UNIVERSIDADE FEDERAL DO PARANÁ

MARCELO MARZOLA LEITE

CONTENT: SISTEMA PARA GERENCIAMENTOS DE FEEDS

CURITIBA

2019

MARCELO MARZOLA LEITE

CONTENT: SISTEMA PARA GERENCIAMENTOS DE FEEDS

Monografia apresentada como requisito parcial à obtenção do grau de Especialista em Engenharia de Software, no Curso de Pós-Graduação em Engenharia de Software, Setor de Educação Profissional e Tecnológica, da Universidade Federal do Paraná.

Orientadora: Profa. Dra. Rafaela Mantovani Fontana

CURITIBA

2019

TERMO DE APROVAÇÃO

MARCELO MARZOLA LEITE

CONTEÚDO: AGREGADOR DE FEEDS E CONTEÚDOS

Monografia apresentada como requisito parcial à obtenção do grau de Especialista no Curso de Engenharia de Software, Setor de Educação Profissional e Tecnológica, da Universidade Federal do Paraná.

Orientador: Profª Dra. Rafaela Mantovani Fontana
Setor de Educação Profissional e Tecnológica, da Universidade Federal do Paraná.

Curitiba, 11 de novembro de 2019.

MINISTÉRIO DA EDUCAÇÃO
SETOR DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
UNIVERSIDADE FEDERAL DO PARANÁ
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
CURSO DE PÓS-GRADUAÇÃO ENGENHARIA DE SOFTWARE - 40001016231E1

TERMO DE APROVAÇÃO

Os membros da Banca Examinadora designada pelo Colegiado do Programa de Pós-Graduação em ENGENHARIA DE SOFTWARE da Universidade Federal do Paraná foram convocados para realizar a arguição da Monografia de Especialização de **MARCELO MARZOLA LEITE** intitulada: **CONTENT: SISTEMA PARA GERENCIAMENTOS DE FEEDS**, que após terem inquirido o aluno e realizada a avaliação do trabalho, são de parecer pela sua APROVAÇÃO no rito de defesa.
A outorga do título de especialista está sujeita à homologação pelo colegiado, ao atendimento de todas as indicações e correções solicitadas pela banca e ao pleno atendimento das demandas regimentais do Programa de Pós-Graduação.

Curitiba, 18 de Novembro de 2019.

RAFAELA MANTOVANI FONTANA
Presidente da Banca Examinadora (UNIVERSIDADE FEDERAL DO PARANÁ)

JAIME WOJCIECHOWSKI
Avaliador Interno (UNIVERSIDADE FEDERAL DO PARANÁ)

R. Alcides Vieira Arcoverde, 1225 - Curitiba - Paraná - Brasil
CEP 80035-455 - Tel: (41) 3361-4904 - E-mail: jaimewo@ufpr.br

Agradeço primeiramente minha
companheira e minha filha, pela paciência,
compreensão e carinho

AGRADECIMENTOS

Agradeço a todos que me ajudaram e dedicaram seu tempo para que eu conseguisse realizar este trabalho. Também fico agradecido aos professores por todo o conhecimento compartilhado e ajuda nos momentos que foi necessário. Em especial, agradeço minha orientadora, Rafaela Mantovani Fontana por todo apoio, paciência e otimismo.

Também sou grato a minha família pela compreensão e carinho em todos os momentos que precisei me ausentar. Foram momentos essenciais e que ter o apoio de vocês me encorajou a persistir.

It is not enough for code to work.
(Robert C. Martin, 2008)

RESUMO

A maioria dos agregadores de feeds disponíveis nas lojas de aplicativos para celular focam em conteúdos de áudio ou texto. São poucos os que enfatizam os dois tipos de conteúdo com o mesmo cuidado. O objetivo do aplicativo desenvolvido é servir como opção a outros agregadores de feeds na organização e consumos dos dois tipos de conteúdo, áudio e texto. O sistema foi desenvolvido para o sistema operacional Android, utilizado em smartphones e tablets. Para organização do desenvolvimento e versionamento, foram empregadas métodos ágeis e a ferramenta Git. O aplicativo que resultou possui uma interface simples, que possibilita tanto a gestão de feeds com conteúdos escritos ou áudios. Também permite ao usuário salvar conteúdos para leitura posterior.

Palavras-chave: feeds, podcast, agregadores.

ABSTRACT

Most feed aggregators available at mobile app stores focus on audio or text content. There are few who emphasize both types of content with the same care. The purpose of the application developed is to serve as an option to other feed aggregators in the organization and consumptions of both content, audio and text. The system was developed for the Android operating system, used on smartphones and tablets. For development organization and versioning, agile methods and the Git tool were employed. The resulting application has a simple interface that enables both the management of feeds with written or audio content. It also allows the user to save content for later reading.

Key-words: feeds, podcast, aggregators.

LISTA DE ILUSTRAÇÕES

FIGURA 1 – CICLO DE SPRINTS NO SCRUM 26
FIGURA 2 – QUADRO KANBAN TRELLO 32
FIGURA 3 – MARCAÇÃO DE TEMPO TOGGL 32
FIGURA 4 – ANDROID BLUEPRINTS MVP+CLEAN ARCHITECTURE 33
FIGURA 5 – TELA PARA LISTAR FEEDS 36
FIGURA 6 – TELA PARA ADICIONAR FEED 37
FIGURA 7 – TELA PARA MOSTRAR FEED 38
FIGURA 8 – TELA DE MENU PRINCIPAL 39
FIGURA 9 – TELA PARA LISTAR CONTEÚDOS 40
FIGURA 10 – MENU PARA CONTEÚDOS 41
FIGURA 11 – MENU PARA FILTRAR CONTEÚDOS 42
FIGURA 12 – TELA PARA MOSTRAR CONTEÚDO 43
FIGURA 13 – TELA PARA MOSTRAR CONTEÚDO - PODCAST 44
QUADRO 1 – SEPARAÇÃO DAS HISTÓRIAS EM SPRINTS 27

LISTA DE TABELAS

TABELA 1 – DIVISÃO DE ESFORÇO PARA AS HISTÓRIAS 28
TABELA 2 – PLANEJAMENTO DAS SPRINTS 28

LISTA DE ABREVIATURAS E SIGLAS

RDF	-	Resource Description Framework
W3C	-	World Wide Web Consortium
WWW	-	World Wide Web
XML	-	Extensible Markup Language
ATOM	-	Atom Syndication Format
IETF	-	Internet Engineering Task Force

LISTA DE SÍMBOLOS

@ - arroba

SUMÁRIO

1 INTRODUÇÃO 14
1.1 PROBLEMA 14
1.2 OBJETIVO GERAL 15
1.3 OBJETIVOS ESPECÍFICOS 15
1.4 JUSTIFICATIVA 16
2 FUNDAMENTAÇÃO TEÓRICA 17
2.1 A WEB 17
2.2 WEB SEMÂNTICA 18
2.3 RSS 19
2.4 ATOM 20
2.5 FEED 20
2.6 PODCAST 21
2.7 AGREGADORES DE FEEDS 22
2.7.1 Read 22
2.7.2 RSS Reader 23
3 MÉTODOS E MATERIAIS 24
3.1 SCRUM 25
3.1 SPRINTS 25
3.2.1 SPRINT 1 27
3.2.2 SPRINT 2 28
3.2.3 SPRINT 3 28
3.2.4 SPRINT 4 29
3.2.4 SPRINT 5 29
3.2 ORGANIZAÇÃO 29
3.3 TECNOLOGIAS 31
4 APRESENTAÇÃO DOS RESULTADOS 34
5 CONSIDERAÇÕES FINAIS 44
5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS 45
REFERÊNCIAS 46
APÊNDICES 50

1 INTRODUÇÃO

A preocupação em como organizar e acessar informações publicadas na internet existe desde seu surgimento. Em 1989 Tim Berners-Lee propôs o que ele chamou de World Wide Web (WWW ou web). Um método para computadores distribuírem informações entre diferentes pontos enviadas por diferentes emissores. O objetivo de Berners-Lee era como lidar com a perda de informação que ele observava em seu trabalho no CERN - European Organization for Nuclear Research (BERNERS-LEE, 1989).

Na segunda metade dos anos 90 começaram a surgir algumas iniciativas com o objetivo de definir modelos para disponibilização de conteúdos na rede. A partir destas tentativas surgiu o padrão RSS, Really Simple Syndication, que com o passar do tempo passou a ser utilizado principalmente por produtores de conteúdos periódicos, como blogs e portais de notícias (BARAK. 2004).

O RSS passou por diversas versões ao longo dos anos e se consolidou como o principal formato adotado na disponibilização de conteúdos informativos e de entretenimento. A grande adoção do formato possibilitou também o surgimento dos agregadores de RSS. Agregadores são aplicações com o objetivo de concentrar as fontes e atualizações enviadas por diversos RSS.

O formato podcast surgiu em 2004, na mesma época que o RSS havia se consolidado. O formato também usou o mesmo padrão RSS para disponibilização de conteúdos. Porém, a ampla adoção do formato por produtores e consumidores de conteúdo é recente. Além disso, a maioria dos agregadores é apenas para conteúdos de artigos e notícias escritos ou de podcast.

1.1 PROBLEMA

A internet é uma fonte inesgotável de informação. Somente no portal colaborativo Wikipédia em 2015 foram publicados 20 mil novos artigos por dia (HETHERINGTON, 2015). Não somente conteúdo em texto, mas também muitos podcasts são produzidos diariamente. Segundo a empresa Apple, de 2014 a março de 2018 o número de episódios publicados em aplicativos da empresa chegou a 50 bilhões (LOCKER, 2018).

Além da grande quantidade de informação produzida, a pluralidade de meios pelos quais temos acesso a esses conteúdos é bastante grande também. No mesmo ano de 2018, segundo a Apple, havia aproximadamente 525 mil canais de podcast ativos por mês. Esses números dão uma ideia da quantidade de conteúdos novos publicados todos os dias (LOCKER, 2018).

Para fazer frente a essas informações, um tipo de ferramenta utilizado é o agregador de conteúdos. Os agregadores são aplicativos para organizar endereços na internet disponibilizados por produtores de conteúdos. Nesses endereços, periodicamente são publicados documentos de textos em formatos específicos, o RSS - Really Simple Syndication. Os RSS são páginas destinadas a serem facilmente interpretadas por aplicativos de terceiros. O intuito disto é facilitar o compartilhamento de informações na internet.

Com a crescente oferta de podcasts nos últimos anos, começaram a surgir também agregadores para canais somente desse tipo de conteúdo. Porém, são poucos os agregadores que organizam conteúdos de formatos diferentes, como áudio, texto e vídeo.

Este trabalho apresenta o desenvolvimento de um aplicativo com a finalidade de servir como agregador para os dois principais tipos de conteúdos em termos de RSS, texto de sites de notícias e blogs, e os conteúdos em áudio dos podcasts. O objetivo é facilitar o acesso e manutenção para essas fontes de diferentes tipos de conteúdos, simplificando a manutenção de canais RSS.

1.2 OBJETIVO GERAL

Criar um aplicativo de gerenciamento de endereços para fontes de conteúdos que são disponibilizados obedecendo padrões já definidos. Os conteúdos sendo dos seguintes formatos: texto ou áudio.

1.3 OBJETIVOS ESPECÍFICOS

Os objetivos específicos do trabalho são:

a) Armazenar endereços de feeds.

b) Consultar por atualizações de conteúdos nos feeds armazenados.

c) Apresentar conteúdos das consultas aos feeds, separados por formatos.
d) Possibilitar armazenamento dos conteúdos no dispositivo, para uso offline.

1.4 JUSTIFICATIVA

A grande quantidade de conteúdos que recebemos atualmente demandam uma forma de gerenciamento. Apesar da estimativa de que 4 bilhões de pessoas utilizem a rede atualmente, muitos ainda estão descobrindo como utilizá-la, experimentando e conhecendo as possibilidades (ROSER; RITCHIE; ORTIZ-OSPINA, 2019).

Centralizar diferentes tipos de conteúdo em um mesmo agregador facilita o gerenciamento e também possibilita uma experiência mais amigável para pessoas que estão buscando se informar através de aplicativos no celular.

Nas próximas páginas será apresentado como os principais padrões utilizados hoje para feeds, o RSS e Atom, surgiram em consonância com a invenção da web. Será abordado em separado cada um dos padrões e contextualizado as primeiras iniciativas neste sentido. Também serão apresentadas outras propostas no mesmo sentido e como elas diferem entre elas e do presente trabalho.

Após esta contextualização, este trabalho mostrará como foi planejado e executado o sistema. Quais foram as técnicas utilizadas para organizar as etapas de desenvolvimento do software. Em seguida, serão elencadas as tecnologias utilizadas e como foi organizada a arquitetura da aplicação. Por fim, serão discutidos os resultados alcançados com o aplicativo.

2 FUNDAMENTAÇÃO TEÓRICA

Neste capítulo será contextualizado o surgimentos dos feeds. Qual a relação que os padrões de compartilhamento de conteúdos, como o RSS e o Atom, foram criados e sua adoção ao longo da história. Para isso, trará uma breve introdução da história da internet, relacionando isto com o desenvolvimento desses padrões.

2.1 A WEB

O embasamento teórico relaciona-se com o início da utilização da internet de forma disseminada. A World Wide Web (WWW) iniciou-se em 1991, quando o primeiro navegador foi disponibilizado na rede mundial de computadores pelo CERN (CASTELLS, 2003). A internet como conhecemos surgiu da conexão, através de padrões comuns, de diversas redes menores, com o intuito de estabelecer uma forma comum de compartilhar informações através de redes de computadores.

O compartilhamento dessas informações precisava ser feito através de padrões que pudessem ser seguidos para facilitar o processamento em computadores. Procurando uma solução para isso que o cientista da computação inglês Tim Berners-Lee propôs, no início da década de 90, enquanto trabalhava no Laboratório Europeu para a Física de Partículas, o CERN (sigla em inglês), uma forma de troca de informações através de redes de computadores (BERNERS-LEE, 1989).

A proposta de Berners-Lee trazia um conjunto de padrões com o intuito de gerenciamento das informações de experimentos realizados no CERN. Esses padrões previam o compartilhamento do conteúdo através de uma rede de computadores que obedeceriam o padrão hoje conhecido como cliente/servidor. Tendo de um lado os computadores servidores, que ficariam disponíveis durante todo o tempo, e do outro as máquinas clientes, com aplicativos capazes de reconhecer informações disponibilizadas pelos servidores e apresentar ao usuário de forma a facilitar seu consumo. (Berners-Lee, 1989)

Assim, a partir de aplicativos “navegadores”, as pessoas poderiam enviar e receber informações sobre estudos acadêmicos, já que a principal utilização da rede de computadores que existia na época era para fins acadêmicos. Para este conjunto

de padrões e a forma de compartilhamento através de computadores servidores e clientes, ele deu o nome de World Wide Web (BERNERS-LEE, 1989).

Em 1990, Berners-Lee compartilhou na rede o primeiro aplicativo “navegador”, também disponibilizando seu código para que outros programadores pudessem fazer melhorias na aplicação (CASTELLS, 2003). Houveram diversas contribuições e outros navegadores foram desenvolvidos. No mesmo ano o CERN publicou o primeiro site na internet, uma página em Hypertext Transfer Protocol (HTTP), um dos padrões que Berners-Lee havia definido.

Nos anos seguintes muitas empresas começaram a se interessar em disponibilizar conteúdos na internet. Já em 1993 os primeiros sites de notícias, como o Bloomberg.com, começaram a ser publicados. Estima-se que ao final de 1993 já haviam 693 sites na web.

O primeiro navegador comercial surgiu em 1995, o Netscape Navigator, desenvolvido pela empresa Netscape Communications. De acordo com Castells (2003, posição 118), este também foi o primeiro ano de uso disseminado da internet, quando existiam aproximadamente 16 milhões de pessoas na rede.

No mesmo ano, percebendo o crescimento acelerado da internet e a importância que a rede estava tomando, a empresa Microsoft lança seu próprio navegador, o Internet Explorer. O navegador foi disponibilizado na versão 95 do sistema operacional Windows, também desenvolvido pela empresa (CASTELLS, 2003).

2.2 WEB SEMÂNTICA

Ainda em meados dos anos 90, concomitante ao lançamento do Netscape Navigator, o crescimento de usuários na rede levou muitas empresas a se inserir neste novo meio. Umas dos primeiros setores empresariais que começaram a utilizar a WWW foram os meios de mídia tradicionais, mais especificamente os jornais impressos.

No ano de 1995 já haviam 150 jornais nos Estados Unidos publicando conteúdos na internet. Em março de 1998 o jornal BBC News teve 8,7 milhões de páginas vistas.

Nesta mesma época estava sendo pensada uma forma de deixar as informações mais organizadas na internet. Em 1997 foi publicado a primeira versão de rascunho de trabalho para o padrão RDF - Resource Description Framework, desenvolvido por Ora Larissa e Ralph Swick. O RDF era uma das bases para uma iniciativa que Berners-Lee chamou de Web Semântica (Powers, 2003). A Web Semântica é uma criação também de Berners-Lee, como a WWW, mas com o intuito de propiciar a troca de informações na internet de maneira organizada. Segundo Berners-Lee (2001):

> A Web Semântica dará estrutura ao conteúdo relevante das páginas web, criando um ambiente onde aplicações de software que percorrem página por página podem prontamente realizar tarefas complexas para os usuários. (Berners-Lee, 2001).

O W3C, World Web Consortium, organização criada por Berners-Lee em 1994 com a finalidade de estabelecer padrões a serem utilizados na Web, definiu o RDF como “ uma linguagem desenhada para dar suporte à Web Semântica, do mesmo modo que o HTML é a linguagem que ajudou a iniciar a Web original." (POWERS, 2003).

2.3 RSS

Em um esforço similar ao de Tim Berners-Lee, mas buscando padronizar um formato para disponibilização de conteúdos atualizados periodicamente na internet, um funcionário da Netscape, Ramanathan V. Guha, criou o padrão RDF Site Sumary (RSS Advisory Board). Esta foi a primeira versão de um RSS, publicada em 1999. Guha também estava envolvido no desenvolvimento do RDF e levou sua experiência para a empresa (GUHA, 2019). A principal motivação da Netscape para o desenvolvimento do RSS era um modelo para disponibilização de notícias através de uma página web, utilizando-se da linguagem de marcação XML – Extensible Markup Language.

Aos longos dos anos o formato RSS foi modificado e atualizado por diferentes autores, visando incorporar melhorias e evoluções que correspondem a forma como os conteúdos são distribuídos na internet. Porém, houve algumas controvérsias na discordância de padrões o que dificultou a definição de um padrão

definitivo. Atualmente, a versão mais recente é a 2.0, cedida por Dave Winer para Berkman Centro para Internet e Sociedade na Escola de Direito de Harvard Harvard (RSS Advisory Board).

2.4 ATOM

O Atom - Atom Syndication Format, é uma opção ao RSS. Diferente deste último, o formato foi desenvolvido como um protocolo web por meio do Internet Engineering Task Force (IETF). A intenção de trazer um novo protocolo seria adotar um padrão definitivo para os feeds e que tivesse melhorias em relação ao RSS, evitando as controvérsias. (HAMMERSLEY, 2005)

A primeira versão do Atom foi publicada em 2005. Dois anos mais tarde, seria também publicado o padrão Atompub, que prescreve um padrão para disponibilização de conteúdos utilizando o formato Atom. Atualmente, esta opção ao RSS é amplamente utilizada. (HAMMERSLEY, 2005)

2.5 FEED

Feed é a forma como são chamados os canais de divulgação que disponibilizam endereços para que outros sistemas possam ter acesso a atualizações periódicas de conteúdos, como postagens, vídeos, podcasts e notícias (WUSTEMAN, 2004). Os dois principais padrões utilizados em feeds são o RSS e Atom.

Os feeds começaram a se popularizar no início dos anos 2000, no momento em que muito do esforço para tornar a web mais organizada era feito. O contexto era pós as muitas incertezas sobre os padrões utilizados nos feeds e um grande crescimento no consumo de notícias online nos Estados Unidos. (BARAK, 2004)

Atualmente, com o grande crescimento no número de podcasts, a utilização de feeds se tornou ainda mais popular e importante. Os conteúdos oferecidos nos diversos canais se diversificaram e atingem milhões de usuários através dos agregadores. (LOCKER, 2018)

2.6 PODCAST

O nome “podcast” foi criado em 2004 pelo jornalista inglês Ben Hammersley. Porém, a prática de distribuição de áudios com pessoas falando sobre determinado assunto foi iniciada em 2000. Uma fabricante de aparelhos para audição de arquivos no formato MP3 (MPEG-1/2 Audio Layer 3) chamada MyAudio2Go criou um serviço que permitia aos donos dos aparelhos baixarem áudios com notícias e outros temas variados. (POLITI, 2019)

Apesar da empresa não ter prosperado, Dave Winer, um dos criadores do RSS, viu que poderia utilizar o padrão desenvolvido por ele para distribuição de notícias e outros conteúdos em áudio (POLITI, 2019). Em 2002, Winer lançou uma versão do padrão RSS que incluiu a possibilidade de endereços de arquivos de áudio para serem baixados. (WINER, 2002)

Foi somente em 2003 que o podcast começou a se popularizar através do padrão RSS, sendo que na época um fator determinante foi o iPod, dispositivo que inspirou a criação do termo podcast. Este aparelho foi um precursor do smartphone inventado pela empresa de tecnologia Apple em 2002 para ouvir arquivos em MP3 (POLITI, 2019). Em 2004, na mesma reportagem que inaugurou o termo “podcast”, Hammersley também falou sobre o incrível crescimento do formato impulsionado pelo iPod, (HAMMERSLEY, 2004)

Atualmente, o podcast é um formato de mídia consolidado e em pleno crescimento. Segundo uma pesquisa do IBOPE (Instituto Brasileiro de Opinião Pública e Estatística), 40% dos internautas brasileiros já escutaram um programa de áudio na internet. Segundo dados da plataforma Spotify, um serviço por assinatura para streaming músicas e podcasts, o Brasil já é o segundo maior mercado no mundo para podcasts, atrá apenas dos Estados Unidos. (Fonseca, 2019)

2.7 AGREGADORES DE FEEDS

O consumo de feeds é feito através de uma aplicação que possibilita salvar endereços de diferentes fontes e apresentar as atualizações com novos conteúdos de maneira organizada (BAJAK, 2004). Os diferentes tipos de agregadores para feeds existem na internet há anos. (CATONE, 2007)

Com o recente crescimento dos podcast surgiram agregadores especializados somente neste tipo de conteúdo. Por outro lado, existem poucos agregadores híbridos, que ofereçam suporte para conteúdos de podcasts e artigos ou notícias. Neste ano a empresa Spotify, responsável pelo aplicativo homônimo, firmou esta tendência ao comprar outras duas empresas norte americanas do ramo de podcasts, a Gimlet e a Anchor. (ALEXANDRO, 2019)

O sistema aqui proposto faz frente a esta falta, propondo uma aplicação que gerencie tanto textos quanto podcasts, o aplicativo Content. Antes de especificar sobre as características da aplicação, serão apresentadas duas outras propostas de agregadores de feeds semelhantes ao sistema Content, os aplicativos Read e RSS Reader.

Foram selecionados os dois pela semelhança ao projeto proposto. Existem inúmeras outras alternativas de aplicativos, porém, que diferem em alguns aspectos relevantes ao sistema Content. As características principal dos dois aplicativos é possibilitarem a inclusão manual de um endereço de feed. Além disso, possibilitam o gerenciamento destes feeds por meio de uma interface simples e sem necessidade de cadastro prévio para ter acesso às funcionalidades do aplicativo.

Os agregadores escolhidos também apresentam algumas funcionalidades que tornam-os opções ainda mais próxima do aplicativos Content, que são: permitir ouvir podcasts, ter filtros e diferentes tipos de ordenação e marcar e desmarcar conteúdos como lidos. Outro dados importante é que os aplicativos foram baixados mais de 10 mil vezes na loja de aplicativos do sistema Android, Play Store.

2.7.1 READ

O aplicativo Read é o primeiro a ser apresentado conforme as características discriminadas ao fim do capítulo anterior. Em outubro de 2019, mês anterior ao da redação deste documento, possuía mais de 100 mil downloads na loja de aplicativos Play Store, disponível nos celulares com sistema operacional Android.

Nele é possível adicionar uma fonte de feed digitando o seu endereço na internet ou fazer uma busca por palavras chaves e adicionar uma das opções retornadas pelo aplicativo. O feeds ficam organizados no menu à esquerda pelo título. Não é possível listar todas as atualizações conforme data, apenas separado por feed.

Um ponto forte do aplicativo é que ele possui um pequeno tutorial demonstrativo de como utilizá-lo. Ele também apresenta a opção de baixar os áudios para podcasts diretamente na memória do celular. Porém, a integração com a usabilidade do aplicativo é parcial, pois os conteúdos baixados ficam disponíveis em um menu à esquerda e os textos que acompanham o conteúdo não são salvos, somente arquivo de áudio.

2.7.2 RSS READER

O RSS Reader é o segundo conforme as características apresentadas. Na mesma data que Reader, possuía também mais de 100 mil downloads na Play Store. A proposta do aplicativo também é bastante parecida com a do Read, apenas diferenciando que possui a opção de listar os conteúdos de todos os feeds na mesma tela.

Um ponto fraco do aplicativo é que, apesar de possibilitar salvar endereços de feeds para podcasts, não possui a funcionalidade de reproduzir ou baixar os áudios. Outro problema do aplicativo é apresentar um botão com a opção de baixar conteúdos ao listar, mas o botão não estar funcionando ou falta o menu onde acessar os conteúdos baixados.

Com a falta de aplicativos agregadores de feeds que gerenciam conteúdos mistos, fica a oportunidade de atingir um público que quer se informar por diferentes fontes, tendo em comum o formato dos feeds.

Neste capítulo foi apresentado o contexto no qual se insere a proposta de software desenvolvida. No próximo capítulo serão descritos os métodos utilizados no desenvolvimento da aplicação e também as técnicas empregadas e materiais de terceiros que serviram de apoio.

3 MÉTODOS E MATERIAIS

Neste capítulo será descrito todo o processo utilizado no desenvolvimento do sistema, assim como técnicas e materiais que auxiliaram. A criação do software foi divididas conforme as principais funcionalidades do software, que são: permitir adicionar um feed informando seu endereço na internet, listar novos conteúdos, visualizar um conteúdo, permitir ouvir podcasts, baixar um conteúdo do feed, ter filtros e diferentes tipos de ordenação e alterar o status de um conteúdo como não lido ou não lido.

3.1 SCRUM

Como metodologia de desenvolvimento foi utilizado o framework Scrum, metodologia ágil utilizada para organizar a criação de sistemas. Foram feitas adaptações necessárias para que fosse aplicado ao desenvolvimento por apenas uma pessoa.

O Scrum foi criado por Jeff Sutherland e Ken Schwaber em meados da década de 90 para auxiliar no desenvolvimento de software. Neste método, o processo de desenvolvimento é dividido em ciclos de períodos regulares chamados sprints. Uma sprint deve ter o período máximo de 1 mês. (PRESSMAN, 2011)

O que será feito em uma sprint é determinado conforme a emergência das tarefas por realizar. Todas as tarefas ainda não listadas em uma sprint aguardam no Backlog. A quantidade de sprints necessárias para concluir um projeto variam conforme o tamanho do projeto a ser desenvolvido. (PRESSMAN, 2011)

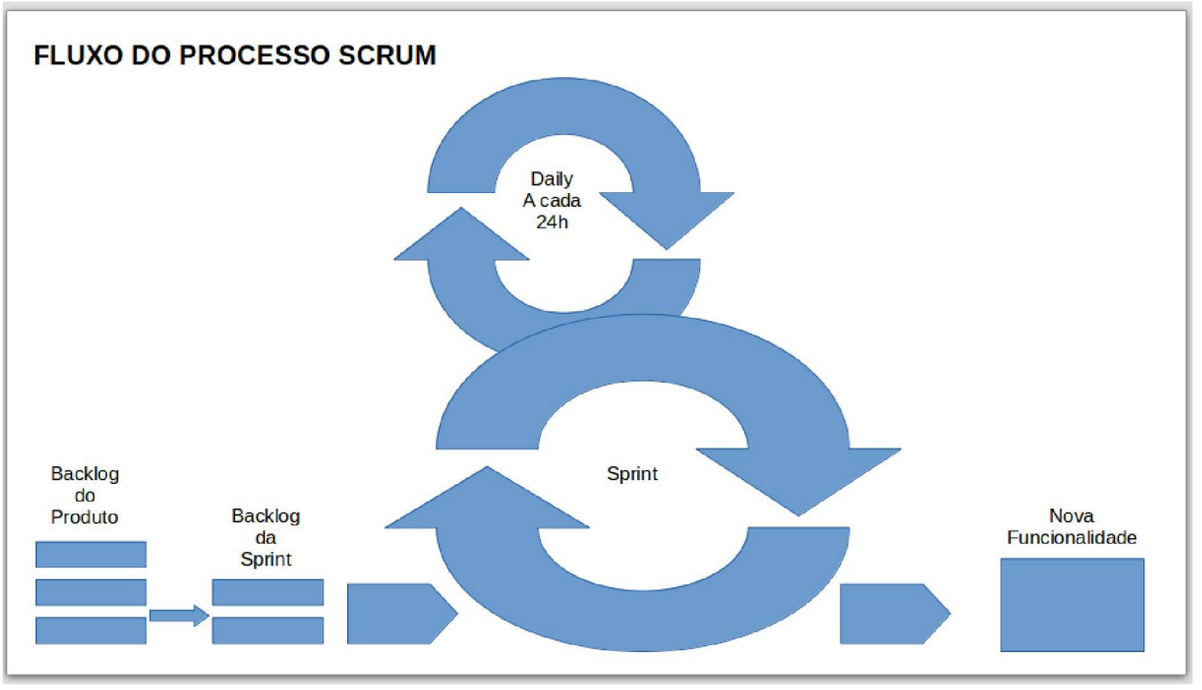
Para cada sprint, o conjunto de tarefas a ser realizado também são definidas como o Backlog. Porém, ao final da sprint tudo que foi escrito de código passa a fazer parte de um resultado parcial do produto, desde que esteja funcionando. Estas finalizações de sprint ao longo do desenvolvimento devem representar acréscimos de funcionalidades acabadas ao software, possibilitando seu uso desde a entrega da primeira sprint. No Scrum, cada entrega feita representa um Demo do sistema. (PRESSMAN, 2011)

No gerenciamento das tarefas e sprints são separados os papéis do Scrum Master, que seria o especialista da equipe e um guia para a metodologia, do Product Owner, que fica responsável por realmente gerenciar a aplicação do método, e da

equipe de desenvolvimento. Como a aplicação foi desenvolvida apenas por uma pessoa, os papéis não existiram de forma distinta no desenvolvimento da aplicação.

Para o desenvolvimento deste projeto, as sprints foram organizadas em ciclos de 4 semanas com 16 horas de duração cada. A fim de definir uma pontuação relacionada ao tempo de trabalho, foi contabilizado 1 ponto para cada hora de desenvolvimento.

FIGURA 1 – CICLO DE SPRINTS NO SCRUM



Fonte da imagem: O autor

As funcionalidades foram definidas conforme histórias de usuários. No ciclo de uma sprint foram desenvolvidas uma ou mais histórias de usuário, dependendo do tempo planejado para desenvolvimento da funcionalidade representada pela história.

As seguintes histórias foram definidas:

a) eu como usuário quero salvar o link de um Feed;

b) eu como usuário quero listar os feeds salvos;

c) eu como usuário quero visualizar informações de um feed;

d) eu como usuário quero excluir ou inativar um Feed;

e) eu como usuário quero visualizar uma lista dos conteúdos de um Feed;

f) eu como usuário quero visualizar um conteúdo;

g) eu como usuário quero ouvir um podcast;

h) eu como usuário quero salvar o conteúdo de um feed no celular para acesso offline;

i) eu como usuário quero ouvir o conteúdo de texto de um feed;

No QUADRO 1 é possível ver como ficaram as divisões da histórias para cada sprint:

QUADRO 1 – SEPARAÇÃO DAS HISTÓRIAS EM SPRINTS

sprint 1	sprint 2	sprint 3	sprint 4	sprint 5
eu como usuário quero salvar o link de um Feed;	eu como usuário quero listar os feeds salvos;	eu como usuário quero visualizar informações de um feed; eu como usuário quero excluir ou inativar um Feed; eu como usuário quero visualizar uma lista dos conteúdos de um Feed;	eu como usuário quero visualizar um conteúdo; eu como usuário quero ouvir um podcast;	eu como usuário quero salvar o conteúdo de um feed no celular para acesso offline; eu como usuário quero ouvir o conteúdo de texto de um feed;

FONTE: Marcelo Marzola Leite (2019).

Ao todo, o desenvolvimento do software foi dividido em cinco sprints, com 16 pontos cada uma delas. Nas sprints foram alocadas as histórias de usuário divididas conforme a duração e também a relação entre as funcionalidades, para evitar conflitos entre partes dependentes que ainda não estivessem implementadas. Abaixo, a TABELA 1 mostra uma visão geral de como foi definida a pontuação de cada história e as datas previstas para iniciarem e terminarem:

TABELA 1 – DIVISÃO DE ESFORÇO PARA AS HISTÓRIAS

Histórias	Ponto	sprint
eu como usuário quero salvar o link de um feed	16	1
eu como usuário quero listar os feeds salvos	16	2
eu como usuário quero visualizar informações de um feed	4	3
eu como usuário quero excluir ou inativar um feed	4	3
eu como usuário quero visualizar uma lista dos conteúdos de um feed	8	3
eu como usuário quero visualizar um conteúdo	8	4
eu como usuário quero ouvir um podcast	8	4
eu como usuário quero salvar o conteúdo de um feed no celular para acesso offline	8	5
eu como usuário quero ouvir o conteúdo de texto de um feed	8	5

FONTE: Marcelo Marzola Leite (2019).

O desenvolvimento do sistema foi iniciado no mês de junho e finalizado em outubro. Abaixo, as datas que fecharam a iniciaram os ciclos de quatro semanas de cada sprint foi definido conforme a TABELA 2

TABELA 2 – PLANEJAMENTO DAS SPRINTS

sprints	Início	Término
sprint 1	03/06	30/06
sprint 2	01/07	28/07
sprint 3	29/07	25/08
sprint 4	26/08	22/09
sprint 5	23/09	20/11

FONTE: Marcelo Marzola Leite (2019).

Antes de iniciar o desenvolvimento do sistema foram feitos protótipos para as telas, que serviram de guia. As funcionalidades foram implementadas tomando por base os protótipos de telas e as Especificações de Casos de Uso, presentes respectivamente no APÊNDICE A e APÊNDICE B. Quanto a interação do usuário com a interface do sistema, foi levado em consideração as possibilidades apresentadas nos Diagramas de Caso de Uso, conforme o APÊNDICE E.

3.2 SPRINTS

Seguindo o método Scrum, todas as sprints foram finalizadas entregando uma funcionalidade pronta. Para isso, a cada sprint foi necessário implementar as mesmas interfaces que fazem as ligações entre os módulos que compõem a arquitetura do software. Isto implica que para todas as sprints praticamente as mesmas interfaces foram implementadas, com poucas diferenças.

3.2.1 SPRINT 1

A primeira história durou todo o ciclo da sprint, pois foi necessária a implementação de todos as classes auxiliares, responsáveis por lidar com assincronicidade e gerenciamento de entidades. Porém, das classes que representam as estruturas de dados do Feed e Conteúdos, conforme mostrado no APÊNDICE C, nesta primeira etapa foi desenvolvida apenas a do feed.

Foi realizada a história “eu como usuário quero salvar o link de um Feed”, que apresenta uma tela onde o usuário pode informar o endereço do Feed desejado (APÊNDICE A). A camada de comunicação com um banco de dados interno ao dispositivo celular do usuário também foi desenvolvida nesta fase do projeto, bem como a parte de comunicação com os feeds.

Portanto, todo o gerenciamento de dados do sistema foi criado, possibilitando já na primeira entrega do projeto tanto salvar os endereços e informações do Feed quanto consultar por atualizações. Para implementar o fluxo de dados entre usuário do aplicativo e armazenamento interno, foi levado em conta os diagramas de sequência, conforme indica o APÊNDICE D.

3.2.2 SPRINT 2

A segunda sprint também durou um ciclo inteiro. Assim como a primeira sprint, está também centrou-se apenas em implementar uma funcionalidade para o feed. A história aplicada foi “eu como usuário quero listar os feeds salvos”, que para funcionar necessita que sejam implementadas classes específicas que apresentem o conteúdo em forma de lista no dispositivo do usuário. Os itens da lista também podem receber métodos que respondem a interação do usuário, fazendo necessário aplicar essas lógicas.

Além disso, a lista de feeds possui filtros conforme o estado dos feeds, que podem estar “ativos” ou “inativos”. Essas diferenças de estados e ações também tiveram como base os diagramas de sequência apresentados no APÊNDICE D.

3.2.3 SPRINT 3

A terceira foi a maior em termos de histórias implementadas. Esta foi a primeira sprint a abordar alguma história relacionada aos conteúdos de um feed. Foram adicionadas as seguintes histórias no backlog da sprint: "eu como usuário quero visualizar informações de um feed", "eu como usuário quero excluir ou inativar um feed" e "eu como usuário quero visualizar uma lista dos conteúdos de um feed".

No momento de desenvolvimento desta sprint, como a maioria das interfaces já estavam construídas e com exemplos em funcionamento, bastou replicar muitas

das estruturas. Foi uma sprint com poucos artefatos novos no conjunto da arquitetura, mas dispendiosa nas horas.

3.2.4 SPRINT 4

A quarta sprint foi inteiramente focada na visualização dos conteúdos, tanto no formato texto quanto para podcast. As seguintes histórias foram inseridas no backlog da sprint: “eu como usuário quero visualizar um conteúdo” e “eu como usuário quero ouvir um podcast”. As duas histórias apresentam a mesma tela para o usuário, sendo que o sistema faz a diferenciação pelos dois tipos de conteúdo e carrega um controle de reprodução caso o tipo do conteúdo seja podcast.

3.2.5 SPRINT 5

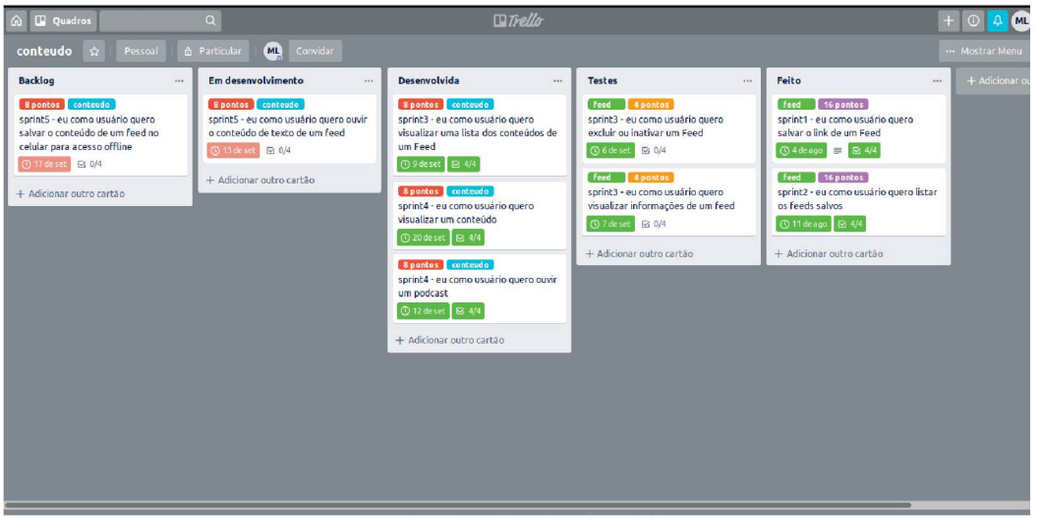
Esta foi a última sprint, porém, acrescentou funcionalidade que permitem a comodidade de baixar conteúdos para leitura e audição posterior, bem como de ouvir o texto de um conteúdo. A histórias realizadas nesta sprint foram duas: “eu como usuário quero salvar o conteúdo de um feed no celular para acesso offline” e “eu como usuário quero ouvir o conteúdo de texto de um feed”.

Quando foram implementadas, a base para o sistema já estava toda pronta, bem como o fluxo de dados e a maior parte das possibilidades de interações com o usuário. Porém, exigiu o tempo de uma sprint inteira por serem duas funcionalidades mais complexas que as implementadas até o momento e que não seria possíveis enquanto as outras partes não estivessem prontas.

3.2 ORGANIZAÇÃO

Para organizar as sprint e as histórias, bem como cada uma das implementações necessárias em cada etapa do desenvolvimento, foi utilizado uma versão online do quadro kanban, disponibilizado pelo site Trello. Além disso, como a plataforma permite adicionar marcações coloridas no cards do kanban, foram utilizadas cores para separar as histórias por número de pontos e entidades que estariam relacionadas no sistema.

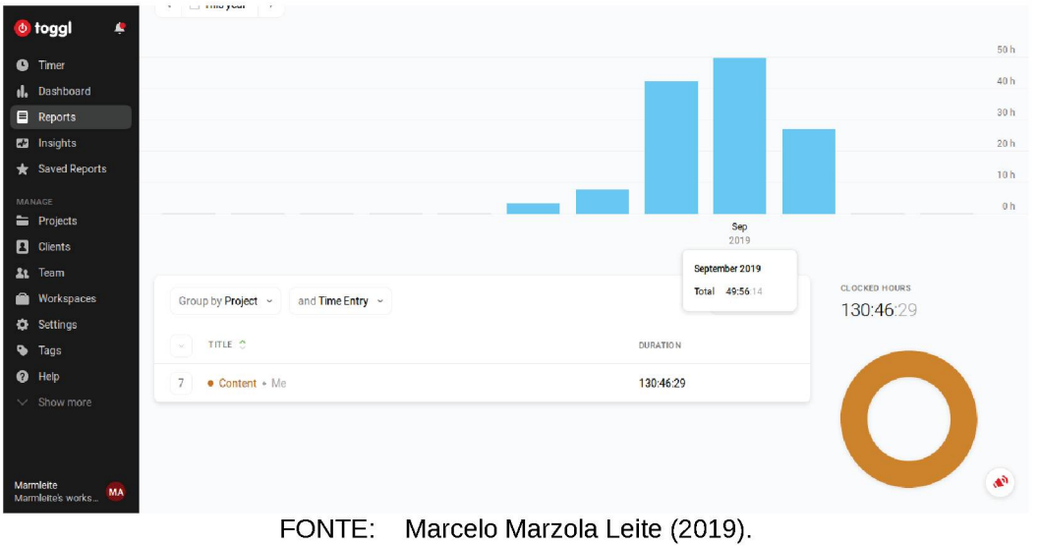
FIGURA 2 – QUADRO KANBAN TRELLO



FONTE: O autor (2019).

Para contar o tempo gasto em cada uma das sprints, foi utilizada a ferramenta de medição Toggl . Assim, cada uma das etapas teve o tempo definido para que houvesse adequação conforme o projeto fosse desenvolvido.

FIGURA 3 – MARCAÇÃO DE TEMPO TOGGL



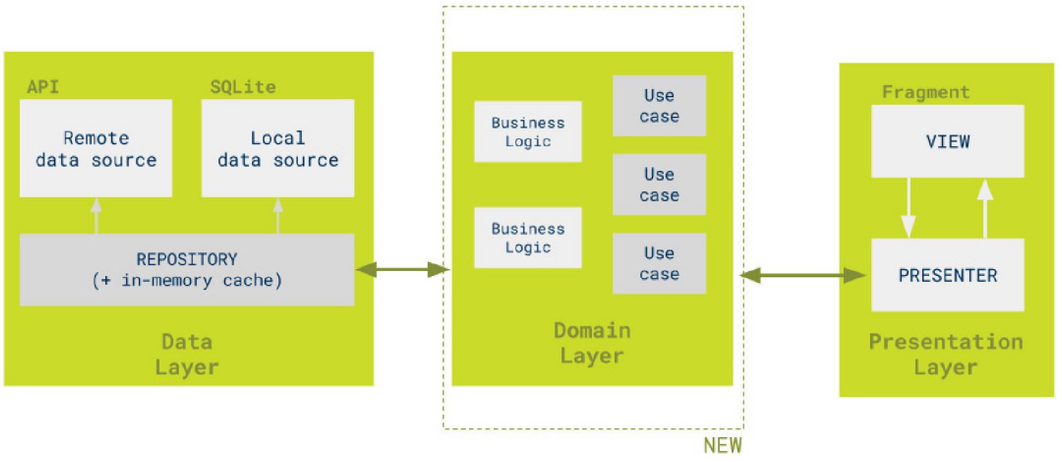
FONTE: Marcelo Marzola Leite (2019).

3.3 TECNOLOGIAS

A aplicação foi desenvolvida especificamente para a plataforma Android utilizando a linguagem Java, através da aplicação AndroidStudio. Foi escolhida esta plataforma porque o aplicativo foi desenvolvido para ser utilizado em dispositivos celulares ou tablets.

No desenho do sistema optou-se por seguir uma das arquiteturas de projetos recomendadas pelo Google no Github. A empresa disponibiliza exemplos de arquiteturas com código livre para modificação e reprodução para servir como base em aplicações desenvolvidas para o sistema operacional Android. Das opções disponibilizadas, a selecionada foi a MVP+Clean Architecture - um padrão de projeto denominado Model View Presenter, que segue também as diretrizes do padrão Clean Architecture (MARTIN, 2017).

FIGURA 4 – ANDROID BLUEPRINTS MVP+CLEAN ARCHITECTURE



FONTE: gitbhub.com/android/architecture-samples/tree/todo-mvp-clean (2019).

O padrão MVP é uma derivação do mais conhecido Model View Controller (MVC), onde a camada Presenter faz o papel de intermediação da camada Controller do MVP. A camada Model ficaria responsável pela lógica dos dados, acesso e estrutura dos dados. Assim, a camada Presenter seria apenas para definir quais são as chamadas necessárias aos dados do model e validações de condições para isso.

No caso do modelo escolhido, definido por Robert C. Martin (2017), acrescenta-se uma camada intermediária para isolar a lógica de negócio do acesso

e apresentação dos dados. Este isolamento determinar quais ações possíveis na Presenter através da implementação de uma interface definida como UseCase.

Esta diferenciação do padrão MVP comum, em que a camada de lógica de negócio fica junto ao model, permite que a aplicação da lógica de negócio não esteja condicionada ao acesso dos dados ou apresentação destes. Apenas faz necessário que faça uma passagem de mensagens entre uma e outra através da implementação de suas interfaces. Assim, caso seja preciso fazer alguma mudança no modo como os são acessados ou apresentados, não terá efeito na camada de domain.

Além disso, para acesso aos dados é utilizado o padrão Repository. O padrão Repository é implementado como uma camada entre o Domain e a camada que faz o mapeamento dos dados, gerenciando um cache e a atualização dos objetos pertencentes aos modelos de negócio criados em tempo de execução.

Para a parte de acesso aos feeds e seus conteúdos, foi utilizada o framework de código aberto Rome, que faz a leitura dos padrões RSS e Atom. Assim, por ser uma ferramenta de código aberto estável, foi dispensada a necessidade de escrever decodificadores de feeds para cada formato apresentado.

Ainda, o Rome também tem código livre para modificação e reprodução, o que permite adicionar novas funcionalidades. Outra utilidade seria acrescentar compatibilidade com outros formatos de feed, caso necessário.

Porém, o framework tem suas próprias classes de negócio para representar um feed e o conteúdo do feed, respectivamente SyndFeed e Entry. Os modelos são baseados nas marcações dos padrões de XML utilizados na implementação de feeds.

Optou-se por fazer o mapeamento desses modelos para os que representam as estruturas de dados definidas no Diagrama de Classes (APÊNDICE C), Feed e Conteúdo. Desta forma, manteve-se o isolamento da aplicação com relação a dependência da biblioteca Rome. O único contato com o framework ficou a cargo das classes que representam as fontes de dados remotas, os feeds.

O acompanhamento das modificações feitas nos códigos, mantendo uma linha histórica do desenvolvimento, foi realizado através do versionamento de código. Foi utilizado a aplicação Git, atualmente a principal ferramenta para este tipo de tarefa, em conjunto com a plataforma Gitlab, também amplamente conhecida. Como o Gitlab também possui uma ferramenta de definir tarefas em um quadro

kanban, foi utilizado isto para replicar os cards do Trello, mas voltada ao desenvolvimento.

Por fim, todas estes materiais e métodos foram aplicados exaustivamente e continuamente durante o período de desenvolvimento do software. Este capítulo apresentou por quais meios foi implementadas essas ferramentas e diretrizes na criação da aplicação. No próximo capítulo será apresentado o resultado deste processo, a aplicação e percepções acerca de sua utilização.

4 APRESENTAÇÃO DOS RESULTADOS

A implementação de todo o trabalho aqui descrito foi feita conforme o planejamento, apenas adequando alguma classes que precisavam ser desenvolvidas concomitantemente com suas dependências. A aplicação funciona conforme o definido, com uma interface simples e direta, apresentando somente as funcionalidades que foram consideradas essenciais.

A FIGURA 5 apresenta todos os feeds já cadastrados pelo usuário. Caso não haja nenhum feed, a tela aparece fazia. No canto inferior direito está o botão que levará à página para adicionar um novo feed.

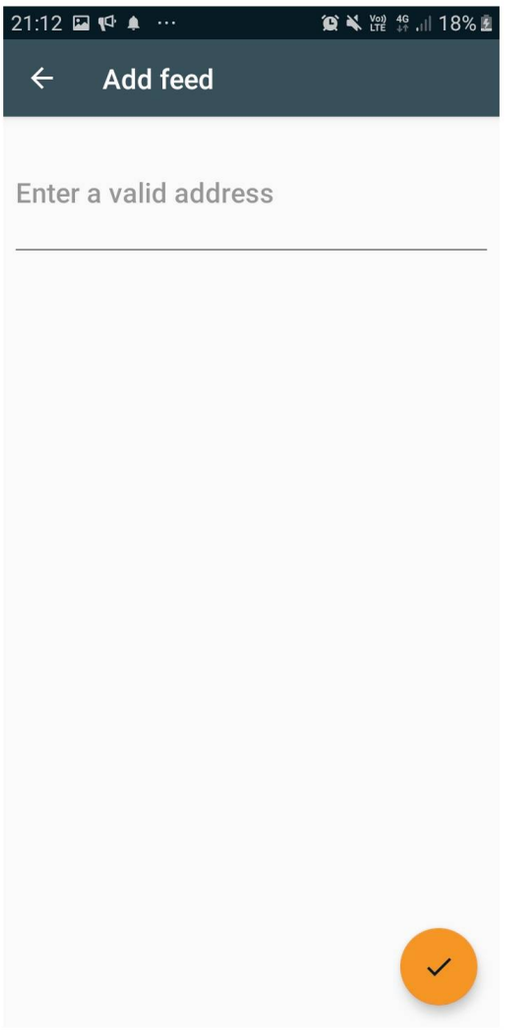
FIGURA 5 – TELA PARA LISTAR FEEDS

FONTE: O autor.

Na FIGURA 6 está a tela do sistema que serve para adicionar um novo feed. Esta figura apresenta o campo para preenchimento com o endereço de um feed. O sistema fará a validação se trata-se de um endereço válido e em funcionamento. No

canto inferior direito fica o botão que sinaliza ao sistema para fazer a verificação e salvar o feed.

FIGURA 6 – TELA PARA ADICIONAR FEED

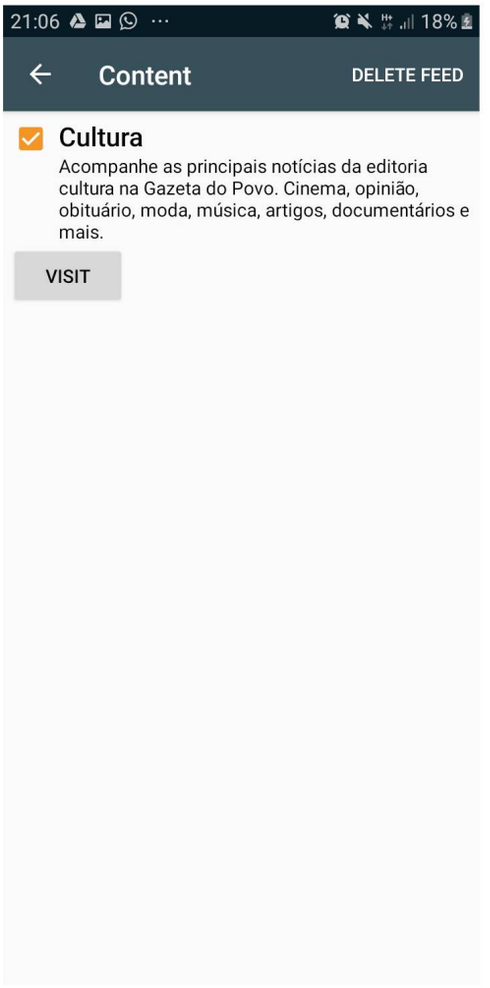


FONTE: O autor.

A FIGURA 7 é responsável por apresentar maiores detalhes do feed. Nela é possível ver sua descrição e também apresenta um link para o endereço onde são apresentados os conteúdos do feed.

Na parte superior, ao lado esquerdo é possível adicionar e remover a marcação da caixa que indica se o feed está ativo ou inativo. Ou seja, se o aplicativo de ou não consultar por atualizações do feed. No lado direito existe a opção de remover o feed da base de dados do aplicativo.

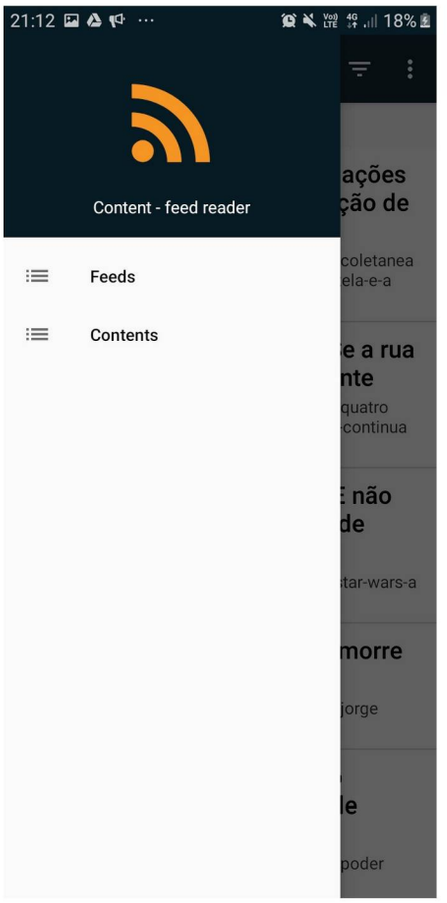
FIGURA 7 – TELA PARA MOSTRAR FEED



FONTE: O autor.

A FIGURA 8 apresenta o menu lateral esquerdo, com a opção de selecionar entra a tela que lista os feeds e a tela que lista os conteúdos.

FIGURA 8 – TELA DE MENU PRINCIPAL



FONTE: O autor.

Continuando a navegação, o usuário que selecionar a opção “Contents”, do menu da FIGURA 8, para exibir os conteúdos, será redirecionado para a tela apresentada na FIGURA 9. Nela é possível visualizar a lista de conteúdos com seus títulos.

No canto superior direito da FIGURA 9, que lista os conteúdos, é possível ver duas opções de menu. A primeira delas está visível na FIGURA 10, onde o usuário pode remover todos os conteúdos ao mesmo tempo marcando-os como lidos, enquanto a segunda opção serve para que o aplicativo faça nova busca por atualizações no endereços de feeds.

FIGURA 9 – TELA PARA LISTAR CONTEÚDOS

FONTE: O autor.

FIGURA 10 – MENU PARA CONTEÚDOS

FONTE: O autor.

A segunda opção de menu que o usuário possui, conforme a FIGURA 11, apresenta opções de filtros. O usuário pode escolher entre ver somente os conteúdos já marcados como lidos, os que foram salvos, ou a opção padrão, os que ainda não foram lidos.

FIGURA 11 – MENU PARA FILTRAR CONTEÚDOS

FONTE: O autor.

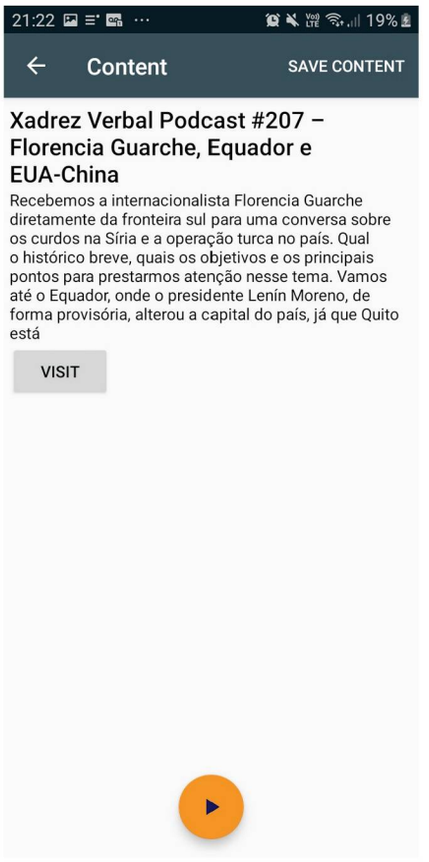
Caso o usuário selecione um dos conteúdos, a FIGURA 12 será apresentada, contendo um texto e a imagem, assim como um link para a página do conteúdo. O tamanho do texto depende de que está disponibilizando o conteúdo. Alguns feeds disponibilizam todo o conteúdo e outros apenas uma parte.

FIGURA 12 – TELA PARA MOSTRAR CONTEÚDO

FONTE: O autor.

Caso o usuário prefira, também poderá salvar o conteúdo offline, para leitura posterior. A mesma tela apresentada na FIGURA 12 também é utilizada para conteúdos no formato podcast. Um player é apresentado ao usuário, como na FIGURA 13, caso o sistema identifique que trata-se deste tipo de conteúdo. Caso seja salvo o podcast, ficará armazenado no dispositivo o texto e o áudio do conteúdo.

FIGURA 13 – TELA PARA MOSTRAR CONTEÚDO - PODCAST



FONTE: O autor.

Com relação ao resultado obtido com o uso das bibliotecas, devido ao tempo, não foi possível testar extensamente os padrões de feed disponíveis. Assim, não há como atestar se o framework Rome, utilizado para fazer a leitura dos XMLs do feeds, possui ampla compatibilidade com todos os padrões.

Ouvir os áudios dos textos de artigos e notícias é uma funcionalidade bastante útil, porém, a audição é dificultosa por ser uma biblioteca nativa do Java. A utilização desta biblioteca deve-se ao fato de ser uma implementação simples, mas possui a desvantagem de não ter o aperfeiçoamento necessário para soar como uma leitura natural.

Optar por mostrar todos os conteúdos organizados por data, independente do feed que pertencem, foi uma decisão acertada. Fica mais natural a leitura em ordem cronológica ao invés de separada por feeds. A experiência de poder gerenciar dois tipos de conteúdos diferentes na mesma aplicação também é bastante prática.

A utilização de propostas já implementadas de alguns dos problemas encontrados, como a divisão entre os componentes da arquitetura do software e o framework Rome, também mostrou-se bastante compensatória.

A arquitetura já definida acrescentou um tempo de aprendizagem e testes para saber como os componentes estavam organizados e se comunicava. Por outro lado, passada essa primeira fase, a implementação de novas funcionalidades, acréscimos e modificações aos diferentes componentes, ficaram mais rápidas e simples.

A biblioteca Roma é bastante completa e não apresentou problemas nos testes realizados até o momento de finalização deste trabalho. Mostrou-se bastante intuitiva no uso, bastando saber quais as especificações dos diferentes padrões de feeds para ter uma base firme para entender a relação entre os diferentes componentes da biblioteca.

Portanto, observou-se que é relevante ter uma aplicativo como opção para gerenciamento de diferentes tipos de conteúdos

5 CONSIDERAÇÕES FINAIS

Apesar de agregadores de feeds não serem algo novo, ainda são poucas as ferramentas que se diferenciam. O crescimento recente dos podcasts trouxe novo impulso à área, que possivelmente passará por transformações e novas experimentações.

Essa tendência de crescimento acompanha o crescimento na quantidade de informações que somos expostos diariamente. Neste sentido, o aplicativo proposto neste trabalho proporciona uma forma simples e centralizada de receber os conteúdos de feeds escolhidos pelo usuário.

Todos os objetivos que foram propostos para a aplicação são cumpridos através das funcionalidades do sistema. Com relação aos filtros e menus, embora o usuário esteja limitado a algumas poucas opções, não deixa de contemplar pontos essenciais.

No contraponto da solução proposta aqui para o problema da crescente quantidade de informações, no decorrer da pesquisa sobre os concorrentes, ficou aparente que os aplicativos mais populares são os que fazem a recomendação de conteúdos. Muitos deles fazem isto sem que seja necessário ao usuário adicionar nenhuma fonte.

Esses aplicativos administram os feeds e as atualizações de conteúdos, filtrando e organizando-os conforme critérios próprios. Apesar disto, o Content supre uma outra parcela de usuários que ainda preferem a possibilidade de adicionar fontes de feed manualmente. É possível confirmar esta preferência através do número apresentados de downloads dos aplicativos concorrentes.

Há ainda opções híbridas, como o aplicativo Feedly, que possui a opção de adicionar feeds informando o endereço na internet, procurar através do nome e sugere outro feeds e conteúdos baseado no comportamento do usuário. Assim, arriscar através de uma aplicação simples apresentar possibilidades de agregar diferentes formatos de rss e mídia é ainda um campo que tem muito a ser experimentado.

5.1 RECOMENDAÇÕES PARA TRABALHOS FUTUROS

Àqueles que forem se aventurar em desenvolver um agregador ou continuar este trabalho, recomenda-se desenvolvimento de sugestões de conteúdos. Para isso, ter uma fonte de dados remota, que guarde somente o que fosse considerado pertinente ou apontado pelo usuário. Com a crescente quantidade de conteúdos, filtrá-los para apresentar ao usuário somente algo que otimize seu dia será uma tarefa bastante importante.

O desafio de uma tarefa como esta seria manter a aplicação do usuário leve e trabalhar com as atualizações em uma plataforma integrada por meio de uma API. Ao que parece, muitas dos agregadores mais populares utilizam de aplicações similares a esta recomendação.

REFERÊNCIAS

ANDROID Architecture Blueprints [beta] - MVP + Clean Architecture. [*S. l.*], 8 nov. 2017. Disponível em: https://github.com/android/architecture-samples/tree/todo-mvp-clean. Acesso em: 1 nov. 2019.

BERNERS-LEE, Tim. Information Management: A Proposal. *In*: **Information Management**: A Proposal. 1. [*S. l.*], 1990. Disponível em: https://www.w3.org/History/1989/proposal.html. Acesso em: 1 out. 2019.

BAJAK, Frank. Enthusiasts Call Web Feed Next Big Thing. **The Associated Press**, [*S. l.*], p. 1-4, 28 fev. 2004. Disponível em: http://www.washingtonpost.com/wp-dyn/articles/A15175-2004Feb28.html. Acesso em: 27 set. 2019.

CATONE, Josh. 5 News Aggregation Methods Compared. **Read/Write Web**, [*S. l.*], p. 1-1, 10 jul. 2007. Disponível em: https://web.archive.org/web/20070828014227/http://www.readwriteweb.com/archives/news_aggregation_methods.php. Acesso em: 27 set. 2019.

FACTS About W3C. [*S. l.*], 2019. Disponível em: https://www.w3.org/Consortium/facts.html. Acesso em: 4 set. 2019.

FONSECA, Ana Carolina. Brasil se consolida como o segundo maior mercado de podcasts do mundo: Evento Spotify for Podcasters Summit reúne criadores de conteúdo e apaixonados por podcast no Brasil; país já é o segundo maior mercado do mundo. **Correio Brasiliense**, [*S. l.*], 2 nov. 2019. Tecnologia, p. 1-1. Disponível em: https://www.correiobraziliense.com.br/app/noticia/tecnologia/2019/11/02/interna_tecnologia,803272/brasil-se-consolida-como-o-segundo-maior-mercado-de-podcasts-do-mundo.shtml. Acesso em: 3 nov. 2019.

GILL, Kathy E. **Blogging, RSS and the Information Landscape: A Look At Online News**. [*S. l.*], 2005. Disponível em: http://www.ramb.ethz.ch/CDstore/www2005-ws/workshop/wf10/gill.pdf. Acesso em: 30 ago. 2019.

GUHA, Ramanathan V. Ramanathan V. Guha. *In*: **Ramanathan V. Guha**. [*S. l.*], 2016. Disponível em: http://www.guha.com/cv.html. Acesso em: 3 set. 2019.

HAMMERSLEY, Ben. Audible revolution. **The Guardian**, [*S. l.*], p. 1-1, 12 fev. 2004. Disponível em: https://www.theguardian.com/media/2004/feb/12/broadcasting.digitalmedia. Acesso em: 27 set. 2019.

HETHERINGTON, Mark. On average how many articles get added to Wikipedia on a daily basis?. *In*: **On average how many articles get added to Wikipedia on a daily basis?**. [*S. l.*], 2015. Disponível em: https://www.quora.com/On-average-how-many-articles-get-added-to-Wikipedia-on-a-daily-basis. Acesso em: 19 set. 2019.

HISTORY of RSS. [*S. l.*], 2004. Disponível em: http://www.rss-specifications.com/history-rss.htm. Acesso em: 5 set. 2019.

RSS History. [*S. l.*], 2009. Disponível em: http://www.rssboard.org/rss-history. Acesso em: 4 set. 2019.

LIÇÕES da História da Internet. *In*: CASTELLS, Manuel. **A Galáxia da Internet**: Reflexões sobre a internet, os negócios e a sociedade. 1. ed. [*S. l.*]: Zahar, 2003. v. 1, cap. 1, ISBN 978-8571107403.

LOCKER, Melissa. Apple?s podcasts just topped 50 billion all-time downloads and streams: platform wars. **Fast Company**, [*s. l.*], 25 abr. 2018. Disponível em: https://www.fastcompany.com/40563318/apples-podcasts-just-topped-50-billion-all-time-downloads-and-streams. Acesso em: 4 set. 2019.

MARTIN, Robert C. **Clean Architecture**: A Craftman's Guide to Software Structure and Design. 1. ed. [*S. l.*]: Prentice Hall, 2017. 432 p. v. 1. ISBN 0134494164. *E-book*.

SHEA, Dave. **What is RSS/XML/Atom/Syndication?**. [*S. l.*], 19 maio 2004. Disponível em: http://www.mezzoblue.com/archives/2004/05/19/what_is_rssx/. Acesso em: 5 set. 2019.

PACETE, Luiz Gustavo. No Brasil, 40% dos internautas já ouviram podcast: Pesquisa do Ibope aponta que 19% dos usuários de internet brasileiros são ouvintes regulares e 32% não conhecem essa mídia. **Meio & Mensagem1**, [*S. l.*], p. 1-1, 13 maio 2019. Disponível em: https://www.meioemensagem.com.br/home/midia/2019/05/13/no-brasil-40-dos-internautas-ja-ouviram-podcast.html. Acesso em: 27 set. 2019.

PRESSMAN, Roger S. **Engenharia de Software: Uma abordagem profissional.** Tradução de Ariovaldo Griesi, Mario Moro Fecchio. 7. ed. Porto Alegre: AMGH, 2011. 780 p. ISBN 978-85-63308-33-7.

PODCASTING today. *In*: MILLER, Michael. **The Ultimate Web Marketing Guide**. Estados Unidos: Pearson Education, 2011. ISBN 978-0-7897-4100-4. PDF.

POLITI, Cassio; ROSA, André. CONHEÇA A HISTÓRIA DO PODCAST NO MUNDO. **Comunique-se**, [*S. l.*], p. 1-1, 27 mar. 2019. Disponível em: https://www.comunique-se.com.br/blog/conheca-a-historia-do-podcast-no-mundo/. Acesso em: 27 set. 2019.

POTTS, Anne; SNOW, Eric; MITCHELL, Ned; LLOYD, Andrew. **World Wide Web Consortium Publishes Public Draft of Resource Description Framework (RDF)**: Key Industry Players Collaborate to Develop Interoperable Metadata for the

Web. [*S. l.*], 6 out. 2010. Disponível em: https://www.w3.org/Press/RDF. Acesso em: 4 set. 2019.

POWERS, Shelley. **Practical RDF**. 1. ed. Estados Unidos: O?Reilly, 2003. 350 p. v. 1. ISBN 9780596002633. MOBI.

WINER, Dave. **RSS 0.92**. [*S. l.*], 2002. Disponível em: http://backend.userland.com/rss092. Acesso em: 27 set. 2019.

THE ATOM Syndication Format. *In*: HAMMERSLEY, Ben. **Developing Feeds with RSS and Atom**: Developers Guide to Syndicating News & Blogs. 1. ed. [*S. l.*]: O 'Reilly Media, 2005. v. 1, cap. 7, p. 115-129. ISBN B00BCQ5Q5K. Disponível em: https://books.google.com.br/books?id=xuPaOWcryZAC&pg=PT129&lpg=PT129&dq=atom+syndication+format&source=bl&ots=rKRLC2-lSS&sig=ACfU3U2tqL1G74zzCXhAonKE9CbOlxLcVg&hl=pt-BR&sa=X&ved=2ahUKEwj6rerFo-PlAhVYHLkGHbxmB2oQ6AEwJ3oECCsQAQ#v=onepage&q=atom%20syndication%20format&f=false. Acesso em: 1 nov. 2019.

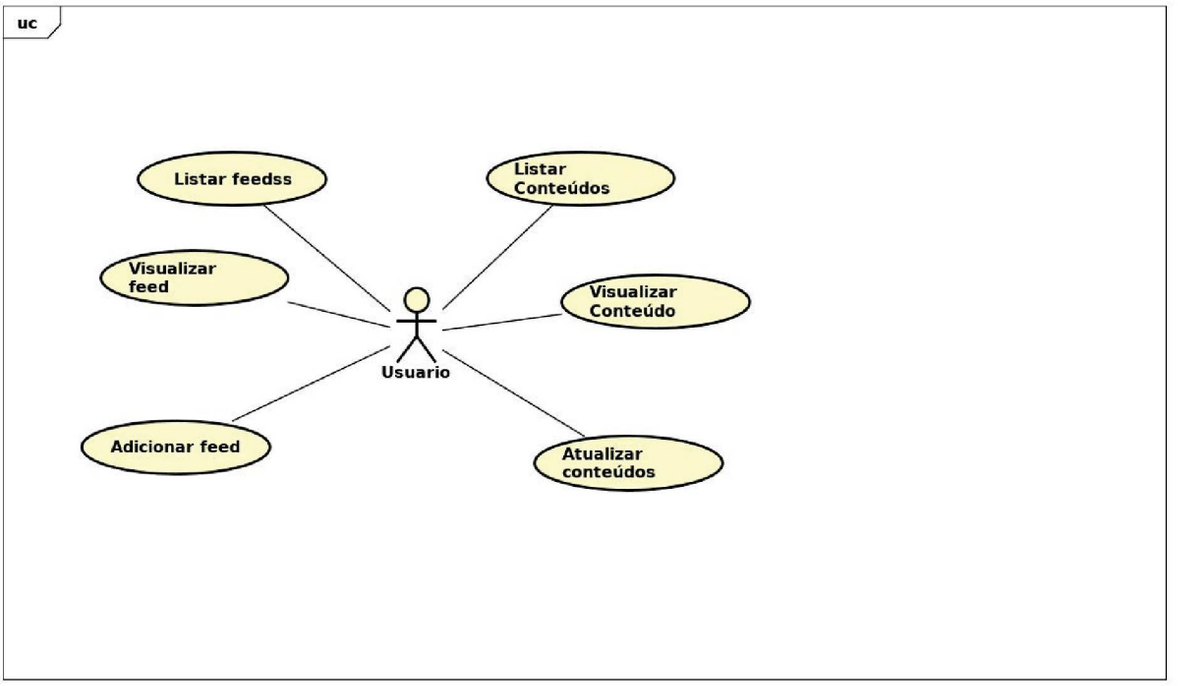
APÊNDICE A – VISÃO

A internet é rica em diversos conteúdos para públicos com interesses heterogêneos. Ter acesso a conteúdos diferentes, otimizados para separar entre gostos individuais e com a opção de diversificar as fontes de conteúdo, é uma tarefa difícil.

Também, muitas vezes lembrar do que foi visto e resgatar conteúdos importantes está além das possibilidades que o tempo diário dispõe. Fazer conteúdos transitarem em grupos diferentes, principalmente para sites pequenos e blogs desconhecidos, parece estar mais custoso com a centralização das redes sociais.

Para ajudar no gerenciamento desses conteúdos, será desenvolvido um aplicativo gerenciador de conteúdos. Será possível salvar links compartilhados de outros aplicativos ou copiados, e assinar fontes de conteúdos em RSS ou Atom. O usuário terá a opção de salvar os textos dos links para leitura posterior e salvar os conteúdos para estar disponível offline.

APÊNDICE B - CASOS DE USO NEGOCIAIS



APÊNDICE C - GLOSSÁRIO

Framework: Abstração que une códigos entre projetos de software para promover uma funcionalidade específica.

Web: Rede mundial de computadores.

Usuário: Dono do celular instalado o aplicativo. O app não necessita cadastro do usuário com login e senha, pois todas as informações pessoas ficam armazenadas no próprio aparelho, apenas compartilhando dados de uso, que serão utilizados para sugestões.

RSS (Really Simple Syndication): padrão de distribuição de informações em tempo real, através da internet. O usuário que faz um requisição para o link que possui este formato, recebe como resposta conteúdos atualizados e disponibilizados neste padrão através de XML.

Atom (Atom Syndication Format): outro padrão também utilizado para distribuição de conteúdos na internet, e surgiu como alternativa ao padrão RSS.

APÊNDICE D - REGRAS DE NEGÓCIO

R1 – É necessário informar um e-mail válido que tenha acesso.

R2 – O token de login deve bater com código informado pelo usuário.

R3 – Validar link de nova assinatura de feed.

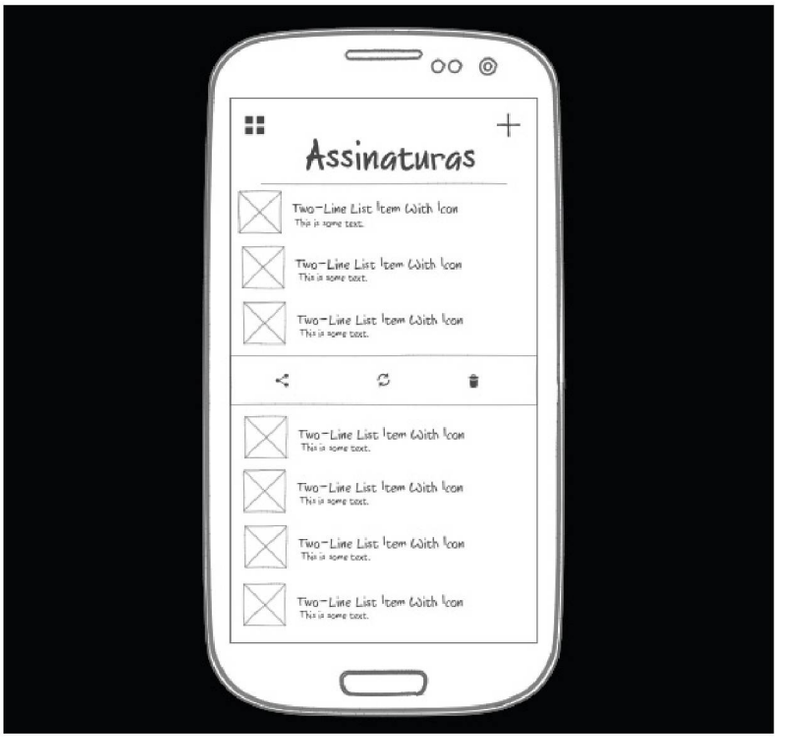
R4 – Manter pressionado um item de uma lista por dois segundos

R5 – O sistema valida se o endereço já existe no banco de dados

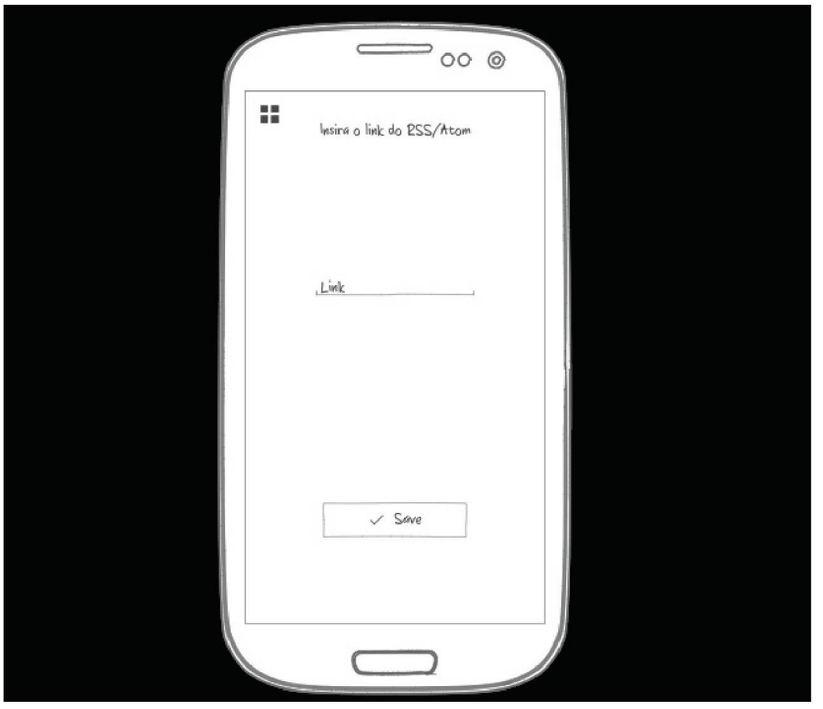
R6 – Necessário haver ao menos uma assinatura para exibir lista

APÊNDICE E - PROTÓTIPOS DE INTERFACES

DV001 – TELA COM LISTA DE FEEDS



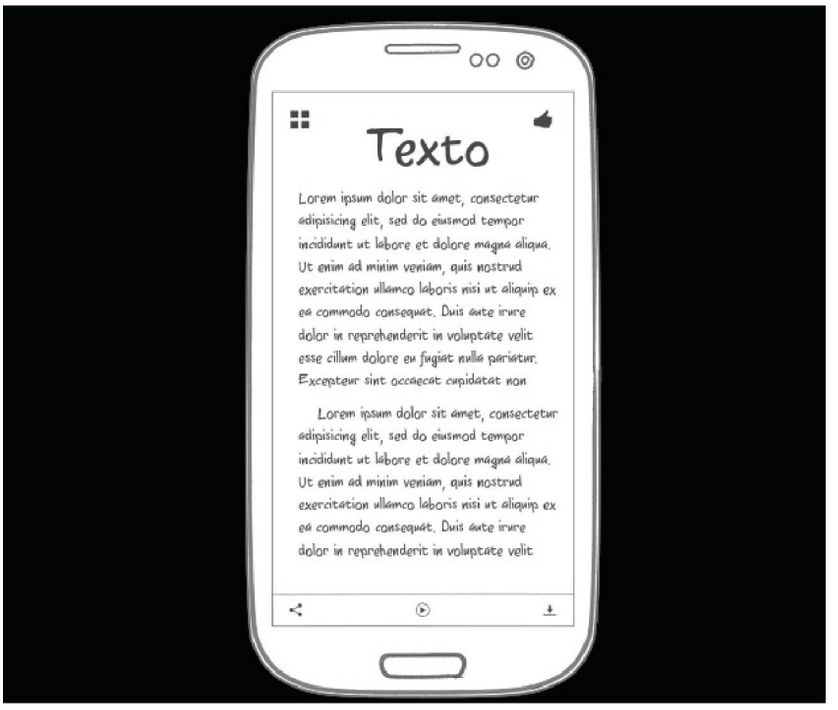
DV002 – TELA ADICIONA FEED



DV003 – MENU

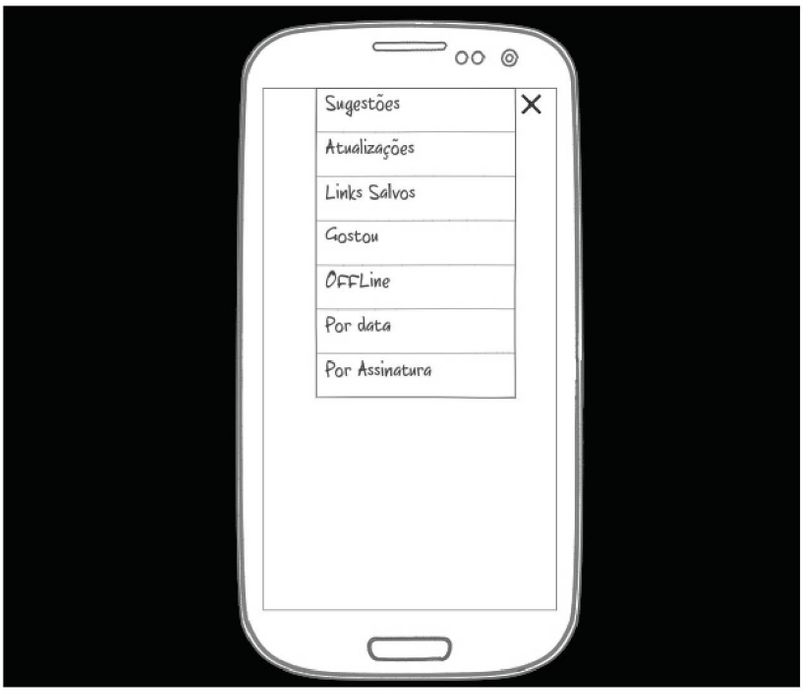
DV006 – TELA LISTA DE CONTEÚDOS

DV007 – TELA DE VISUALIZAÇÃO DE TEXTOS



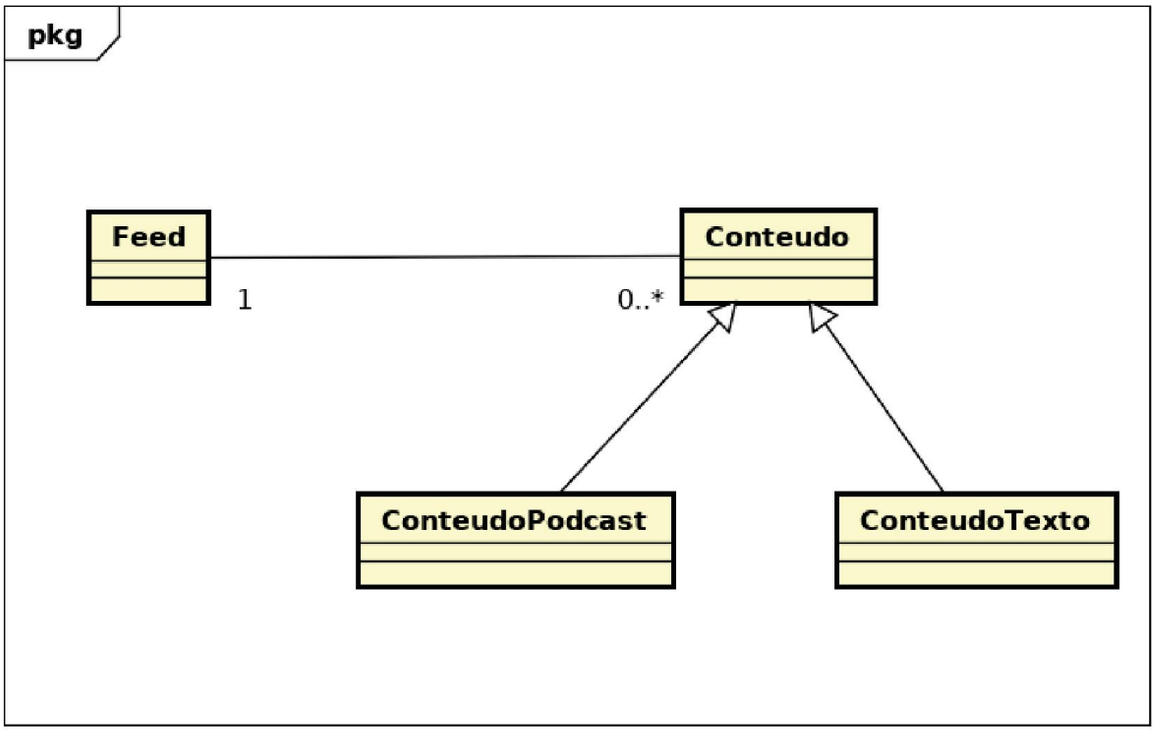
DV008 – TELA DE VISUALIZAÇÃO DE PODCAST

DV009 – TELA DE FILTROS

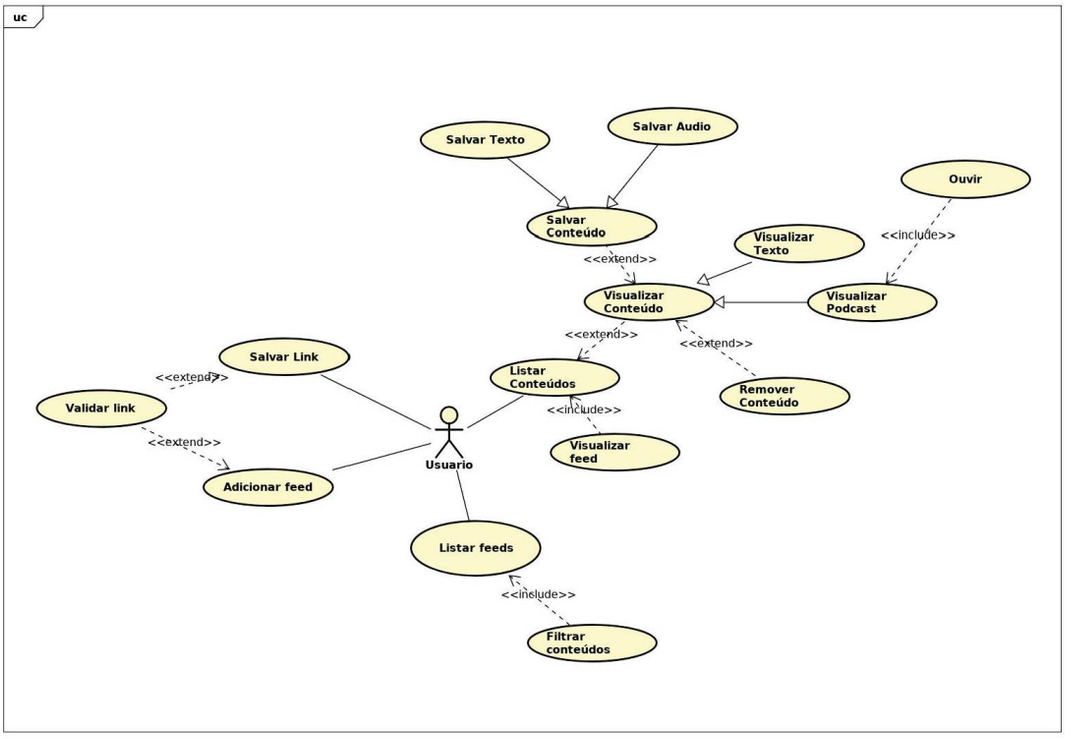


DV010 – TELA VISUALIZAR FEED

APÊNDICE F – MODELO DE OBJETOS NEGOCIAIS



APÊNDICE G – CASOS DE USO



Especificação do Caso de Uso

UC-001 Listar feeds

Descrição

Este caso de uso descreve a tela que lista as assinaturas de feeds.

Data View

DV001 – Tela lista de assinaturas feeds.

Pré-condições

O usuário estar logado no sistema.

Pós-condições

Não há nenhuma pós condição.

Ator primário

Usuário.

Fluxo de eventos principal

1. O sistema carrega o botão de **MENU**.
2. O sistema carrega o botão **FILTROS**.
3. O sistema carrega o botão **ADICIONAR.**
4. O sistema carrega a lista de feeds.(**R6) (A5**)
5. O sistema exibe a tela. (**DV001**).
6. O usuário pressiona o botão **ADICIONAR. (A1**)(**A2**)(**A3**)(**A4**)
7. O sistema chama o caso de uso **UC002.**
8. O caso de uso é finalizado.

Fluxos alternativos

A1. O usuário pressiona o botão **MENU**

1. O sistema exibe a tela **DV003**

A2. O usuário pressiona o botão **FILTROS**

1. O sistema exibe a tela **DV009**

A3. O usuário pressiona um item da lista de assinaturas por dois segundos (**R4**)

1. O sistema altera o item para inativo
2. O sistema atualiza o item no banco de dados.
3. O sistema exibe a mensagem “Feed inativado”.

A4. O usuário clica em uma das opções da lista.

1. O sistema chama o caso de uso **UC003.**

A5. O usuário não possui assinaturas

1. Sistema exibe a mensagem “Nenhum feed!”

Especificação do Caso de Uso

UC-002 Adicionar Feed

Descrição

Este caso de uso descreve a tela de adicionar um endereço de feed.

Data View

DV002 – Tela adiciona feed

Pré-condições

O usuário estar logado no sistema e seguido o fluxo principal do UC-003

Pós-condições

O usuário visualizar a mensagem “Feed adicionado.”

Ator primário

Usuário.

Fluxo de eventos principal

1. O sistema carrega o título da aplicação.
2. O sistema carrega o campo **LINK**.
3. O sistema carrega o botão **SALVAR.**
4. O sistema exibe a tela (**DV002**).
5. O usuário insere um endereço de RSS no campo **LINK**.
6. O usuário clica no botão **SALVAR.** (**E1**) (**R3**)
7. O sistema salvar o endereço no banco de dados.
8. O sistema exibe a mensagem “Feed adicionado”.
9. O caso de uso é finalizado.

Fluxos de exceção

E1. O usuário informa um endereço inválido

O sistema exibe a mensagem “Endereço inválido.”

Especificação do Caso de Uso

UC-003 Visualizar Feed

Descrição

Este caso de uso descreve a tela de visualizar um feed.

Data View

DV0010 – Tela visualizar feed

Pré-condições

O usuário estar logado no sistema e seguido o fluxo principal do UC-002

Pós-condições

O usuário visualizar a mensagem “Feed adicionado.”

Ator primário

Usuário.

Fluxo de eventos principal

10. O sistema carrega o título da aplicação.
11. O sistema carrega o campo **TÍTULO**.
12. O sistema carrega o campo **DESCRIÇÃO**.
13. O sistema carrega o botão **VOLTAR.**
14. O sistema carrega o botão **VISITAR.**
15. O sistema carrega o campo **INATIVAR.**
16. O sistema exibe a tela **DV010**.
17. O usuário clica no botão **VISITAR. (A1) (A2)**
18. O sistema chama a aplicação de navegação na internet.
19. O caso de uso é finalizado.

Fluxo alternativo

A1. O usuário pressiona o botão VOLTAR

O sistema chama o **UC001**

Especificação do Caso de Uso

UC-004 Listar conteúdos

Descrição

Este caso de uso descreve a tela que lista os conteúdos dos feeds.

Data View

DV006 – Tela lista de conteúdos.

Pré-condições

Não há nenhuma pré-condição.

Pós-condições

Não há nenhuma pós condição.

Ator primário

Usuário.

Fluxo de eventos principal

1. O sistema carrega o botão de **MENU**.
2. O sistema carrega o botão **FILTROS**.
3. O sistema carrega a lista de conteúdos.(**A1**) (**R6**)
4. O sistema exibe a tela. (**DV005**).
5. O usuário clica em uma das opções da lista. (**A2**)(**A3**)(**A4**)
6. O sistema chama o caso de uso **UC005.**
7. O caso de uso é finalizado.

Fluxos alternativos

A1. O usuário não cadastrou nenhum feed

1. O sistema exibe a Mensagem “Nenhum feed adicionado”.

A2. O usuário pressiona o botão **MENU**

1. O sistema exibe a tela **DV003.**

A3. O usuário pressiona o botão **FILTROS**

1. O sistema exibe a tela **DV009**

A4. O usuário pressiona um item da lista de conteúdos por dois segundos(**R4**)

1. O sistema altera o estado do conteúdo para “lido”
2. O sistema atualiza o conteúdo no banco de dados.
3. O usuário exibe a mensagem “Conteúdo marcado como lido”. **(E1)**

Fluxos de exceção

E1. Erro ao salvar

1. O sistema exibe a mensagem “Não foi possível atualizar. Tente novamente.”

Especificação do Caso de Uso

UC-004 Visualizar conteúdo

Descrição

Este caso de uso descreve a tela para visualização de um conteúdo.

Data View

DV007 – Tela de visualização de conteúdo - textos.

Pré-condições

O usuário estar logado no sistema e ter selecionado um item da lista de conteúdos.

Pós-condições

Não há nenhuma pós condição.

Ator primário

Usuário.

Fluxo de eventos principal

1. O sistema carrega o botão de **MENU**.
2. O sistema carrega o botão **VOLTAR**.
3. O sistema carrega o botão **VISITAR**.
4. O sistema carrega o botão **SALVAR**.
5. O sistema carrega o conteúdo.
6. O sistema exibe a tela. (**DV007**). (**A1**)
7. O sistema atualiza o status do conteúdo para lido.
8. O usuário pressiona no botão **VISITAR**. (**A2**)(**A3**)(**A4**)
9. O sistema chama a aplicação para navegação na internet.
10. O caso de uso é finalizado.

Fluxos alternativos

A1. O conteúdo é do tipo podcast

1. O sistema faz a requisição do áudio no endereço do feed.
2. O sistema carrega o botão os botões do **PLAYER**
3. O sistema exibe a tela **DV008**
4. O usuário pressiona o botão **TOCAR**
5. O sistema inicia o stream do áudio. (**E2**)

A2. O usuário pressiona o botão **SALVAR**

1. O sistema faz salva o conteúdo no banco de dados local. (**E1**)
2. O sistema apresenta a mensagem “Conteúdo salvo!”.

A3. O usuário pressiona o botão **MENU**

1. O sistema exibe a tela **DV003**

A4. O usuário pressiona o botão **VOLTAR**

1. O sistema chama o **UC003**

Fluxos de exceção

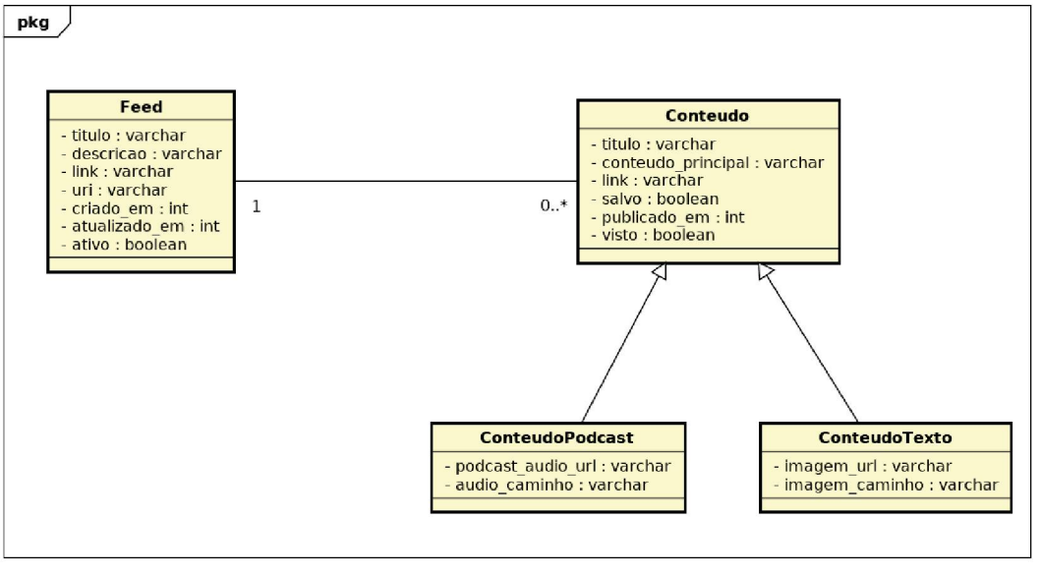
E1. Erro ao salvar

3. O sistema exibe a mensagem “Não foi possível atualizar. Tente novamente.”

E2. Erro ao tocar

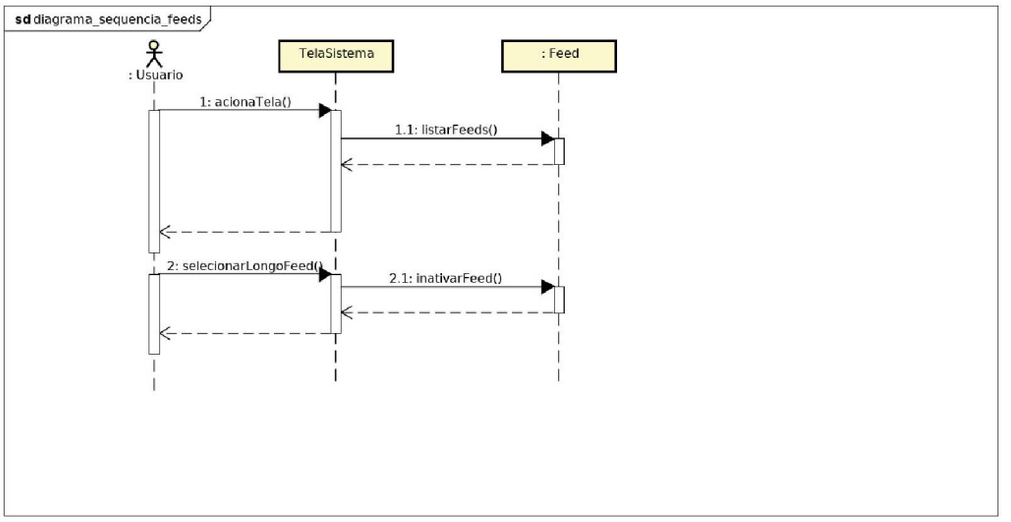
1. O sistema exibe a mensagem “Não foi possível reproduzir . Tente novamente.”

APÊNDICE H – MODELO DE OBJETOS

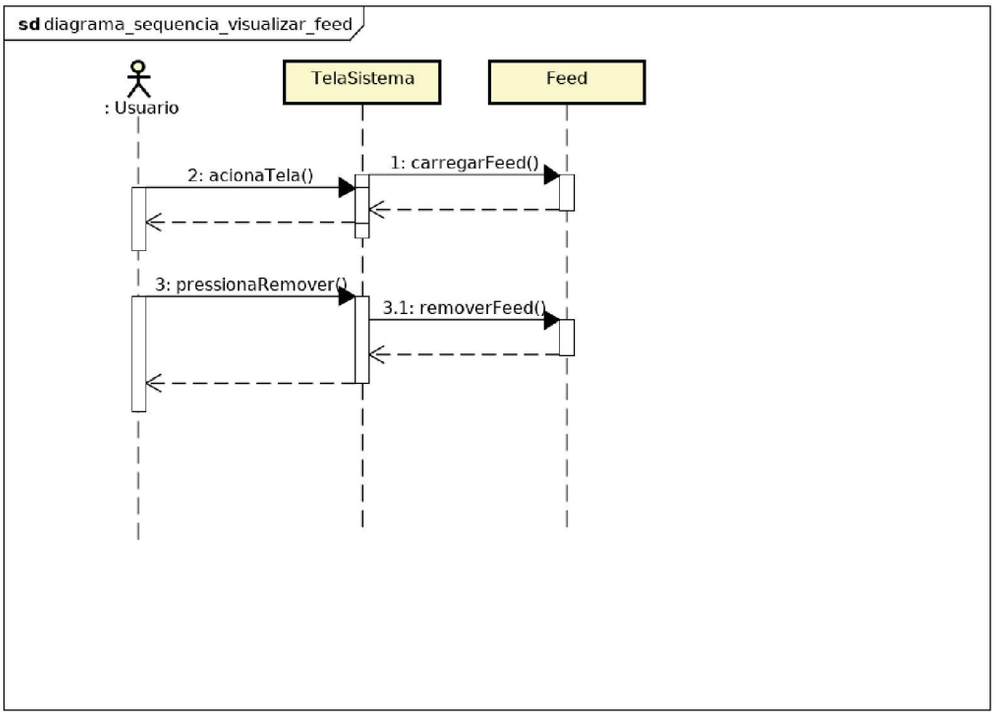


APÊNDICE I – DIAGRAMAS DE SEQUÊNCIA

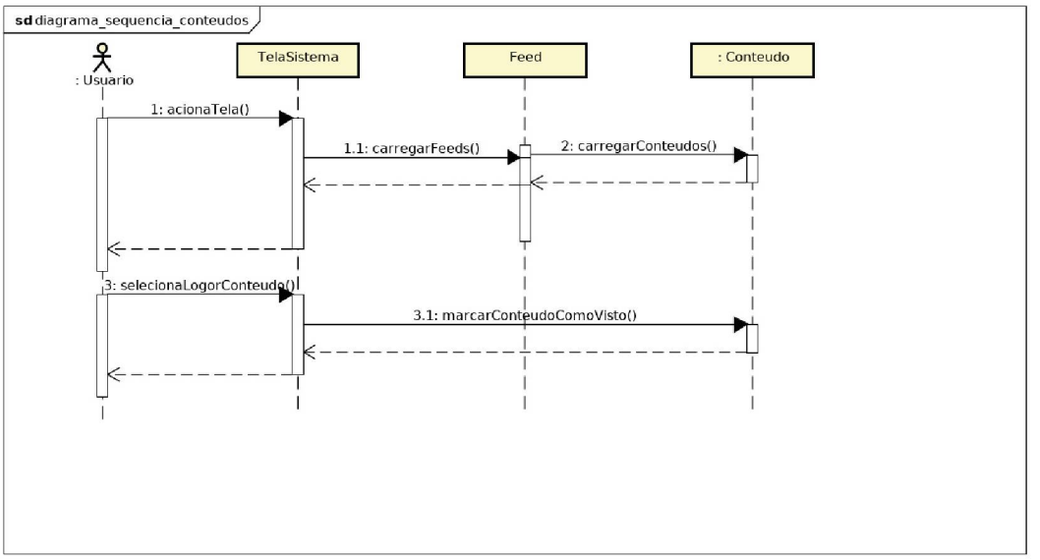
CASO DE USO LISTAR FEEDS



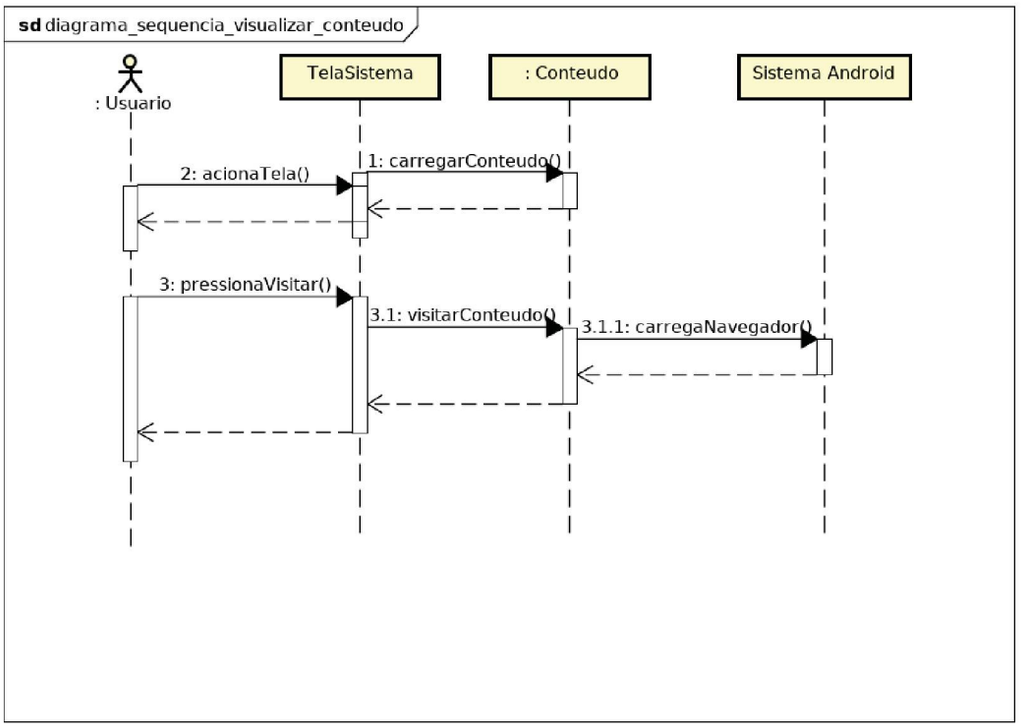
CASO DE USO VISUALIZAR FEEDS



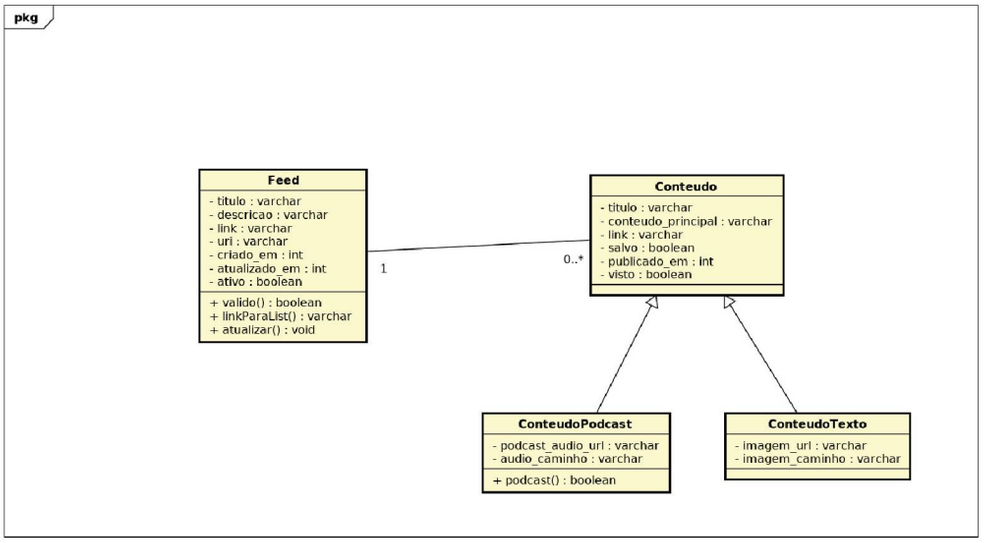
CASO DE USO LISTAR CONTEÚDOS



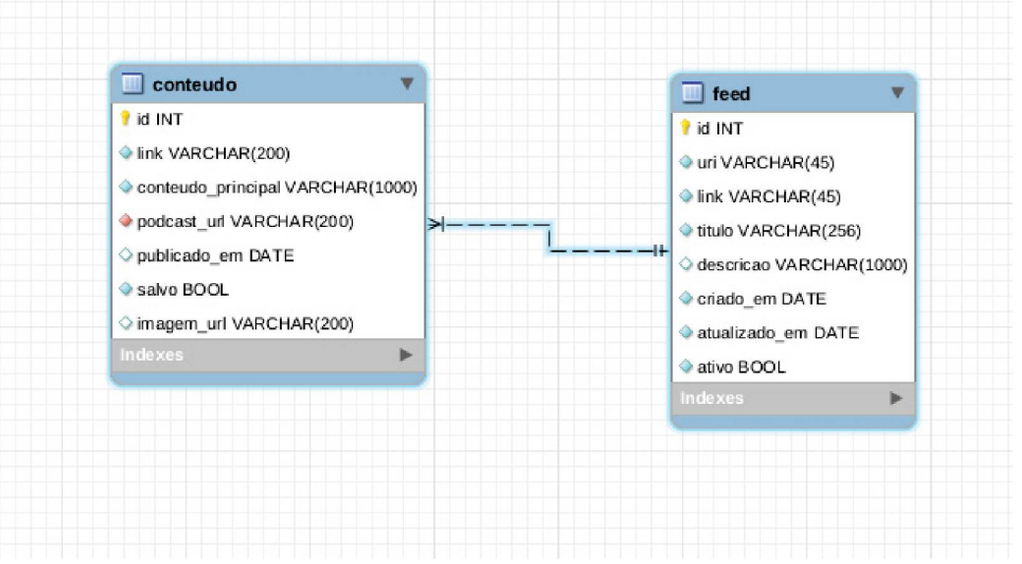
CASO DE USO VISUALIZAR CONTEÚDO



APÊNDICE J – MODELO DE OBJETOS



APÊNDICE K – MODELO FÍSICO DE DADOS



APÊNDICE M – CASOS E LOG DE TESTES

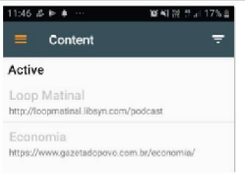
Especificação do Caso de Teste
Content

Caso de Uso	Listar feeds		
Pré-condições	O usuário deve ter adicionado ao menos um feed.		
Elaborador	Marcelo Marzola Leite	**Data de Elaboração**	10/02/2019
Executor		**Data de Execução**	27/11/2019

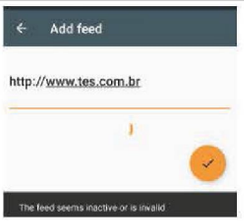
N.º	Pré-condições	Entrada	Ação	Resultado Esperado	Evidência de teste
01	O usuário deve ter ao menos um feed adicionado.		Abrir o aplicativo ou selecionar a opção "Feeds" no menu principal	O sistema deve exibir a DV001 com com o título "Feeds ativos" e uma lista de feeds ativos.	
02	O usuário deve ter ao menos um feed inativo		Abrir o aplicativo ou selecionar a opção "Feeds" no menu principal	O sistema deve exibir a DV001 com com o título "Feeds inativos" e uma lista de feeds inativos.	

Especificação do Caso de Teste
Content

Caso de Uso	Adicionar Feed		
Pré-condições	1. O usuário deve passar pela tela de lista feeds conforme o UC-001.		
Elaborador	Marcelo Marzola Leite	**Data de Elaboração**	10/02/2019
Executor		**Data de Execução**	27/11/2019

N.º	Pré-condições	Entrada	Ação	Resultado Esperado	Evidência de teste
1	O usuário deve pressionar o botão ADICIONAR na tela DV001	O usuário deve informar um endereço de feed no formulário	Pressionar botão ADICIONAR	O sistema deve chamar o UC-001 apresentado a DV001 com o feed adicionado pelo usuário	

Especificação do Caso de Teste Content		

2	O usuário deve pressionar o botão ADICIONAR na tela DV001	O usuário deve informar uma url inválida no formulário	Pressionar botão ADICIONAR	O sistema apresentar a mensagem "Link inválido"	
3	O usuário deve pressionar o botão ADICIONAR na tela DV001	O usuário deve informar um endereço de feed inativo ou inválido no formulário	Pressionar botão ADICIONAR	O sistema apresentar a mensagem "Endereço inválido ou feed inativo"	

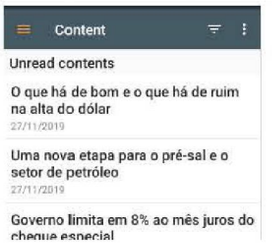
Especificação do Caso de Teste Content		

Caso de Uso	Visualizar feed		
Pré-condições	1. O usuário deve ter selecionado um item no UC-001.		
Elaborador	Marcelo Marzola Leite	**Data de Elaboração**	10/02/2019
Executor		**Data de Execução**	27/11/2019

N.º	Pré-condições	Entrada	Ação	Resultado Esperado	Evidência de teste
01	O usuário ter selecionado um item no UC-001			O sistema deve exibir a tela DV-003 com o feed selecionado pelo usuário.	

Especificação do Caso de Teste Content		

Caso de Uso	Listar conteúdos		
Pré-condições	2. O usuário ter seleciona a opção Conteúdos na DV-003.		
Elaborador	Marcelo Marzola Leite	**Data de Elaboração**	10/02/2019
Executor		**Data de Execução**	

N.º	Pré-condições	Entrada	Ação	Resultado Esperado	Evidência de teste
01	O usuário ter selecionado a opção "Conteúdos" no menu na DV-003. O usuário deve ter adicionado ao menos um feed no UC-002			O sistema apresentar um ícone que indica o carregamento de novos conteúdos	

Especificação do Caso de Teste Content					
02	O usuário ter selecionado a opção "Conteúdos" no menu na DV-003. O usuário deve ter adicionado ao menos um feed no UC-002 e já passado pelo UC-004			O sistema deve apresentar a lista de conteúdos já atualizados e a mensagem "Sem conteúdos novos".