciar esses

requisitos, ou seja, desde a faze de descobri até a faze restrições do sistema é conhecido como engenharia de requisitos. Vale a pena ressaltar que é importantíssimo levantar corretamente os requisitos e delimitar o escopo do trabalho, visto que, muitas das vezes erros relacionados aos requisitos são mais caros de serem consertados do que outros erros.

Os requisitos podem ser divididos em duas categorias básicas: requisitos funcionais e não funcionais.

1.2.2.1 Requisitos funcionais

Podem ser considerados como declarações de serviços que o sistema disponibilizara, de como ele deve se comportar a determinadas entradas específicas, detalhes de suas funções e suas possíveis entradas e saídas, e pode haver casos que eles descrevem o que o sistema não irá disponibilizar. Pode-se ressaltar que “eles dependem do tipo de software a ser desenvolvido, de quem são seus usuários e da abordagem geral adotada pela organização ao escrever os requisitos” (Sommervile. 2013, p.59).

1.2.2.2 Requisitos não funcionais

Segundo Sommerville (2013) são as limitações dos serviços ou ações que o sistema irá desempenhar, que geralmente é aplicado ao sistema como um todo. As limitações podem ser como por exemplo, restrições de timing, no processo de desenvolvimento do produto ou até mesmo restrições que são impostas pelas normas da própria empresa. Tais requisitos, originam-se das necessidades dos clientes sejam através das limitações de orçamentos, necessidade de integrar os novos sistemas com outros sistemas do usuário, legislações de privacidade, etc.

Geralmente eles podem estar relacionados ás propriedades emergentes dos sistemas, alguns exemplos citaremos a seguir:

Usabilidade: Ao pensa-se em usabilidade, tem-se como objetivo garantir que o sistema que está sendo desenvolvido seja fácil de ser utilizado e agradável para qualquer ação que o usuário for desenvolver com o mesmo. Desta forma, pode-se considerar a usabilidade como sendo a medição do nível de desempenho e o de satisfação do usuário. Que pode ser expressar em termos da **facilidade de aprender**

que é o tempo e esforço mínimo exigido ao usuário para alcançar um nível satisfatório de desempenho no uso do sistema e **facilidade de uso** sendo à velocidade na execução de tarefas e à redução de erros no sistema.

Confiabilidade: A confiabilidade pode ser quantificada em um número, ou seja, é a probabilidade de o software não causar uma falha durante um período de tempo pré-definido e sob uma determinada condição. Mas não é simples relacionar a disponibilidade de um sistema sobre uma determinada falha especifica, depende de vários fatores como por exemplo o tempo que o sistema gasta para voltar ao seu estado natural após uma falha. Por isso existem algumas métricas que são utilizadas na tentativa de avaliar a confiabilidade de um sistema, sendo algumas delas: **disponibilidade** medida utilizada para medir o quanto o sistema está disponível para uso**. Taxa de ocorrência de falha** medida utilizada para quantificar a frequência de falha que ocorre no sistema ao tentar executar uma solicitação realizada pelo usuário. **Probabilidade a falha** medida utilizada para quantificar o quanto o sistema irá se comporta de maneira inesperada. **Tempo de ocorrência de falha** medida utilizada para calcular o tempo desde a última falha.

Desempenho: Os requisitos de desempenho podem ser divididos em termos de tempo e espaço. Em termo de tempo pode-se subdividi-los em **tempo de resposta** que especificam o tempo de resposta do sistema que seja tolerável para os usuários e **tempo de processamento** que especifica a número de dados que deveria ser processado em período de tempo para que o sistema de uma resposta. Em relação desempenho de espaço, pode-se considerar o quanto de memória o sistema necessitara para executar uma determinada aplicação.

Manutenibilidade: A manutenibilidade não se restringe somente a manutenção de erros após o sistema ter tido como acabado. O termo manutenibilidade engloba tanto atividades de reparos em certas funcionalidades do sistema com erro quanto atividades de alteração/evolução do sistema com a inclusão ou exclusão de funcionalidades durante o decorrer do projeto.

1.2.3 Diagrama de Caso de Uso

As definições apresentadas a seguir terão como base os seguintes autores Wilson de Padula, Sommerville, Presmam e UML.

O diagrama de caso de uso é utilizado para delinear os requisitos funcionais do sistema. Tais requisitos são escritos em função de atores, casos de usos e relacionamentos.

1.2.3.1 Atores

Os não são parte do sistema, eles representam, papéis que são desempenhados por usuários, dispositivos, hardware ou até mesmos outros softwares que de alguma forma poderão utilizar os serviços e funções oferecidas pelo sistema. Utiliza-se ícones humanos para representar os atores do sistema.

1.2.3.2 Casos de Usos

Os casos de usos modelam o diálogo entre ator e o sistema, ou seja, uma sequência de ações que são executadas pelo sistema, produzindo um resultado para um determinado ator. Cada Caso de Uso descreve uma aplicação ou uso completo do *software.*

As especificações das funcionalidades de um sistema utilizando o diagrama de casos de usos permitirá uma visão mais abrangente das aplicações do sistema para a equipe de desenvolvimento proporcionando um levantamento mais completo dos requisitos.

1.2.3.3 Relacionamentos

Os relacionamentos podem acontecer entre atores e casos de usos e entre casos de usos.

O relacionamento de <<*include*>> e <<extends>>: no sistema pode acontecer de alguns casos de usos utilizar pedaços de pequenas funcionalidades. Com isso, pode-se colocar essas funcionalidades separadas em outro caso de uso, evitando que a mesma seja documentada em todos esses casos de usos que precisa dela. Desta forma, para que esses casos de uso a acesse, realiza-se o relacionamento com o novo caso de uso criado. O relacionamento de <<*include*>> é utilizado para indicar comportamento de obrigatoriedade, como por exemplo, em um sistema de biblioteca sempre que for deletar um usuário é necessário busca-lo. O relacionamento de <<*extends*>> é utilizado para indicar comportamento opcional, como por exemplo, ao buscar um usuário no sistema de biblioteca citado anteriormente pode-se ou não altera-lo ou exclui-lo.

O relacionamento de generalização entre atores: representa uma relação conceitual entre atores do sistema indicando que um determinado ator especial com funcionalidades especificas herdar todas as funcionalidades de um ator mais genérico, como por exemplo, em um sistema o de biblioteca o administrador pode herda todas as funcionalidades do ator usuário.

O relacionamento de generalização entre casos de usos: assim como na generalização entre os atores, elas podem ser utilizadas também para casos de usos, ou seja, um caso de uso especifico pode herda de um caso de uso mais geral.

1.2.4 Levantamento de requisitos por Entrevista

A técnica de levantamento de requisitos por entrevista é bastante fácil de ser utilizada e também muito eficiente, isto porque as informações são obtidos diretamente do cliente e demais interessados no sistema. Tal técnica pode ser dividida em entrevistas formais onde as perguntas são respondidas em sequência e sem perder o foco do assunto, e entrevistas informais que possuem uma lista de questões pré-definidas, o diferencial é que o cliente possui mais espaço para prestar maior esclarecimento das perguntas. No decorrer da entrevista é importante que se registre o que está acontecendo seja através de um gravador ou até mesmo com anotações para que após a entrevista seja possível gerar um relatório relatando tudo que aconteceu. Apesar de ser uma técnica fácil de ser utilizada é necessário que o entrevistador planeje antes de realizar a entrevista, para obter um melhor desempenho. (SILVA, 2012)

1.2.5 Metodologia de Desenvolvimento

Antigamente os softwares eram desenvolvidos sem planejamento e sem padronização, vindo a produzir um produto de má qualidade ao cliente. Com o passar do tempo, as equipes viram a necessidade de se planejar e padronizar o processo de desenvolvimento, surgindo então as metodologias de desenvolvimento. A mesma pode ser dividida em metodologias tradicionais e ágeis.

1.2.5.1 Metodologias Tradicionais

Na metodologia tradicional se dá grande ênfase nas documentações e por isso é conhecida como metodologia pesada. Neste tipo de metodologia, possui como

característica dividir os processos de desenvolvimento em etapas. Apresentaremos as duas mais comuns sendo elas:

Cascata: No modelo cascata a equipe de desenvolvimento não passa para uma nova tarefa sem terminar a tarefa anterior, este tipo de metodologia seque as seguintes etapas: engenharia do sistema, análise de requisitos, geração de código, testes e manutenção. Pode-se observar que em cada uma destas etapas citadas anteriormente possui uma sequência de atividades pré-estabelecidas que são executadas gerando os artefatos, onde os mesmos serviram de bases para as próximas fases. (FERNANDES,2011)

Incremental: pode-se considerar a metodologia incremental como sendo uma melhora da metodologia Cascata. Nesta metodologia, tem como objetivo minimizar o retrabalho durante o desenvolvimento do projeto, proporcionando ao cliente especificar as tarefas com maior prioridade e depois definir novas entregas com funcionalidades com menor prioridade. Podemos ressaltar também que se pode acrescentar novas funcionalidades, onde elas serão desenvolvidas através de um incremento que descreve o modelo Cascata. (SOUZA, 2014)

1.2.5.2 Metodologias Ágeis

Em relação as metodologias ágeis podem-se ressaltar que ela presa enfoques nas pessoas ao invés de processos e que elas não tentam prever o que o poderá ocorrer na fase de desenvolvimento do sistema e sim adaptar a novos fatores que pode ocorrer. Outra curiosidade é que ela presa gastar maior tempo com a implementação do sistema do que gastar maior tempo com a documentação do sistema.

Scrum: na metodologia *SCRUM* todos os projetos são divididos em ciclos conhecidos como *Sprints*. Todas as vezes antes de se começar uma determinada *Sprint* é realizada uma reunião com todos os membros de desenvolvimento no qual são escolhidos todos os itens que serão implementados e onde a equipe a equipe planeja a melhor forma e data para o cumprimento do ciclo. No *Scrum* é realizada uma reunião de curta duração onde é exposto as dificuldades, se está ocorrendo tudo bem e o que tem que melhorar para o dia seguinte. (MOURA et al,2013)

O processo do *SCRUM* é provido de duas subfases: sendo a primeira o planejamento onde é realizada na definição do sistema contendo todos os requisitos conhecidos baseados na lista de *blacklog;* e fase de desenho, na qual é desenvolvido

e aperfeiçoado o desenho do sistema com base na lista de *blacklog.* Todos os requisitos do sistema são realizados um a um no Sprint do blcklog. (FERNANDES, 2011).

***Extreme Programming* (XP):** Com relação ao *XP* podemos notar que o processo de desenvolvimento é realizado com curtas interações, realizando pequenas entregas e rápidas realimentações ao cliente, tendo uma interação maior com o cliente, constante comunicação e coordenação, refatoração contínua, integração e testes contínuos, posse coletiva do código e programação aos pares. Na figura 1 pode-se visualizar algumas das práticas da metodologia *XP*. (MOURA et al,2013)

Práticas do XP	Definições
Planejamento	A equipe prioriza o necessário para ser feito no projeto, baseando-se em requisitos atuais e não futuros para o desenvolvimento do *software*.
Projeto simples	Para cada novo requisito há uma atualização, permitindo a análise e validação sempre que ocorra uma nova especificação do cliente.
Código padrão	Padronização deve ser definida antes de iniciar o desenvolvimento devendo ser seguida por toda equipe.
Padrão de codificação	Elaboração de procedimentos nos quais cada profissional envolvido fará a descrição de como o *software* funcionará, gerando uma linguagem comum entre os envolvidos.
Semanas de 40 horas	Não sobrecarregar as pessoas, e não tornar o trabalho cansativo.
Integração contínua	Permite interagir e construir o sistema de *software* várias vezes ao dia, mantendo os programadores em sintonia, além de possibilitar processos rápidos.
Cliente no local	Visa sanar todas as dúvidas dos elementos do projeto, evitando atrasos ou até entregas erradas.
Metáfora	Requisitos futuros só deverão ser implementados se realmente existirem.
Propriedade coletiva	Caso alguém se desintegre do grupo, o restante do grupo conseguirá prosseguir o trabalho com poucas dificuldades, uma vez que todos conhecem as partes do *software*.
Testes	Primeiramente os programadores realizam testes para depois desenvolver o *software*.
Codificação por testes	Em todo o processo de desenvolvimento são realizados testes de forma validar o processo.
Reestruturação	Objetiva apurar o projeto do *software* estando presente em todas as fases do desenvolvimento. Devendo ser feita sempre que possível

Figura 1 - Práticas do XP Fonte: (Moura et al, 2013)

1.3 Arquitetura de software

Segundo MENDES (2002, pg 6):

> Arquitetura de software é o estudo da organização global dos sistemas de software bem como do relacionamento entre subsistemas e componentes. Desde suas origens, quando descrições qualitativas de organizações de sistemas eram consideradas uteis, a arquitetura de software tem amadurecido ao longo da última década, buscando englobar e explorar notações, ferramentas e técnicas de arquitetura de software, com trabalhos sobre reuso com base em componentes, projeto de software, classes de componentes, analise de programas e linhas de produto.

Utilizando arquitetura de software em seus projetos podem trazer alguns benefícios:

- Suporte ao reuso
- Analise da consistência e da dependência.
- Estrutura aos requisitos de sistema

1.4 **Android**

"O *Android* é a nova plataforma de desenvolvimento para aplicativos móveis como *smartphones* e contêm um sistema operacional baseado em *Linux* " (LECHETA, 2013, pg. 23).

Segundo DEITEL (2015):

> Os aplicativos *Android* são desenvolvidos com Java- uma das linguagens de programação mais usadas do mundo. Essa linguagem foi uma escolha lógica para a plataforma *Android*, pois é poderosa, gratuita, de código-fonte aberto e milhões de desenvolvedores já a conhecem. Os programadores Java experientes podem se aprofundar rapidamente no desenvolvimento com *Android*, usando as APIs (interfaces de programação de aplicativo) *Android* do *Google* e de outros.

Na Figura 2, podemos ver a arquitetura do *Android* que é dividida em:

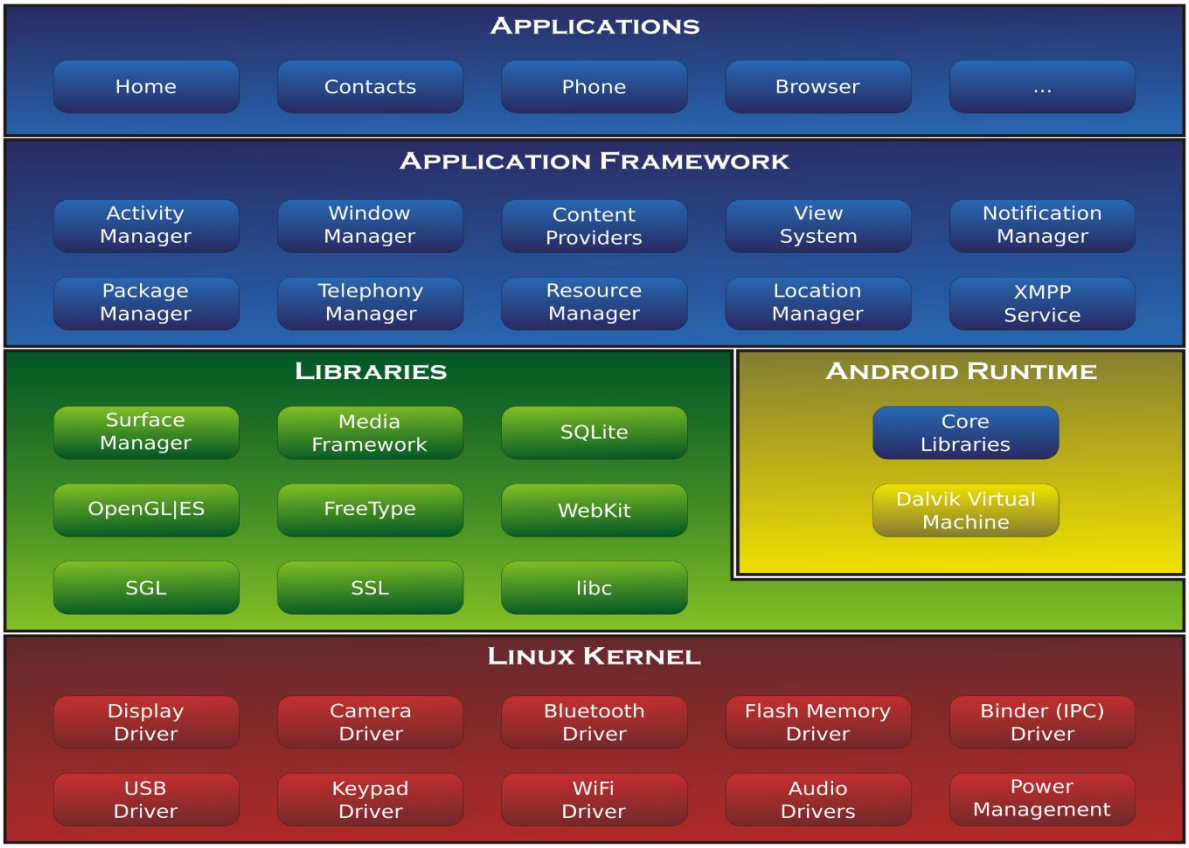


Figura 2: Arquitetura do sistema operacional Android. Fonte: http://developer.andoid.com/.

- *Applications* – onde são encontrados todos os aplicativos fundamentais do *Android*.

- *Application Frameworks* – são todas as APIs e os recursos utilizados pelo aplicativo.
- *Libraries* – carrega um conjunto de bibliotecas C/C++ utilizados pelo sistema.
- *Linux Kernel* – atua como uma camada de abstração entre o software é o resto da pilha de software.

1.4.1 Android Studio

De acordo com LECHETA (2013) o *Android Studio* é a IDE oficial de desenvolvimento para *Android*. O *Android Studio* foi anunciado no Google I/O 2013 e é baseado no *InteliJ IDEA* da *JetBrains*.

Suas vantagens são:

- Editor visual mais fluido e com mais opções.
- Sistema de build mais moderno baseado.
- Diversas utilidades e facilidades ao desenvolver para *Android*.
- Atualizações e melhorias frequentes.

Segundo LECHETA (2013):

> Uma grande diferença entre o Eclipse e o *Android* é o processo de compilação dos projetos. No Eclipse cada projeto é compilado do jeito clássico, como qualquer projeto Java dentro do Eclipse. Mas no *Android* Studio a compilação é feita pelo *Gradle*, que é um moderno sistema de builds. Segundo o site oficial do *Gradle*(gradle.org), ele é definido com a seguinte frase: "*Gradle* combina o poder e a flexibilidade do *Ant* com o gerenciamento de dependência e convenções do *Maven*, em uma maneira eficaz".

A versão mais atual do *Android Studio* é a 2.2.2 que foi lançada em outubro de 2016. Segundo a *Google* (https://developer.android.com) os novos recursos da nova versão são:

- *Samples Browser*
- *Instant Run Improvements*
- *APK Analyzer*
- *Build cache (Experimental)*
- *Virtual Sensors in the Android Emulator*
- *GPU Debugger (Beta)*
- *Espresso Test Recorder (Beta)*

1.4.2 SQLite

De acordo com LECHETA (2013) o *Android* tem suporte nativo ao *SQLite*, um leve e poderoso banco de dados. O *SQLite* é um banco de dados relacional *open-source* e fornece suporte para comandos *SQL*.

1.4.3 Activity

A classe *activity* pode ser considerada como sendo uma tela de aplicação que tem como objetivos gerenciar os eventos e selecionar a *View* que será responsável por projetar a interface gráfica do usuário. É importante lembrar que a cada nova tela implementada no aplicativo é necessário a criação de uma nova classe *activity,* possuindo um ciclo de vida. Onde segundo Lecheta (2013, p. 99) “Uma *activity* tem um ciclo de vida bem definido. Cada *activity* iniciada é inserida no topo de uma pilha, chamada de *activity stack*.”

1.4.4 View

É considerada como sendo a classe mãe dos elementos visuais de uma aplicação em *Android*. A mesma é utilizada por todos os componentes gráficos, sendo necessários que seus filhos implementem o método *onDraw*() de forma a conseguir realizar desenhos na interface gráfica. Tal classe possui vários métodos de forma a facilitar a criação de interfaces de maneira mais simples e que proporcionará maior facilidade de interação entre o usuário e o aplicativo.

1.4.5 Fragments

Algumas pessoas acreditam que os *fragments* são apenas para dividir telas, no entanto seu objetivo e muito mais além, sendo um componente de código reutilizável, que é responsável pela criação da própria *view*, gerenciando seu próprio conteúdo e seus eventos. Os *fragments* são amplamente utilizados para organizar e aproveitar o máximo de espaço da tela do smartphone ou *tablets*, de forma em uma única *view* pode haver vários *fragments*.

1.4.6 Material Design

O Material Design chegou ao *Android* a partir da versão 5.0(*Lollipop*).

Segundo o LECHETA (2015):

> O Material Design é um guia completo sobre como implementar o visual, animações e interação entre os componentes de um layout, considerando que o *Android* se tornou uma plataforma comum para vários dispositivos, como smartphone e *tablets* (*Android*), wearables (*Android Wear*), óculos (*Google Glass*), TVs (*Android* TV) e carros (*Android* Auto).

O Material *Design* foi um esforço da *Google* de padronizar um guia completo de design para nos auxiliar nessa tarefa.

1.4.7 *Intents*

Uma *intent* está presente em todos os ambientes e pode ser considerada como uma mensagem (conhecida como *broadcast*) que é enviada da aplicação para o sistema operacional, requerendo que uma determinada tarefa seja realizada, vale a pena ressaltar que a *indent* tem um papel fundamental na arquitetura do *Android*, realizando as integrações de diferentes aplicações. E de total responsabilidade do sistema operacional interpretar essa mensagem e tomar as ações necessárias, que pode ser simplesmente em realizar uma ligação para determinado número ou até mesmo abrir o *browser* em uma determinada página na internet. LECHETA (2013)

1.4.8 Threads

Em sistemas operacionais tradicionais, cada processo tem um espaço de endereçamento e um único thread de controle. Contudo, frequentemente há situações em que e desejável ter múltiplos *threads* de controle para executando em quase-paralelo. (TANENBAUM, 2003).

Quando um aplicativo Android e aberto um processo dedicado no sistema operacional é criado para executá-lo. Cada processo tem um único *thread*, é conhecida como *Main Thread*. Que e responsável por gerenciar todos os eventos da tela. Quando uma *activity* executa uma tarefa mais longa como uma requisição no *Web Service* e recomendado utilizar uma nova *Thread*.

1.5 **Web Service**

Umas das novas tecnologias que está sendo amplamente utilizada em nosso cotidiano é o *Web Service*. A mesma é utilizada para realizar a integração e comunicação entre diferentes aplicações, com objetivos de proporcionar a ambas aplicações solicitações de informações entre elas quando necessário. Essas comunicações podem ser proceder de diferentes formatos, sendo que os dois tipos mais comuns que são utilizados no dia-a-dia *XML* e *JSON*. Uma das grandes ao se construir um *Web Service* é que seus usuários podem utilizar os arquivos de forma padronizada e sem se preocupar com a linguagem de programação que está sendo utilizada, ou seja, ele é independente da linguagem de programação. (LECHETA,2015)

No entanto, quando se pensa na utilização de um Web Service para dispositivos móveis deve-se lembrar que eles não possuem o mesmo poder de processamento de um computador, não sendo capaz de executar as mesmas bibliotecas. Uma alternativa para solucionar tal problema é utilização de bibliotecas mais leves e realizar a compactação das mesmas de forma a otimizar seu tamanho. LECHETA (2013)

Maven é uma ferramenta de gerenciamento e automação de construção (*build*) de projetos. Forneces diversas funcionalidade através do uso de *plug-ins*.

Segundo LECHETA (2015, p 128):

> Ao adicionar uma dependência ao projeto, o *Maven* faz o download desta dependência diretamente de um repositório mundial de projetos conhecido como *Maven* Central. A grande vantagem é que o *Maven* também faz o *download* e gerencia todas a arvore de dependência para você.

2 Desenvolvimento

Primeiramente vale lembrar que utilizou-se os documentos da Fábrica de Tecnologia Turing (FTT) da Unievangelica, para realizar a documentação do aplicativo APPVIS. Esta documentação encontra-se em anexo.

Ao iniciar-se o desenvolvimento do aplicativo APPVIS devido ao fato dele ser principalmente voltado para pessoas com algum tipo de deficiência visual, e com isso a grande maioria deles não conseguiram visualizar a interface do aplicativo, nos deparou-se com as seguintes dúvidas: de que forma o aplicativo deve interagir com o usuário? Como o usuário conseguirá identificar o ícone do aplicativo dentre tantos que pode existir no seu celular? Como o usuário conseguirá localizar o código de barras em cada produto?

Com isso, na primeira etapa do desenvolvimento, realizou-se uma série de pesquisas na *internet*, livros e artigos procurando autores que abordassem sobre determinado assunto. Após alguns dias de pesquisas sem obter grande sucesso, optou-se então por utilizar a técnica Entrevista de levantamento de requisitos com alguma pessoa que possua deficiência visual. No entanto, não tinha-se contato com nenhuma pessoa com deficiência visual para realizar a entrevista.

Conseguiu-se tal pessoa, ao ter contato com uma fonoaudióloga, onde comentou-se sobre o presente Trabalho de Conclusão de Curso que era voltado para acessibilidade através da criação de um aplicativo destinado a pessoas com algum tipo de deficiência visual e que estava à procura de pessoas com essa característica para realizar uma entrevista, com o objetivo de se retirar algumas dúvidas sobre a forma com que eles utilizavam o seu *smartphone.* Após isso a fonoaudióloga disse que conhecia uma pessoa com essa característica e marcou um local para para que a entrevista pode-se acontecer. Optou-se por identificar tal pessoa pelo seu sobrenome *Massani*.

O senhor *Massani* é deficiente visual com baixo grau de visão desde o seu nascimento, trabalha no Centro Municipal de Apoio ao Deficiente localizado na cidade de Anápolis, onde ministra aulas de Braile.

No primeiro momento explicamos para o senhor Massani com iria funcionar o aplicativo, o segundo momento realizou-se as perguntas. Perguntou-se ao senhor

Massani de que forma ele consegue diferenciar os ícones dos aplicativos existentes no seu *smartphone*? Ele respondeu que consegue diferenciar através da utilização do aplicativo *TalkBack* que já vem instalado para os *smartphone* da plataforma *Android* a partir da versão do *Android* 4.0. No entanto, esse aplicativo vem nos celulares desativados, e para ativá-lo é necessário ir em configurações → Acessibilidade→ *TalkBack* → Ativar.

O TalkBack é um serviço de acessibilidade que ajuda os usuários com deficiência visual ou problemas de visão a interagir com os dispositivos deles.

O TalkBack implementa feedback falado, audível e por vibração ao dispositivo. Esse app vem pré-instalado na maior parte dos dispositivos Android.

Perguntou-se também ao senhor Massani, se ele possui dificuldades em ler as informações dos produtos que ele compra nas lojas? Ele disse que sim, principalmente nos produtos alimentícios e remédios onde as informações vêm escritas em letras minúsculas quase impossíveis para ele ler.

Perguntou-se a ele sobre o que achava da viabilidade do aplicativo? Ele respondeu que acha o aplicativo bastante viável pois proporcionará maior praticidade e facilidade em suas compras, proporcionado a ele maior autonomia e como na maioria das lojas não possui pessoas para auxiliar as pessoas com deficiência visual a realizar sua compra evitaria o constrangimento de ter que ficar parando pessoas desconhecidas ou até mesmo de ficar indo no caixa pedir informações sobre os produtos.

Com essa pesquisa observou-se que em aplicativos convencionais que realiza a leitura dos códigos de barras, é necessário que o usuário focalize o mesmo em um pequeno espaço da tela do seu celular, dificultando para pessoas com deficiência visual. Desta forma, no aplicativo APPVIS o leitor do código de barras realiza a leitura do código de barras em toda a tela do *smartphone*, sem que seja necessário focalizar uma determinada região da tela, facilitando assim para o usuário.

2.1 Arquitetura

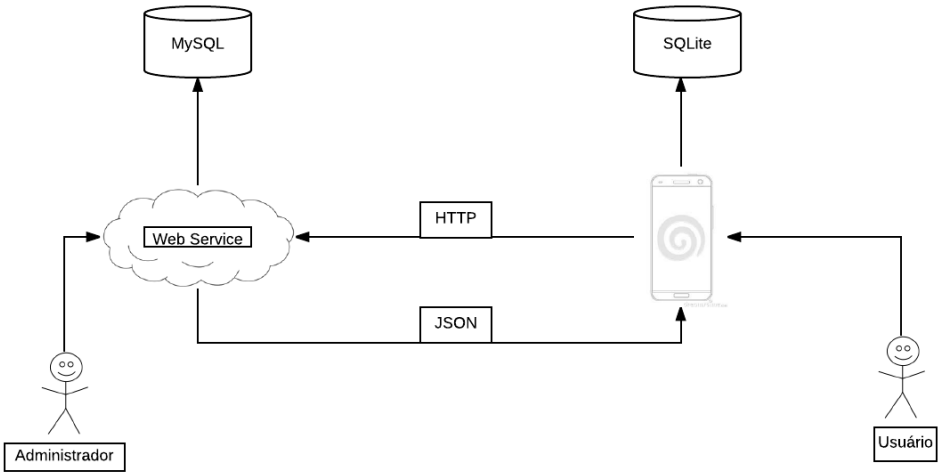


Figura 3 – Arquitetura *Fonte: Autores da pesquisa*

A arquitetura utilizada para construir o APPVIS encontra-se na figura 3. E sobre ela pode-se destacar o seguinte. A arquitetura possui dois bancos sendo que um e local, ou seja, hospedado dentro do próprio aplicativo e o outro em um servidor externo.

Todas as informações inseridas pelo administrador através do Web Service serão armazenadas no banco de dados externo.

Sobre seu funcionamento podemos destacar que: ao usuário abrir o aplicativo, o mesmo irá verificar se o smartphone está conectado na Internet figura 4, caso esteja, será feita uma consulta ao Web Service por meio de uma requisição HTTP. Ao receber essa aquisição o Web Service realizará uma busca no banco de dados e retornará ao aplicativo um arquivo no formato JSON com os produtos armazenados no banco externo. Ao receber esse arquivo JSON, o aplicativo realizara sua conversão para objetos salvando todos os produtos em seu banco local. Caso não esteja conectado em rede de internet o aplicativo realiza somente uma consulta em seu banco de dados local.

No momento em que o usuário *scannear* o código de barra de um determinado produto, será realizada uma busca em seu banco de dados local, retornando as seguintes informações, nome, preço, descrição e fornecedor do produto de forma auditiva figura 5.

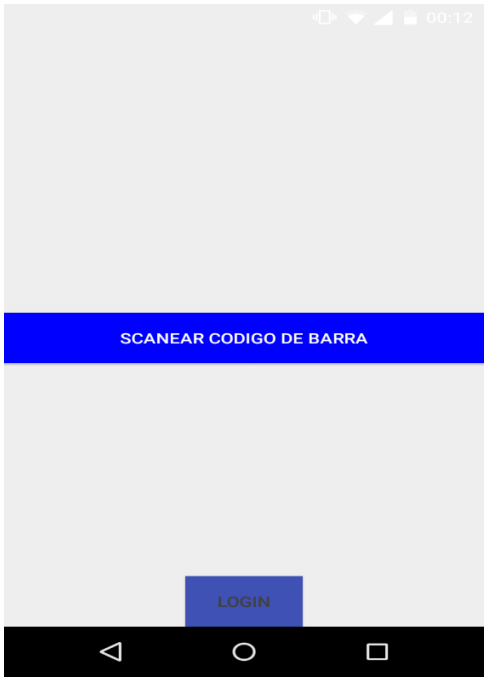


Figura 4 - Tela inicial *Fonte: Autores da pesquisa*

Sobre a arquitetura vale a pena ressaltar que o aplicativo realiza o scanner do código de barras sem que seja necessário que o usuário o focalize em uma pequena região da tela do *smartphone*, ou seja, é feito o scanner em toda a tela.

Figura 5 - Informações do produto *Fonte: Autores da pesquisa*

2.2 Criação do Web Service

O Web Service é o responsável por alimentar o aplicativo *Android* com os produtos, ou seja, é local onde o administrador realizará os cadastros de todos os produtos da loja. Para a criação do *Web Service* foram utilizadas as seguintes ferramentas, *MySQL*, *Maven*, *RESTful* e *API* (*Jersey*). Para a página *web* que vai interagir com o administrador utilizou-se o *AngularJS*.

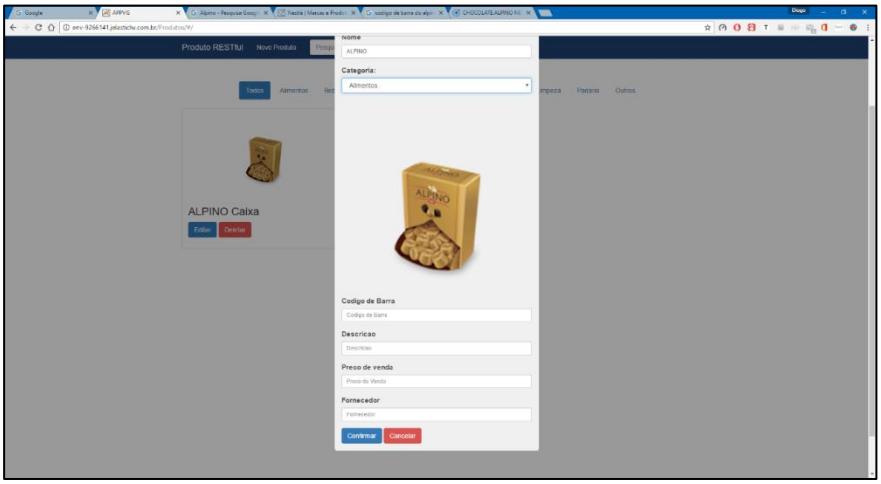


Figura 6 - Cadastrar Produto *Fonte: Autores da pesquisa*

No *Web Service* o administrador poderá realizar as operações de cadastrar, alterar, listar e deletar. Foram também adicionadas as operações de busca pelo nome, id e pela categoria do produto.

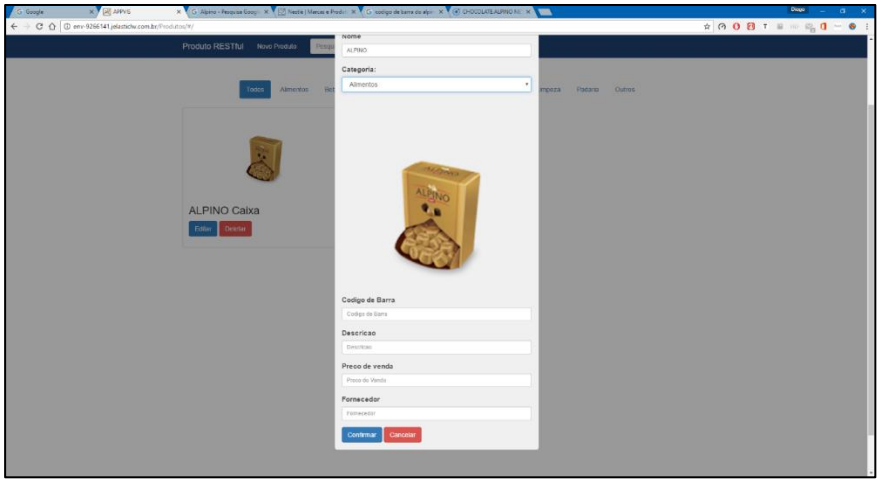


Figura 7 - Editar Produto *Fonte: Autores da pesquisa*

A ferramenta *MySQL* foi utilizada para a criação do banco de dados relacional.

A ferramenta *Maven* foi utilizado para auxiliar no processo de compilação e gerenciamento das dependências do projeto. A grande vantagem de utiliza-la é que a mesma fica responsável de baixar e disponibilizar os arquivos *jar* do seu projeto. O programador deverá somente adicionar a dependência das bibliotecas que deseja atualizar. Fica sobre sua responsabilidade ir no site do *Maven* e escolher as dependências que seu projeto irá necessitar copia-las e cola-las dentro do *pom.xml*. Feito isso o *Maven* realizara os *downloads* dos arquivos *jar* solicitados.

O *Jersey* foi utilizado para ajudar a implementação do *RESTful*, pois é um *framework* bastante utilizado e segue perfeitamente a especificação oficial da *Oracle*.

O *REST* foi utilizado para criar os serviços, que são retornados nos formados *JSON*. Dentre os serviços que o *REST* fornece, foram utilizados no projeto os seguintes: o *GET* utilizado para a consulta de um produto, *POST* utilizado para inserir um produto, *PUT* utilizado para atualizar dados de um produto e *DELETE* para excluir um produto da tabela.

O *RESTful* retorna em arquivo *JSON* todos os produtos pela categoria. O aplicativo vai consumir esse arquivo *JSON* para retornar os produtos. O *AngularJS* foi utilizado pois utiliza uma abordagem mais ligada à sintaxe *HTML* e pelo seu recurso mais focado para *REST*.

2.3 Criação do Aplicativo Android

Para o desenvolvimento do aplicativo, inicialmente criou-se o projeto com o nome de “APPVIS”. Vale a pena ressaltar que o mesmo só funciona a partir das versões do android 4.1(*Jelly Bean*), isto porque somente as mesmas possuem o aplicativo *Google TalkBack*. Na figura 8 pode-se visualizar umas das etapas da criação do projeto.

Durante a implementação observou-se que era necessário criar um arquivo *strings.xml* para realizar as configurações de todos os textos que seriam utilizados no APPVIS. Isto torna o código mais elegante, é uma boa prática de programação, e também através deste arquivo o aplicativo *Google TalkBack* conseguirá reconhecer todos os textos que estão sendo utilizados no APPVIS.

Pode-se destacar três classes importantes que foram implementadas, sendo elas, Produto.java, ProdutoService e ProdutoDB.

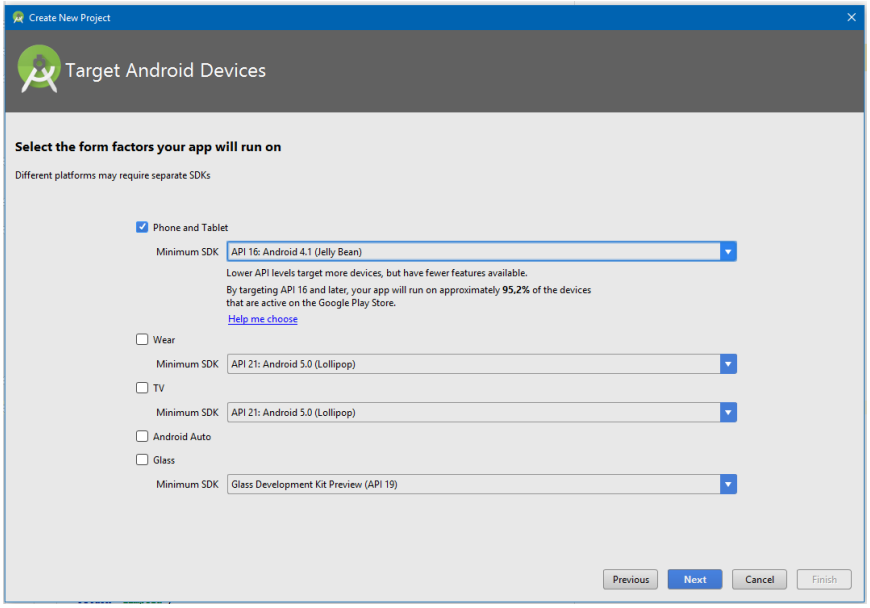


Figura 8 - Requisitos Android *Fonte: Autores da pesquisa*

Com relação a classe Produto.java, ela foi implementada utilizando *Parcelable* ao invés do *Serializable*. Isto por motivo de performance, sendo que a mesma é mais rápida na passagem de parâmetro de objeto. Utilizou-se a seguinte anotação "*@org.parceler.Parcel*", a vantagem de utilizar essa anotação é que a biblioteca *parceler* consegue criar um objeto *Parcelable* em tempo de execução baseado no objeto anotado.

No *ProdutoService* tem-se alguns métodos que são importantes no funcionamento do APPVIS. Um deles e o "*getProdutosFromWebService*" que é o responsável de solicitar o serviço de listar produtos via *Http* para o *Web Service* utilizando o *GET*. Nesta solicitação sempre será necessário passar a *URL* juntamente com a categoria do produto solicitado. Ao receber a solicitação o Web Service retornará um arquivo *JSON*. Ao receber o arquivo *JSON* é realizada a chamada do método *parserJSON*(*context*, *json*) que irá ler o *JSON* e criar uma lista com todos os produtos. Posteriormente é realizada a chamada do método salvarProdutos.

O método salvarProdutos irá receber a lista de produtos, posteriormente abrindo o banco de dados e deletando todos os produtos da categoria passada como parâmetro, inserindo após isto a nova lista de produtos.

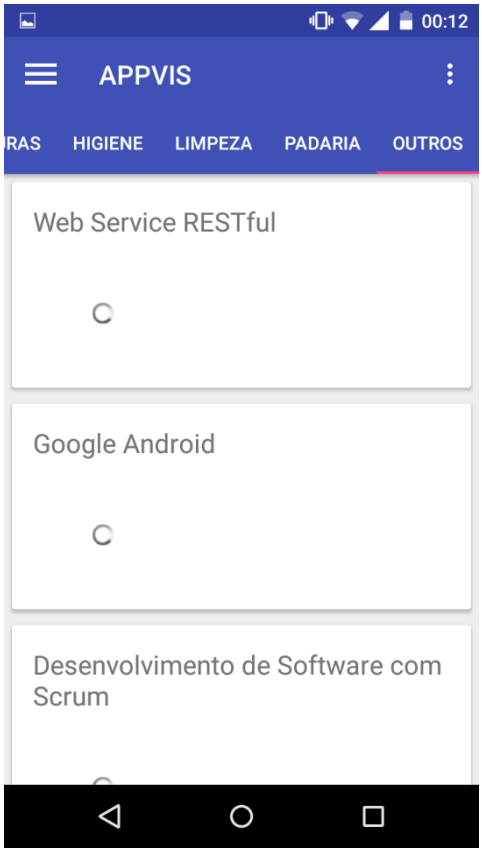


Figura 9 - Lista de Produto *Fonte: Autores da pesquisa*

No método parseJSON e o responsável pela leitura do arquivo *JSON*. Na tabela 9 tem a implementação do método parseJSON.

Na classe ProdutoDB encontra-se os métodos de criação do banco de dados. O método *update* para realizar uma nova versão do banco que são classes obrigatórias do *SQLiteOpenHelper*. Foram também implementadas classes de *save*, delete, *deleteProdutosByCategoria*, *findAll*, *findAllByTipo*, *findAllByNome*, *findAllByCodigoBarra*, *toList*, *execSQL*.

Com relação a leitura do código de barras, optou-se por realizar a adaptação da API desenvolvida pela equipe da Apache, ao invés de começar toda a implementação do zero. Para realizar a leitura da informação do código de barras utilizamos a biblioteca do *Android*: "*TextToSpeech*". Ao reconhece-lo o aplicativo solicitará ao *Web Service* uma busca pelo código de barras, caso positivo, será retornado os seguintes campos: nome, descrição, preço e fornecedor do produto, caso negativo, vai retornar para a página principal do produto.

3 Considerações Finais

Ao analisar a tecnologia computacional pode-se observar que aumentou muito o seu uso pela sociedade, principalmente para automatização de processos de forma a facilitar as atividades que são realizadas no cotidiano das pessoas. Uma aplicação da tecnologia que vem ganhando grande enfoque nos últimos anos é a de Acessibilidade, que tem como objetivo inserir no meio social as pessoas com algum tipo de deficiência.

Neste cenário, vale ressaltar que o uso de aplicativos para *smartphone* é cada vez mais viável nos dias atuais. Podendo ser utilizado como ferramenta de inclusão social de pessoas com deficiência visual, já que basicamente grande parte da população possui pelo menos um *smartphone*.

De um modo geral, utilizando destes aparelhos e de recursos existentes em produtos (código de barra) desenvolvemos um aplicativo que funciona como facilitador para essas pessoas, em que muitos casos não conseguem identificar os produtos que estão comprando. Este aplicativo informa informações como nome, preço, descrição, fornecedor do produto entre outras.

O motivo pelo qual se escolheu este tema é que existem relativamente poucos aplicativos para ajudar o deficiente visual no seu dia-a-dia, e aplicativos que de alguma forma contribua para inclusão das pessoas com algum tipo de deficiência no meio social tem grande possibilidade de fazer sucesso.

Durante o desenvolvimento deste trabalho passamos por inúmeras dificuldades que serviram de motivação para continuarmos com este tema. Sendo que elas agregaram inúmeros conhecimentos para nós seja tanto para o meio profissional quanto para o meio social, onde passamos a crescer como cidadão e, dar maior atenção, ter admiração e respeito pelas pessoas com algum tipo de deficiência. Já que até mesmo para uma tarefa simples que é realizar alguma compra se torna uma tarefa difícil para eles.

Ao realizarmos uma entrevista com o senhor Massani, tivemos convicção que a criação do aplicativo é viável. Pois, podemos ouvir de um deficiente visual que que o aplicativo é bom facilita, para os deficientes visuais realizarem suas compras.

Realizou-se também uma pesquisas, com objetivo de analisar a aplicabilidade do aplicativos em algumas lojas das cidades de Anápolis e de Ouro Verde do estado de Goiás, uma importante observação, sobre as mesmas é que praticamente em toda parte das ruas da cidade meios que facilite a locomoção das pessoas com deficiência visual e com deficiência física. Nesta pesquisa foram entrevistadas alguns funcionários de forma oralmente, de redes de farmácias, supermercados, sorveterias, restaurantes e lanchonetes. Através da mesma pode-se comprovar que todas elas não possuem atendimentos especiais aos deficientes visuais, tornando assim o aplicativo viável para estas redes de lojas, proporcionando a elas se tornarem redes inclusivas dando maior praticidade, autonomia e um diferencial em relação aos seus concorrentes.

Com isso, é possível apontar que essa monografia atende seu objetivo, ou seja, proporciona aos deficientes visuais se inserirem no meio social, trazendo aos mesmos uma maior autonomia e facilidade ajudando-os na realização de compras em lojas, supermercados e etc. que utilizam o APPVIS. Desta forma, em trabalho futuros temos como objetivo realizar melhoras em nosso projeto, com a finalidade de torna-lo possível para comercialização.

REFERÊNCIAS BIBLIOGRÁFICAS

_____**. LEI N° 10.098, DE 19 DE DEZEMBRO DE 2000**. Estabelece normas gerais e critérios básicos para a promoção da acessibilidade das pessoas portadoras de deficiência ou com mobilidade reduzida. Disponível em: http://www.planalto.gov.br/ccivil_03/leis/L10098.htm. Acesso em: 15/04/2016.

APARECIDA, Luciana dos Santos Giareta. **O papel da tecnologia assistiva na inclusão de deficiência visual**. Guarujá – São Paulo: UNAERP – Universidade de Ribeirão Preto (Campus Guarujá), Brasil, 2004. Disponível em: http://www.unaerp.br/documentos/836-o-papel-da-tecnologia-assistiva-na-inclusao-digital-dos-portadores-de-deficiencia-visual/file.

AVILA, Barbara G. **Comunicação Aumentativa e Alternativa para o Desenvolvimento da Oralidade de Pessoas com Autismo**. Disponível em:< https://www.lume.ufrgs.br/bitstream/handle/10183/32307/000785427.pdf?sequence=1>. Acesso: 05/10/2016.

BRASIL. Decreto nº. 3.298 (20/12/1999). Disponível em: <http://www.planalto.gov.br/ccivil_03/decreto/d3298.htm>. Acesso em: 10/05/2016.

DEITEL, Harvey, DEITEL, Paul, DEITEL, Abbey. ***Android* para Programadores**, 2. ed. Bookman, 01/2015. VitalSource Bookshelf Online.

FERNANDES, Matheus. R. **SCRUM E XP:** Um comparativo no processo de desenvolvimento de software. Disponível em: < http://professores.dcc.ufla.br/~terra/publications_files/students/2011_fumec_fernandes.pdf>. Acesso em: 29/09/2016.

FILHO, PADUA, Wilson Paula. **Engenharia de Software** - Fundamentos, Métodos e Padrões, 3ª edição. LTC, 11/2008. VitalSource Bookshelf Online.

FONSECA, J. J. S. **Metodologia da pesquisa científica**. Fortaleza: UEC, 2002.

GUEDES, Gilleanes T. A. UML 2: uma abirdagem prática. 2. Ed. São Paulo: Novatec Editora, 2011.

LECHETA, Ricardo R. **Google Android**: aprenda a criar aplicações para dispositivos móveis com Android SDK. 3.ed. São Paulo: Novatec Editora, 2013.

LECHETA, Ricardo R. **Google Android**: aprenda a criar aplicações para dispositivos móveis com Android SDK. 5.ed. São Paulo: Novatec Editora, 2015.

LECHETA, Ricardo R. **Web Services RESTful**. 1.ed. São Paulo: Novatec Editora, 2015.

MAIOLA, Carolina dos Santos; SILVEIRA, Tatiana dos Santos da. **Deficiência Visual.** Indaial: Grupo UNIASSELVI, 2009.

MANZINI, E. J. **Tecnologia Assistiva para Educação**: recursos pedagógicos adaptados. In: Ensaios pedagógicos: construído escolas inclusivas. Brasília: SEESP/MEC, p. 82-86, 2005.

MENDES, Antônio. **Arquitetura de software: desenvolvimento orientado para arquitetura.** 1.ed. Rio de Janeiro: Campus, 2002.

MONTEIRO, Janete Lopes. **Os Desafios dos cegos nos espaços sócias: um olhar sobre a acessibilidade.** In: SEMINÁRIO DE PESQUISA EM EDUCAÇÃO DA REGIÃO SUL, IX ANPED, 2012, FLORIANÓPOLIS. Anais... FLORIANÓPOLIS: UFSC, 2012. pg. 3-10.

MOURA, Andrea S. D; Oliveira, Carlos. R. R. et al. **Análise das metodologias ágeis xp e scrum no desenvolvimento de um software de gerenciamento de abastecimento.** Disponível em: < http://www.abepro.org.br/biblioteca/enegep2013_TN_STP_184_049_22827.pdf>. Acesso em: 29/09/2016.

ORMELEZI, Eliana Maria. **Os Caminhos da Aquisição do Conhecimento e a Cegueira: do universo do corpo ao universo simbólico**. 2000. 205 f. Dissertação (Mestrado em Psicologia e Educação) – Faculdade de Educação, Universidade de São Paulo, São Paulo.

PELOSI, MiRyan Bonadiu in.: Seminário internacional sociedade inclusiva. PUC Minas, Belo Horizonte: 2003 Anais. P. 183-187.

PRESSMAN, R. S. **Engenharia de software**. 6. ed. São Paulo: Mc Graw Hill, 2006.

PRESSMAN, Roger, MAXIM, Bruce. **Engenharia de Software**, 8th ed. AMGH, 01/2016. VitalSource Bookshelf Online.

SILVA, Samuel. F. B. **ENGENHARIA DE REQUISITOS:** Uma análise das técnicas de levantamento de requisitos. Disponível em: <>. Acesso em: 10/10/2016.

SILVA, Edna Lúcia; MENEZES, Estera Muszkat. **Metodologia da pesquisa e elaboração de dissertação**. 4.ed. ver. Atual. – Florianópolis: UFSC, 2005.

SOUZA, Diogo. R. **Implantação da Metodologia Ágil Scrum em um Ambiente de Desenvolvimento.** Disponível em: < http://professores.dcc.ufla.br/~terra/publications_files/students/2012_fumec_silva.pdf >. Acesso: 03/10/2016.

TANENBAUM, Andrew S. **Sistemas operacionais modernos** / Andrew S. Tanenbaum; tradução Ronaldo A.L. Gonçalves, Luís A. Consularo ; revisão técnica Regina Borges de Araujo. - 2. ed. -- São Paulo: Pearson Prentice Hall, 2003.

VILLELA, Flávia. **IBGE: 6,2% da população têm algum tipo de deficiência.** EBC (Agência Brasil), 2015.

Anexo

Anexo A- Documento de Visão

ENGENHARIA DE COMPUTAÇÃO – FTT

DOCUMENTO DE VISÃO

APPVIS

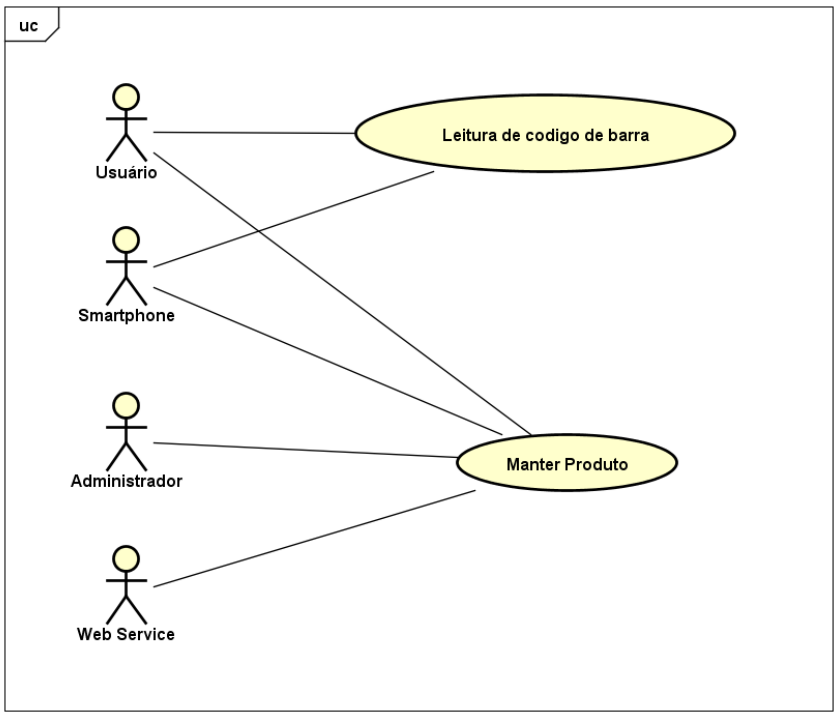
Autores:
Diogo Mendes
Luiz Felipe

Anápolis – GO
2016

1. VISÃO DO PRODUTO

Visto que atualmente os estados das embalagens dos produtos é altamente visual e em algumas delas as informações são escritas em letras tão pequenas que até mesmo pessoas que não possuem deficiência visual tem dificuldades em lê-las, os problemas mais sérios de acessibilidade estão ligados a pessoas com deficiência visual. Tendo como este pretexto, o aplicativo é **destinado** ás pessoas com algum tipo de deficiência visual (pessoas cegas ou com baixa nível de visibilidade). **É um** software mobile desenvolvido na plataforma *Android*, com funcionalidades bem simples e intuitivas, permitindo aos seus usuários uma rápida assimilação ao aplicativo. **Possibilita** ao usuário consultas sobres produtos previamente cadastrados no sistema e para administrador tem a opção de manter produtos (incluir, alterar e excluir) que podem ser realizadas no próprio aplicativo ou em um WebService. **Nosso produto** irá facilitar o processo de fornecimento de informações sobre determinados produtos (previamente cadastrados no sistema) de forma a auxiliar o usuário na tomada de decisão, tais informações sobre os produtos serão fornecidas em forma de áudio.

2. Caso de Uso



3. ABREVIATURAS, CONVENÇÕES E SIGLAS

Todas as siglas, abreviações, convenções e estrangeirismo existem nos documentos do projeto estarão relatados na tabela abaixo.

Siglas	Descrição
Manter	Permite ao ator as ações de alterar, buscar, inserir e excluir.

4. REQUISITOS FUNCIONAIS

4.1 Manter Produto

UC.	Descrição	Status	Prioridade
UC.01	Manter Produto	A Realizar	Essencial
Detalhamento: Este requisito permite ao ator realizar as seguintes ações: cadastrar, alterar, buscar e excluir um determinado produto.			

4.2 Manter Produto

UC.	Descrição	Status	Prioridade
UC.01	Leitura de Código de Barra	A Realizar	Essencial
Detalhamento: Este requisito permite ao ator realizar uma leitura de um código de barra.			

5. REQUISITOS NÃO FUNCIONAIS

5.1 Usabilidade

Descrição	Status	Prioridade
Usabilidade	A Realizar	Essencial
Detalhamento	O sistema realizará a leitura do código de barra aproveitando os recursos da câmera. Exemplo: o foco da câmera, Tela inteira, zoom.	

5.2 Desempenho

Descrição	Status	Prioridade
Desempenho	A Realizar	Essencial
Detalhamento	O sistema utilizara de técnicas e procedimento de forma minimizar o tempo de busca e envio de objetos utilizados pela *Activity,* através da utilização classe *Parcelable*.	

5.3 Confiabilidade

Descrição	Status	Prioridade
Confiabilidade	A Realizar	Essencial
Detalhamento	O sistema utilizara na consulta de produto um elemento que é exclusivamente dele de forma a evitar transpor informações de produtos trocados. Por Exemplo o código de barras é único para cada produto.	

Anexo B- DOCUMENTO DE ESPECIFICAÇÃO DE CASO DE USO MANTER PRODUTO

ENGENHARIA DE COMPUTAÇÃO – FTT

DOCUMENTO DE ESPECIFICAÇÃO DE CASO DE USO

APPVIS

MANTER PRODUTO

Autores:
Diogo Mendes
Luiz Felipe

Anápolis – GO
2016

1. **ATORES**

Ator	Descrição
Administrador	Ator humano que representa o administrador.

2. **PRÉ-CONDIÇÕES**

- O usuário deverá acessar o site do sistema.
- O usuário deverá estar cadastrado na base de dados.

3. **FLUXO DE EVENTOS**

3.1 Fluxo Principal

P1. O sistema disponibiliza a página de lista de Produtos[4.1][A1][A2][A3][A4].
P2. O caso de uso é encerrado.

3.2 Fluxos Alternativos

A1. Novo Produto

P1. O ator seleciona a opção “Novo Produto”.
P2. O sistema redireciona o ator para a página de cadastro de produto[4.2].
P3. O ator preenche os campos e seleciona a opção salvar.
P4. O sistema recebe os dados e cadastrar no banco de dados.
P5. O sistema redireciona o ator ao passo [P1] do fluxo principal.

A2. Editar Produto

P1. O ator seleciona a opção “Editar”.
P2. O sistema redireciona o ator para a página com o formulário de alteração de dados[4.3].
P3. O ator altera o formulário.
P4. O ator seleciona a opção “Salvar”.
P5. O sistema recebe os dados e cadastrar no banco de dados.
P6. O sistema redireciona o ator ao passo [P1] do fluxo principal.

A3. Deletar Produto

P1. O ator seleciona a opção “Deletar”.
P2. O sistema disponibiliza uma mensagem de confirmação de exclusão do produto[4.4].
P3. Caso o ator escolha sim, o sistema redirecionar para o passo P5.
P4. Caso o ator escolha não, o sistema redirecionar para o passo [P1] do fluxo principal.
P5. O sistema deleta o produto, e retorna uma mensagem de produto deletado com sucesso.
P6. O sistema redireciona o ator ao passo [P1] do fluxo principal.

A4. Buscar Produto

P1. O ator digita o nome do produto que deseja buscar e seleciona buscar.
P2. O sistema realiza a busca dentro do banco de dados.
P3. O sistema disponibiliza os produtos solicitados.

4. PROTÓTIPOS

Para estabelecer a definição do “Tipo do Campo”, deverá seguir o seguinte domínio:

- **Campo de texto:** São referidos aos campos do tipo texto, inteiro, data, etc.
- **Área de texto:** Referem-se a campos onde o deverá ser inserido um texto com uma quantidade maior de caracteres.
- **Caixa de seleção:** São referidos a campos do tipo checkbox.
- **Botão de Opção:** Referem-se a campos do tipo radio.
- **Item de Múltipla Seleção:** São referidos a campos do tipo select com seleção múltipla.

4.1. Listar Produtos

Definição dos Campos

Campo	Descrição	Tipo do campo	Tamanho	Editável	Obrigatório	Valores Possíveis	Restrição
Pesquisar Produto	Campo para realizar uma pesquisa por um produto	Campo de texto	100	Sim	Sim	N/A	

Definição dos Comandos

Nome	Ação	Tipo	Restrições
Novo Produto	Apresenta a tela de inclusão de novo produto.	Link	
Editar	Permiti ao usuário editar dados do produto.	Botão	
Deletar	Permiti ao usuário excluir o produto selecionado.	Botão	

Regras de Apresentação

Descrição
Cada produto deverá aparecer dentro da sua categoria ou categoria “todos”.

4.2. Novo Produtos

Definição dos Campos

Campo	Descrição	Tipo do campo	Tamanho	Editável	Obrigatório	Valores Possíveis	Restrição
Nome	Campo para informar o nome do produto.	Campo de texto	100	Sim	Sim	N/A	
Categoria	Campo de seleção da categoria do produto.	Item de Múltipla Seleção		Não	Sim	Alimentos Bebidas Carnes Frios Frutas Higiene Limpeza Padaria Outros	
Código de Barra	Campo para digitar o código de barra do produto.	Campo de texto	50	Sim	Sim	N/A	
Descrição	Campo para informar descrição do produto	Campo de texto	500	Sim	Sim	N/A	
Preço de Venda	Campo para informar o preço do produto.	Campo de texto	20	Sim	Sim	N/A	
Fornecedor	Campo para informar o fornecedor do produto	Campo de texto	100	Sim	Sim	N/A	

Definição dos Comandos

Nome	Ação	Tipo	Restrições
Confirmar	Quando selecionado irá cadastrar um produto no sistema.	Botão	
Cancelar	Quando selecionado irá cancelar o cadastro do produto.	Botão	

Regras de Apresentação

Descrição
Se nenhuma imagem não for selecionada, o produto cadastrar uma imagem padrão previamente selecionada pelo administrador.

4.3. Editar Produtos

Definição dos Campos

Campo	Descrição	Tipo do campo	Tamanho	Editável	Obrigatório	Valores Possíveis	Restrição
Nome	Campo para informar o nome do produto.	Campo de texto	100	Sim	Sim	N/A	
Categoria	Campo de seleção da categoria do produto.	Item de Múltipla Seleção		Não	Sim	Alimentos Bebidas Carnes Frios Frutas Higiene Limpeza Padaria Outros	
Código de Barra	Campo para digitar o código de barra do produto.	Campo de texto	50	Sim	Sim	N/A	
Descrição	Campo para informar descrição do produto	Campo de texto	500	Sim	Sim	N/A	
Preço de Venda	Campo para informar o preço do produto.	Campo de texto	20	Sim	Sim	N/A	
Fornecedor	Campo para informar o fornecedor do produto	Campo de texto	100	Sim	Sim	N/A	

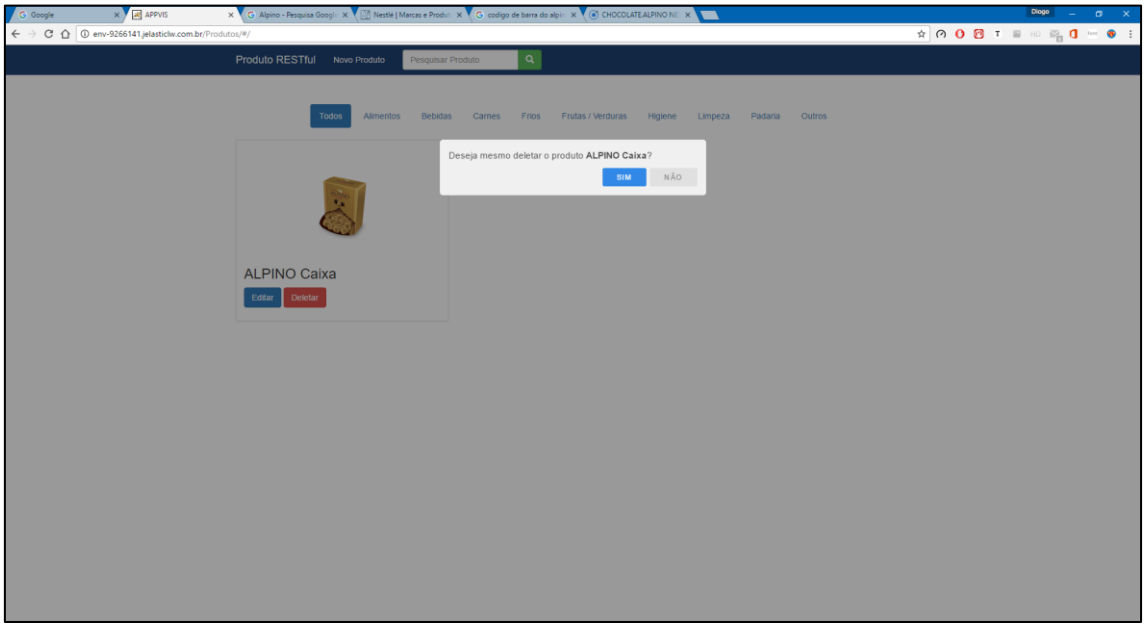
Definição dos Comandos

Nome	Ação	Tipo	Restrições
Confirmar	Quando selecionado irá editar um produto no sistema.	Botão	
Cancelar	Quando selecionado irá cancelar o cadastro do produto.	Botão	

Regras de Apresentação

Descrição
Se nenhuma imagem não for selecionada, o produto cadastrar uma imagem padrão previamente selecionada pelo administrador.

4.4. **Deletar Produtos**



Definição dos Comandos

Nome	Ação	Tipo	Restrições
Sim	Quando selecionado irá deletar um produto no sistema	Botão	
Não	Quando selecionado irá cancelar o delete de um produto no sistema	Botão	

Anexo C- DOCUMENTO DE ESPECIFICAÇÃO DE CASO DE USO LEITURA CODIGO DE BARRA

ENGENHARIA DE COMPUTAÇÃO – FTT

DOCUMENTO DE ESPECIFICAÇÃO DE CASO DE USO

APPVIS

LEITURA CODIGO DE BARRA

Autores:
Diogo Mendes
Luiz Felipe

Anápolis – GO
2016

1. **ATORES**

Ator	Descrição
Usuário	Ator humano que representa o usuário.

2. **PRÉ-CONDIÇÕES**

- O usuário deverá acessar o aplicativo.

3. **FLUXO DE EVENTOS**

3.1 Fluxo Principal

P3. O aplicativo deseja a boa vinda ao usuário e dar dicas de como ser utilizar o aplicativo sempre usando a fala.
P4. O ator seleciona scanear código de barra.
P5. O aplicativo abrir a câmera e ativa várias funcionalidades da câmera[4.1].
P6. Ao detectar o código de barra o sistema buscar os dados do produto do banco através do código e redirecionar o ator com dados do produto.
P7. O ator ouvi informação relevantes do produto.
P8. O caso de uso e encerrado.

4. **PROTÓTIPOS**

Para estabelecer a definição do “Tipo do Campo”, deverá seguir o seguinte domínio:

- **Campo de texto:** São referidos aos campos do tipo texto, inteiro, data, etc.
- **Área de texto:** Referem-se a campos onde o deverá ser inserido um texto com uma quantidade maior de caracteres.
- **Caixa de seleção:** São referidos a campos do tipo checkbox.
- **Botão de Opção:** Referem-se a campos do tipo radio.
- **Item de Múltipla Seleção:** São referidos a campos do tipo select com seleção múltipla.

4.5. **Ativar Câmera**