Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Gerenciador de mídias sociais para redação jornalística

Victória Goularte Resende

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Orientador
Prof. Dr. Edison Ishikawa

Coorientador
Prof. Dr. Benedito Medeiros Neto

Brasília
2021

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

RR433g

Resende, Victória Goularte
Gerenciador de mídias sociais para redação jornalística / Victória Goularte Resende; orientador Edison Ishikawa; co orientador Benedito Medeiros Neto. -- Brasília, 2021.
77 p.

Monografia (Graduação - Engenharia de Computação) -- Universidade de Brasília, 2021.

1. aplicativo. 2. MVP. 3. gerenciador de mídias sociais. 4. redação jornalística. I. Ishikawa, Edison, orient. II. Neto, Benedito Medeiros, co-orient. III. Título.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Gerenciador de mídias sociais para redação jornalística

Victória Goularte Resende

Monografia apresentada como requisito parcial
para conclusão do Curso de Engenharia da Computação

Prof. Dr. Edison Ishikawa (Orientador)
CIC/UnB

Prof. Dr. Maria de Fátima Ramos Brandão
UnB

Prof. Dr. Zanei Ramos Barcellos
UnB

Prof. Dr. João José Consta Gondim
Coordenador do Curso de Engenharia da Computação

Brasília, 7 de Maio de 2021

Dedicatória

Dedico esse trabaho aos professores, pesquisadores e jornalistas.

Agradecimentos

Agradeço aos professores Edison e Benedito Medeiros pela indispensável orientação e pela oportunidade de realizar esse trabalho. Agradeço à minha família por todo apoio e força, por acreditarem em mim e não me deixarem desistir. Em especial aos meus pais, Flávio e Silésia, aos meus irmãos, ao André, à minha tia Simone e à minha prima Marcela. E sobretudo, agradeço a Deus por ter iluminado toda minha vida e minha jornada acadêmica com saúde e forças para chegar até aqui.

Este projeto de graduação foi apoiado pela Fundação de Apoio à Pesquisa do Distrito Federal (FAP-DF), por meio do projeto de pesquisa científica intitulado "Sistemas de informações organizacionais flexíveis baseados em processos de negócio com orientação semântica contextual", processo número 00193-00000096/2019-78.

Resumo

Este trabalho apresenta uma prova de conceito para um gerenciador de mídias sociais com foco em uma redação jornalística experimental. Sendo possível, com esse gerenciador, monitorar e analisar informações provenientes de ações e engajamentos sobre publicações feitas pela redação nas mídias sociais. E assim, a partir desses dados, inferir posicionamentos e ações benéficas à modelos e métodos de publicações que promovam as notícias e a redação jornalística. Além disso, o gerenciador também permite realizar publicações de notícias com destino à uma ou mais plataformas de mídias sociais instantâneamente, bem como consultar e recuperar todo o conteúdo publicado, independente da plataforma de mídia, em momentos posteriores à publicação. A metodologia utilizada é uma abordagem exploratória, ágil e incremental para produção de um produto viável mínimo ou *Minimum Viable Product (MVP)*, e compreendeu entrevista com jornalista. No trabalho foi desenvolvido como MVP um aplicativo para dispositivos móveis que atua como o gerenciador de mídias sociais.

Palavras-chave: gerenciador de mídias sociais, aplicativo, MVP, redação jornalística

Abstract

This paper presents a proof of concept for a social media manager with a focus on experimental journalistic writing. With this manager, it is possible to monitor and analyze information from actions and engagements on publications made by the newsroom on social media. And so, from these data, infer beneficial positions and actions to models and methods of publications that promote news and journalistic writing. In addition, the manager also allows you to make news publications destined for one or more social media platforms instantly, as well as consult and retrieve all the content published after the publication. The methodology used is an exploratory, agile and incremental approach to the production of one (MVP), and included an interview with the journalist. In the work it was developed, for the proof of concept, as (MVP) an application for mobile devices that acts as the manager of social media.

Keywords: social media manager, application, MVP, newsroom

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

API Application Programming Interface.

CMS Content Management System.

DSR Design Science Research.

FAC/UnB Faculdade de Comunicação/Jornalismo da Universidade de Brasília.

HTTP Hypertext Transfer Protocol.

JSON JavaScript Object Notation.

JVM Máquina Virtual Java.

KMS Knowledge Management System.

MVP Minimum Viable Product.

MVVM Model-View-ViewModel.

MVVMi Model-View-ViewModel-Interactor.

REST Representational State Transfer.

SDK Software Development Kit.

SMMS Social Media Management System.

SoC Separation of Concerns.

UI User Interface.

UnB Universidade de Brasília.

URL Uniform Resource Locator.

W3C World Wide Web Consortium.

Capítulo 1

Introdução

Nos últimos anos a indústria de notícias tem observado uma mudança inquestionável e radical devido à expansão das mídias sociais, que alterou as normas e padrões de uma redação jornalística tradicional. O crescente nível de interesse e eficácia nas mídias sociais por todo o mundo, se deve, basicamente, ao modo como mantém todas as pessoas conectadas. Da mesma forma, no campo das notícias, como as redações jornalísticas, isso se torna muito interessante no sentido de aproximar-se do leitor e manter-se conectado à eles [10].

Com as mídias sociais tem-se o aumento da velocidade de publicação e de consumo das notícias, além de mexer com o perfil da audiência, que também se sente capaz de discutir e opinar sobre os conteúdos publicados por um jornal [11]. O padrão de comunicação deixa de ser unidirecional [2], aquele que não considera as implicações do leitor, como quando ele expressa insatisfação, interferências políticas, religiosas, etc, e passa a ser de cunho interativo [12], que enfatiza a participação do público e aponta a necessidade dos jornalistas de encorajar e solicitar *feedbacks*.

Assim, jornais impressos têm perdido cada vez mais espaço [13] e as organizações de comunicação enfrentam um grande desafio devido a esse crescimento tecnológico que também tem como consequência a concorrência pela execução e distribuição da notícia somado à diminuição de leitores pagantes.

Além disso, a transitoriedade das mídias sociais tem exigido dos profissionais do jornalismo qualificação constante para acompanhar as necessidades do público e estar apto a produzir conteúdo da melhor forma e em diversos formatos, sempre consciente que determinadas habilidades podem cair em desuso com o tempo [11].

Partindo-se desses pontos, é notório a necessidade de ingresso e engajamento dos jornais nas mídias sociais e sua flexibilização para acompanhar as constantes mudanças que acontecem nessas plataformas. E, segundo [14], a trajetória dessas evoluções tecnológicas são também acompanhadas pelo Conselho Editorial do Campus Multiplataforma, do

jornal experimental digital da FAC/UnB. Em que, para cada plataforma de mídia social, dentre elas o *Twitter, o Instagram, Site, YouTube, Facebook e WhatsApp*, dentre outros, as publicações são feitas com conteúdos próprios, produzidos em linguagem adequada a ela e ao seu público, por grupos de jornalistas dedicadas a cada mídia social.

A intenção deste trabalho, é inspirada em cenários como o acima citado, em que são utilizadas várias plataformas de mídia sociais para divulgação de suas notícias, e surge a partir da identificação de um problema de pesquisa: como centralizar o gerenciamento de várias plataformas de mídias sociais utilizadas como meio de comunicação e divulgação de notícia por uma redação jornalística a fim de contribuir e minimizar os impactos das constantes mudanças que cada uma dessas plataformas pode sofrer?

Considerando que gerenciar cada conta de mídia social pelo jornal requer dedicação e qualificação de um número cada vez maior de profissionais, torna-se útil concentrar em uma única ferramenta, ou em outras palavras, em um único *login*, a divulgação de notícias e o acompanhamento das mídias sociais utilizadas. Uma vez que a divulgação realizada através da plataforma única se adaptaria ao contexto de cada plataforma de mídia de destino; e o acompanhamento trataria do engajamento com os leitores e seguidores do jornal unindo informações de todas as mídias sociais utilizadas, possibilitando ao jornal medidores e quantificadores que possibilitam fazer inferências benéficas sobre suas publicações e seu relacionamento com os seus leitores.

Portanto, a proposta deste trabalho é demonstrar a possibilidade de um gerenciador de mídias sociais capaz de centralizar a divulgação e publicação de notícias para as n plataformas de mídias sociais distintas utilizadas pelo jornal, que permite ainda adaptar o conteúdo da notícia à mídia de destino, sendo capaz também de monitorar o engajamento para com as publicações realizadas através desse gerenciador à fim de tornar mais eficiente o veículo jornalístico, aprimorar o ingresso do jornal nas mídias sociais e facilitar a conexão bidirecional entre jornal e leitores. Além de dispor de busca e consulta por publicações feitas através do gerenciador, independente da mídia social em que foi publicada, para que os profissionais tenham controle sobre seu acervo em mídias sociais.

Para isso, propõe-se a implementação de um aplicativo gerenciador de mídias sociais para dispositivos móveis integrado às mídias sociais como MVP, com o intuito de tornar o artefato mais acessível e portátil, de modo a facilitar o processo de publicação e monitoramento das notícias. Seu desenvolvimento foi baseado no método de pesquisa DSR e a metodologia utilizada para desenvolvimento do artefato foi baseada em métodos ágeis de desenvolvimento de *software* posteriormente compreendidos.

1.1 Objetivos

1.1.1 Objetivos gerais

O objetivo geral deste trabalho é demonstrar a possibilidade de um gerenciador de mídias sociais para redação jornalística através de um aplicativo para dispositivos móveis como um produto viável mínimo (MVP).

1.1.2 Objetivos específicos

Para garantir o objetivo geral, os seguintes objetivos específicos foram definidos:

1. identificar como é feito o gerenciamento de mídias sociais e suas dificuldades;
2. analisar e comparar gerenciadores de mídias sociais disponíveis;
3. desenvolver o design da proposta de prova de conceito em forma de um aplicativo para *mobiles*;
4. implementar o MVP;

1.2 Estrutura do Documento

A estrutura do documento se baseia no apresentado em [15] e está organizada da seguinte maneira:

- **Capítulo 1:** Expõe e contextualiza o problema. Introduz os objetivos da proposta;
- **Capítulo 2:** Apresenta a metodologia utilizada para guiar o desenvolvimento da pesquisa e do artefato;
- **Capítulo 3:** Expõe de forma breve as tecnologias utilizadas para o desenvolvimento e que dão suporte à análise das mídias sociais. Descreve o processo de revisão da literatura feita para obter embasamento teórico necessário para realizar o trabalho. Também apresenta artefatos semelhantes;
- **Capítulo 4:** Descreve a arquitetura da solução, as etapas de implementação e detalha os módulos e funcionamento do artefato;
- **Capítulo 5:** Expõe os atefatos e propõe possíveis trabalhos futuros.

Capítulo 2

Revisão Bibliográfica

Este capítulo apresenta o referencial teórico necessário para compreensão do trabalho e traz ferramentas com propósitos semelhantes que foram utilizados como referências para o seu desenvolvimento.

2.1 Jornalismo interativo

2.1.1 Origem

A ideia de "jornalismo interativo"atualmente remete à ideia de produção e publicação de notícias através da internet, o que infere, de certa maneira, que a prática de jornalismo está aberta a todos. Mas segundo [16] o jornalismo interativo não nasceu com a internet. Em sua origem mais profunda, deve-se considerar que o jornalismo interativo nasceu através de contatos por cartas, telefones ou mesmo através de pessoas que conseguiam contato com o jornal através de meios de comunicação tradicionais onde levantavam sugestões de pautas, denúncias, críticas, dentre outros.

2.1.2 Características

Segundo [17] o jornalismo interativo concentra-se como característica inerente da comunicação atual, a comunicação online e em rede entre jornais e leitores, em benefício do conteúdo das notícias, ou seja, dos modos de aproveitamento da participação e contribuição dos leitores (audiência) para as notícias no jornalismo. É o fato de o jornalismo superar o modelo convencional de transmissão como emissor-meio-mensagem-receptor, e adotar o receptor como um agente que colabora, participa, produz, coleta e dissemina informações [18].

2.1.3 O jornalismo interativo e a Internet

Partindo dos tópicos anteriores em que destacamos brevemente a origem e as principais características do jornalismo interativo, percebemos um fenômeno que, mesmo não sendo novo, ganhou expressiva visibilidade através da Internet, uma vez que esse meio de comunicação dispõe de facilidade quanto à produção, cooperação e partilha da informação de uma maneira quase que instantânea e globalizada com destino à leitores sem restrições quanto à etnia, localização, classe social, idade, dentre outros. E, sendo assim, podemos inferir uma maior participação, partilha e colaboração por parte dos leitores, fazendo com que o jornalismo se preocupe com isso e dê mais atenção a esses pontos e ao que é retornado pelo público.

2.1.4 O jornalismo interativo e as mídias sociais

Com a Internet também surgiram as mídias sociais. E com o crescimento acelerado dessas mídias sociais, os jornais tiveram de se adaptar a este ecossistema otimizando seus modos de publicação e engajamento com os leitores para potencializar suas distribuições de notícias e conquistar mais leitores oferecendo mais espaços para comentários, participações e colaborações.

Neste contexto, segundo [19] evidenciam-se conceitos como "*mass self communication*"– coexistência entre a comunicação individual e de massas – e "comunicação em rede"– uma fusão da comunicação interpessoal e em massa, ligando audiências, emissores e editores sob uma matriz de relacionamentos em rede.

No cenário específico das mídias sociais, o número de usuários nas mídias como *Facebook*, *Twitter* e *Instagram* contribuem ativamente com o jornalismo interativo, colocando em evidência o relacionamento com as fontes, o alcance, a distribuição e compartilhamento de notícias, a rapidez da informação, a fidelização e colaboração dos leitores. Para [20] a utilização das mídias sociais é um dado adquirido, nomeadamente como agregador de notícias, como plataforma de informação e até como uma forma de captar leitores. Neste sentido, reconhece-se a existência de vantagens ao utilizar esses meios. Isso, de certa maneira obriga o jornalista a entender e imergir no mundo das mídias sociais se engajando com essa interação para com os leitores e verificando informações dos usuários que são relevantes para determinadas situações, além de aumentar sua responsabilidade quanto ao que publica, uma vez que se sua publicação contém informações não verdadeiras, com omissões ou plágios ele pode ser duramente criticado e exposto nos meios de interação da própria publicação na mídia social.

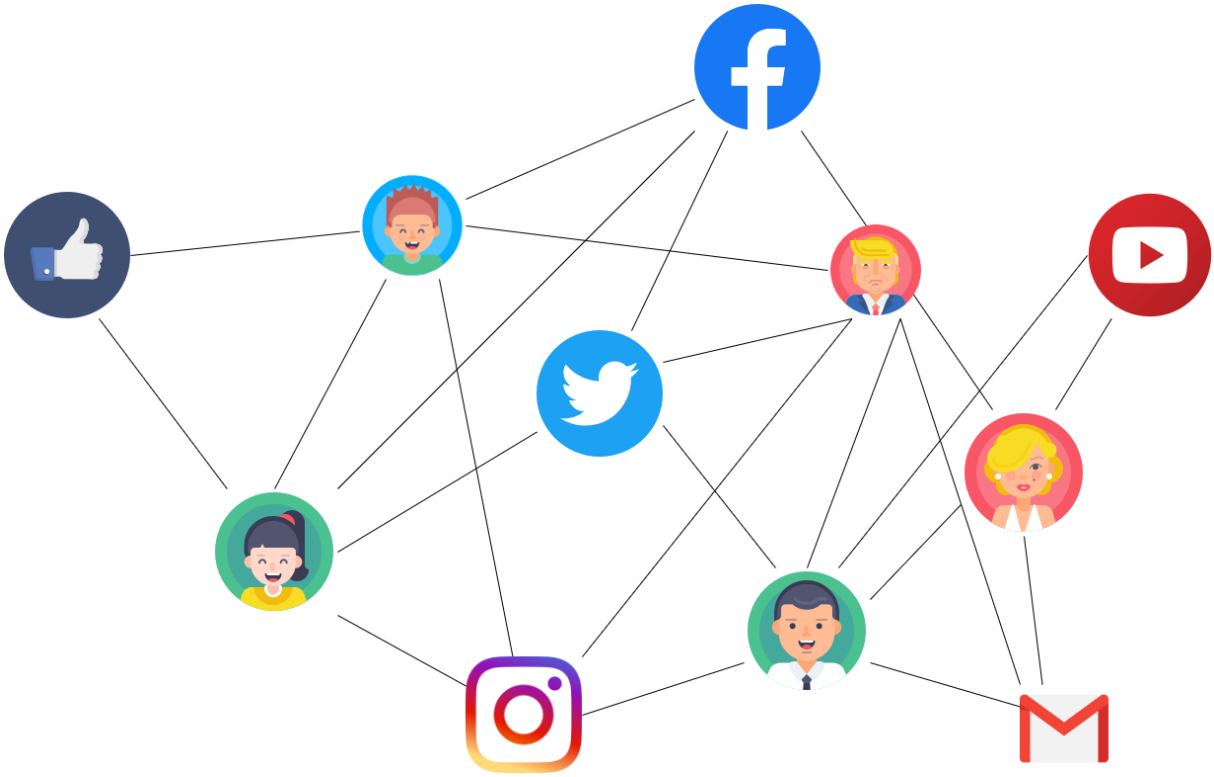


Figura 2.1: Grafo de interações de um jornalismo participativo. Adaptado de [2].

2.2 Aplicações para gerenciamento de mídias sociais

Nesta seção são apresentadas algumas plataformas de gerenciamento de mídias sociais disponíveis atualmente no mercado para quem trabalha com mídias sociais e precisa centralizar as informações em um único canal para facilitar a gestão das informações e publicações através desse ecossistema. Isso porque manter postagens atualizadas em tantas mídias pode ser um desafio, assim como monitorar o engajamento e interação sobre inúmeras publicações. E, baseado no cenário abordado nesse trabalho, é perceptível que essas ferramentas sejam benéficas para o meio jornalístico no cenário de jornalismo interativo. Portanto, é fundamental conhecer algumas das ferramentas já existentes.

As aplicações aqui apresentadas não são necessariamente com foco em jornalismo, mas em *marketing digital*, não excluindo seu uso no jornalismo, uma vez que apresenta funcionalidades muito convenientes para o gerenciamento de mídias sociais de uma redação jornalística. Antes de listarmos as ferramentas, para um melhor entendimento, fizemos a Tabela 2.1 que compara pontos importantes ao marketing digital de acordo com [21] e que adequamos ao cenário do jornalismo.

Os profissionais de marketing [3] reportam que aplicativos de marketing inteligentes coletam dados dos visitantes mais precisos alimentando melhores características de

Tabela 2.1: Principais benefícios de uma aplicação de gerenciamento de mídias sociais para o Marketing Digital e o Jornalismo.

Benefício	**Marketing Digital**	**Jornalismo**
Mais oportunidades de impacto	Oportunidades de apresentar uma marca, produto ou serviço para o cliente. Cada uma das mídias sociais de hoje são focadas em determinados formatos e a escolha da melhor opção está ligada ao seu público-alvo.	Oportunidade de apresentar uma notícia ou artigo adaptando-as ao formato escolhido para publicação destinando a notícia a um nicho de leitores alvo.
Construção de autoridade	Publicar conteúdo de qualidade é uma maneira eficiente de se posicionar como autoridade. Importante mostrar que a sua empresa entende do assunto. O relacionamento é muito importante para a fidelização de clientes.	Publicar notícias confiáveis e com qualidade é uma maneira de se posicionar como autoridade. Importante passar segurança, verdade e embasamento para a fidelização dos leitores.
Mensuração	Poder de mensuração de campanhas e análise de retorno sobre o investimento. Painéis de análise oferecem uma ajuda adicional na geração de insights sobre a origem da audiência.	Poder de mensuração de campanhas e análise de participação e colaboração sobre a notícia. Painéis de análise oferecem uma ajuda adicional na geração de insights sobre a origem da audiência.
Satisfação do cliente	Os usuários utilizam as mídias sociais não apenas para relatar coisas boas, mas frustrações também. As pessoas querem se sentir bem atendidas e terem uma boa experiência com as marcas que se relacionam.	Os leitores utilizam as mídias sociais para expressar apreço ou insatisfação sobre determinadas notícias. Os leitores gostam de se sentir representados e ter uma boa experiência com a fonte de informações a que recorre.
Prospecção de novos consumidores	Atrair os consumidores por meio de conteúdo de qualidade.	Atrair leitores por meio de conteúdo de qualidade e verídico.

personalização. O que também pode ser usado ao jornalismo no seu contexto em suas próprias características de personalização, destacando leitores-alvo, horários melhores de publicações de notícias, dentre outros. O gráfico abaixo mostra a relação entre um conteúdo publicado *versus* qualidade e o tempo. No gráfico é possível ver a importância de ferramentas para gestão do conteúdo quando ele aponta que aplicações que coletam dados de intenção mais precisos e alimentam melhores perfis de personalização, e que, consequentemente tornam mais fácil o consumo por parte dos visitantes, possuem conteúdos mais sofisticados e com qualidade.

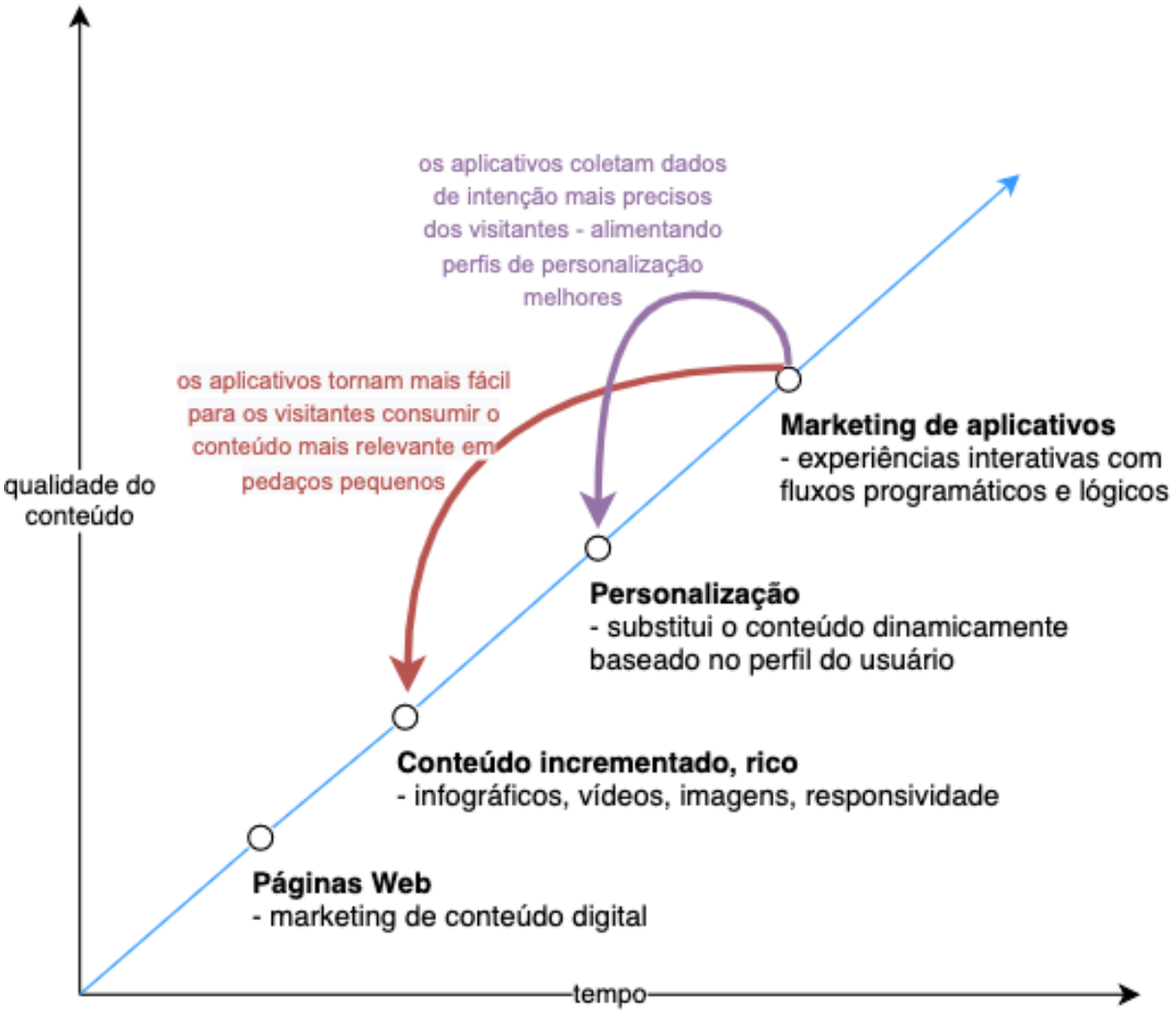


Figura 2.2: Comparação de um conteúdo publicado com qualidade pelo tempo. Adaptado de [3].

Hoje, segundo [9], as ferramentas de gerenciamento mais utilizadas e tomadas como referencial nesse trabalho podem ser vistas na tabela comparativa Tabela 2.2.

Tabela 2.2: Aplicativos Gerenciadores de Mídias Sociais. Tabela adaptada de informações retiradas de [9].

	Buffer	**Hootsuite**	**Later**
Capacidade de gerenciamento de mais de uma plataforma de mídia social	Sim	Sim	Não, apenas Instagram
Permite agendar publicações futuras	Sim	Sim	Sim
Gera sumarizado para números de curtidas, compartilhamentos e comentários	Sim	Sim	Sim
Gera sumarizado para números de curtidas, compartilhamentos e comentários	Sim	Sim	Sim
Aplicativo pago	Sim	Sim	Sim
Possui suporte à utilização por mais de um usuário pessoa na mesma conta	Não	Sim	Não
Permite adicionar termos importantes à publicação que possibilitam filtros e pesquisas	Não	Não	Sim
Persiste todas as publicações feitas através do gerenciador para buscas e consultas futuras	Não	Não	Não

Para [22] que compara e analisa o estado da arte de gerenciadores de mídias sociais e considera os aplicativos Buffer e Hootsuite acima citados, é importante, dentre alguns pontos, analisar:

1. a quantidade de mídias sociais integradas para gerenciamento;
2. o nível de suporte técnico fornecido pelo gerenciador;
3. os atributos de segurança, devido à sensibilidade das informações que estão sendo gerenciadas, além do *login* seguro na plataforma gerenciadora; e
4. o valor da plataforma gerenciadora.

Em sua conclusão, [22] diz que nenhum aplicativo estudado atende a todas esses pontos, incluindo os anteriormente apresentados: Buffer e Hootsuite. Mas afirma que para um usuário específico, o estado da arte de um gerenciador é aquela destinada e elaborada especificamente para ele.

Assim sendo, com base nas informações acima e no que diz [22] e [3], a personalização de um sistema gerenciador para um usuário alvo, com as características convenientes à esse usuário alvo, o torna o sistema gerenciador ideal para seu uso. E como o foco deste trabalho é um gerenciador de mídias sociais para uma redação jornalística, analisamos os pontos positivos e negativos dos aplicativos apresentados e extraimos funcionalidades talvez úteis e convenientes com foco no jornalismo como candidatas à uma prova de conceito, excluindo as funcionalidades desnecessárias. E o compilado das funcionalidades extraídas

Tabela 2.3: Tabela comparativa entre aplicativos gerenciadores de Mídias Sociais e aplicativo gerenciador para Prova de Conceito .

	Buffer	**Hootsuite**	**Later**	**SMMS - MVP**
Capacidade de gerenciamento de mais de uma plataforma de mídia social	Sim	Sim	Não, apenas Instagram	Sim
Permite agendar publicações futuras	Sim	Sim	Sim	Sim
Gera sumarizado para números de curtidas, compartilhamentos e comentários	Sim	Sim	Sim	Sim
Aplicativo pago	Sim	Sim	Sim	Não
Possui suporte à utilização por mais de um usuário pessoa na mesma conta	Não	Sim	Não	Sim
Permite adicionar termos importantes à publicação que possibilitam filtros e pesquisas	Não	Não	Sim	Sim
Persiste todas as publicações feitas através do gerenciador para buscas e consultas futuras	Não	Não	Não	Sim

e candidatas ao gerenciador Minimum Viable Product (MVP), que aqui chamamos de Social Media Management System (SMMS), podem ser comparadas na tabela Tabela 2.3 com os aplicativos listados.

2.3 Graph API

Partindo-se das seções anteriores, é notório a conveniência de se usar as mídias sociais para um jornalismo participativo. Também é possível perceber as dificuldades para se administrar tantos canais de mídias sociais e mantê-los, o que alternativamente pode ser resolvido utilizando-se de ferramentas capazes de concentrar as informações das mídias sociais envolvidas e compilá-las de maneira personalizada e descomplicada. Mas, para se ter acesso às informações de qualquer rede social é preciso que essas ferramentas gerenciadoras das mídias se integrem e consumam dados de cada plataforma de rede social,

que são disponibilizados pelas próprias plataformas através de Application Programming Interface (API) abertas em servidores.

A Graph API [23] é a principal forma de inserir e extrair dados da plataforma do *Facebook*. É uma API baseada em ()HTTP, o que significa que funciona com qualquer linguagem de programação que tenha uma biblioteca HTTP, na qual aplicativos terceiros podem usar para consultar dados de forma programática, publicar novas histórias, gerenciar anúncios, carregar fotos e realizar diversas outras tarefas nas mídias *Facebook* e *Instagram*.

A Graph API é voltada à cientistas da computação, sendo chamada a própria documentação da ferramenta como "*Facebook* para desenvolvedores - (ou *Facebook for Developers*). Uma vez que conhecer métodos e liguagens capazes de integrar serviços de APIs requer considerável conhecimento em liguagens de computação.

Esse nome, Graph API, é baseado na idéia de um "grafo social". De acordo com [24], a Graph API veio como uma forma revolucionária de compreender e acessar a vida social das pessoas. Cada informação recuperada da plataforma *Facebook*, inferindo-se sempre que pode ser tanto da rede social *Facebook* quanto da *Instagram*, seja ela uma informação simples, com poucos dados, ou uma mais complexa, com uma quantidade de dados considerável, representa um elemento desse "grafo social". Sua composição pode ser descrita através da seguinte analogia de um grafo, e representada por algo como ilustra a Figura 2.3:

- **nós:** representam "objetos"individuais. Como, por exemplo, um Usuário, Foto, Página ou Comentário;
- **bordas:** representam as conexões entre uma coleção de objetos e um objeto único. Como, por exemplo, Fotos em uma Página ou Comentários em uma Foto;
- **campos:** representam os dados a respeito de um objeto. Como, por exemplo, a data de aniversário do Usuário ou o nome de uma Página.

Em resumo, estrutura de uma API Graph é tal que, uma API que usa nós para obter dados sobre objetos individuais, que possuem identificadores únicos. Usa bordas para obter a coleção de objetos associados a um nó, ou para publicar objetos para essas coleções. Por fim, usa-se campos para especificar que todos os dados que se deseja sejam incluídos nas respostas.

Segundo [24], a Graph API representa uma revolução no fornecimento de dados em grande escala. E tornou as ofertas da empresa e os dados gerados por seus usuários economicamente viável. É capaz de converter pessoas e seus gostos, conexões, locais, atualizações, redes, histórias em, literalmente, "objetos- que em ciência da computação, é uma referência a um local da memória de máquina que possui um valor.

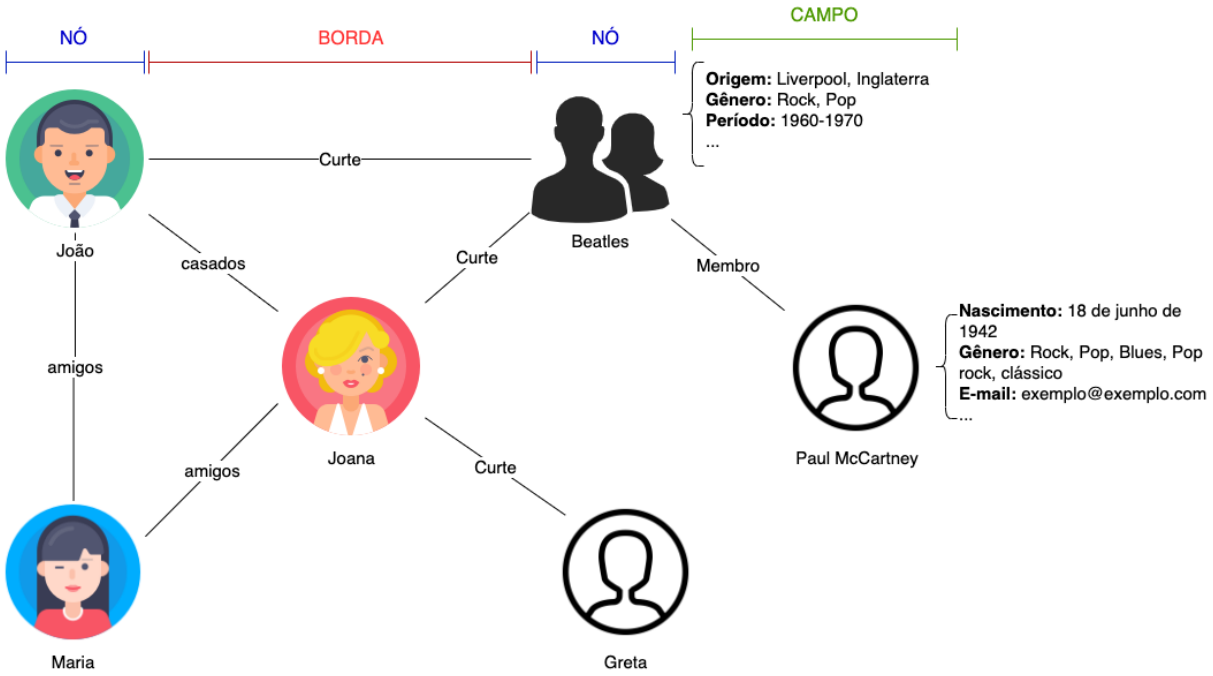


Figura 2.3: Graph API representada por "Grafo social". Adaptado de [4].

Todos os pontos de acesso do grafo, através de solicitações HTTP para se obter ou inserir informações, exigem um tipo de autorização: um *token* de acesso. Devido a isso, toda requisição feita à API deve conter um *token*. Os *tokens* de acesso podem ser recuperados programaticamente através de uma interface *web* ou de aplicações móveis, no caso de a solicitação de acesso ser feita por um aplicativo, em que o usuário ingresso na rede social autoriza que a aplicação tenha acesso ao token para acessar informações específicas do usuário Figura 2.4.

2.4 Desenvolvimento de aplicativos móveis

Aplicativos móveis são *softwares* desenvolvidos para serem usados em dispositivos móveis, como *smartphones* e *tablets*. Por meio desses aplicativos, esses dispositivos se transformam em um grande pacote de ferramentas que permitem, por exemplo, como dito em [25], acessar mídias sociais, conteúdos educacionais, entretenimento, jogos eletrônicos, localização geográfica, acesso a banco e outros serviços. Dessa forma, os aplicativos customizam os dispositivos de acordo com interesses e necessidades do usuário de uma maneira totalmente portátil, acessível, flexível, disponível e de baixo custo. Além de quase que instantâneamente permitir que os usuários resolvam problemas através de poucos toques na tela do telefone. Ao contrário de aplicações desenvolvidas para computadores que foram, de certa forma, substituídos pelos celulares justamente por possuírem tantas ferra-

Figura 2.4: Interface de solicitação de acesso ao usuário.

mentas disponíveis, por não serem tão fáceis de transportar e acessar a todos os momentos e especialmente pelo custo elevado se comparado aos dispositivos móveis.

Os aplicativos móveis podem ser classificados como a parte visual de um todo de uma aplicação, aquilo que o usuário consegue interagir. Nele há processamento tanto de informações inseridas pelo usuário quanto recebidas por um servidor através da Internet. As solicitações feitas através do aplicativo a um servidor são feitas através de uma linguagem para Internet que classificamos como lado cliente (*client-side*) do servidor, são solicitações que aguardam respostas de um servidor. Essas respostas podem ser resultado de outro processamento feito entre o que chamamos de lado *server-side*, que são liguagens de código que somente o servidor entende e processa para mandá-las ao *client-side*. A grosso modo, *client-side* é conhecido como *front-end* e *server-side* conhecido como *backend*.

2.4.1 Aplicativos Android

O Android é uma plataforma desenvolvida pela Google voltada para dispositivos móveis, totalmente aberta e livre (*Open Source*), que foi divulgada em 5 de novembro de 2007 [26]. Por ser uma plataforma aberta, torna o seu desenvolvimento mais barato e consequentemente mais acessível ao consumidor final, e, por esse e outros motivos, é o sistema operacional mais usado hoje para dispositivos móveis [27].

2.4.2 Arquitetura da plataforma

A plataforma possui uma arquitetura separada por camadas, sendo cada camada responsável por seus processos. A Figura 2.5 a seguir mostra a maioria dos componentes da plataforma Android.

As camadas da arquitetura:

- **Aplicativos de sistema:** funcionam como aplicativos para os usuários e fornecem capacidades principais que os desenvolvedores podem acessar pelos próprios aplicativos.
- **Android framework:** nesta camada está disponível o conjuntos de recursos fornecidos pelo próprio sistema operacional Android através de APIs desenvolvidas em liguagem Java. Esses recursos representam as funções básicas do telefone como sistema de visualização, gerenciador de recursos, de localização, dentre outros. A camada de aplicações acessa a Android Framework utilizar essas funções.
- **Bibliotecas nativas e Android Runtime:** cada aplicativo executa seus próprios processos em uma instância própria de execução, o que aumenta a velocidade de resposta e otimiza o consumo de energia. Nesta camada, o código compilado e interpretado por uma Máquina Virtual Java (JVM) disponível é transcrito em instruções nativas do dispositivo. Aqui também estão as bibiliotecas nativas do Android, escritas em liguagem C e C++.
- **Camada de abstração de hardware:** dispõe de interfaces que acessam as capacidades de *hardware* do dispositivo através de bibliotecas. Um exemplo de acesso ao hardware através dessas bibliotecas são os módulos de câmera ou Bluetooth.
- **Kernel Linux:** representa a base do sistema. É fundamental para execução do Android Runtime de cada aplicativo auxiliando em funcionalidades como gerenciamento de memória de baixo nível, recursos de segurança, dentre outros.

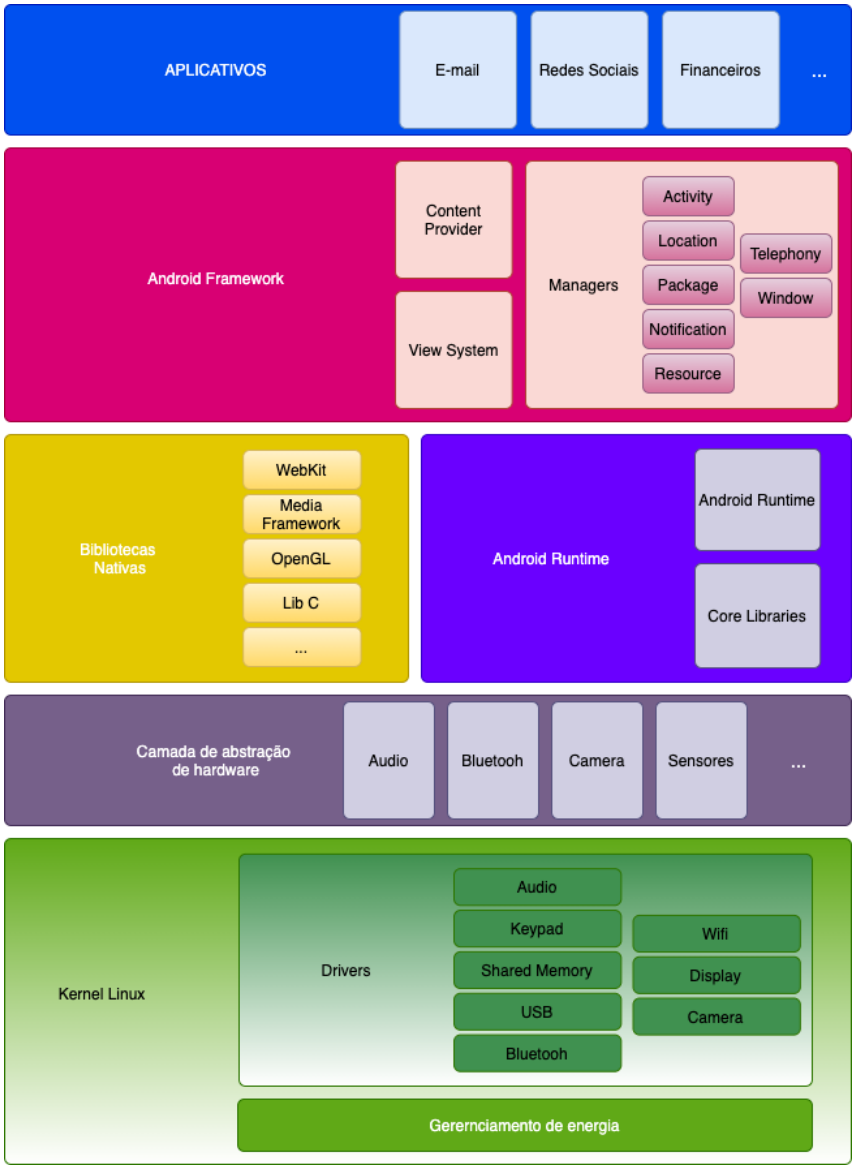


Figura 2.5: A pilha de software do Android. Adaptado de [5].

2.4.3 Conceitos básicos

Um aplicativo Android possui quatro principais componentes que são a base de conhecimento para se desenvolver uma aplicação, são eles:

- **Atividades** ou *Activities*: representa uma das classes mais importantes que compõe um aplicativo. É a classe encarregada de gerenciar toda a parte de User Interface (UI). Um aplicativo pode conter várias atividades que podem ser apresentadas ao usuário como janelas de telas inteiras, janelas flutuantes, dentre outros. A atividade tem também uma importante subclasse chamada de fragmento. Os fragmentos são usados para melhor modularizar o código, permitindo construir interfaces de mais

sofisticadas para telas maiores e ajuda a dimensionar a aplicação entre telas pequenas e grandes. Essas classes possuem ciclos de vida e é de extrema importância entendê-lo para se desenvolver um aplicativo. Esses ciclos são classificados normalmente em quatro estados:

  - Se uma atividade estiver em primeiro plano da tela (na posição mais alta de uma pilha de atividades), ela está *ativa* ou em *execução* . Normalmente, é a atividade com a qual o usuário está interagindo no momento.
  - Se uma atividade perdeu o foco, mas ainda é apresentada ao usuário, ela é *visível*. Acontece quando essa atividade é deixada em segundo plano no dispositivo, por exemplo.
  - Se uma atividade for completamente ocultada por outra atividade, ela será *interrompida* . Ela pode ainda reter as informações do estado, no entanto, não será mais visível para o usuário, portanto, sua janela fica oculta e, com frequência, será eliminado pelo sistema quando a memória for necessária em outro lugar.
  - O sistema elimina a atividade da memória pedindo que ela seja concluída ou simplesmente eliminando seu processo, *destruindo-o*. Quando a atividade é exibida novamente para o usuário, ela deve ser completamente reiniciada e restaurada ao seu estado anterior.

- **Serviços:** é um componente capaz de realizar operações de execução longas sem a necessidade de interagir com o usuário através de interfaces e permite que outros aplicativos sejam executados sem que seu processo seja interrompido. Um aplicativo que reproduz músicas, por exemplo, executa um serviço para que a lista de músicas toque ainda que não esteja sendo executado em primeiro plano.
- **Receptores de transmissão broadcast:** são componentes responsáveis por receber e tratar eventos provenientes do próprio sistema ou de outras aplicações. Esses receptores são configurados de modo que sejam executados até que o evento seja recebido, tratando-o e notificando à aplicação que evento ocorreu, disparando novas intruções fornecidas pelo receptor.
- **Provedores de conteúdo:** responsável pelo gerenciamento de acesso a um conjunto de dados. Eles encapsulam os dados e os fornecem aos aplicativos por meio de uma única interface, permitindo que duas aplicações diferentes compartilhem os mesmos dados.

Uma guia sobre desenvolvimento para aplicações Android é disponibilizado gratuitamente pela própria Google [8]. Em seus documentos aconselha padrões de projetos

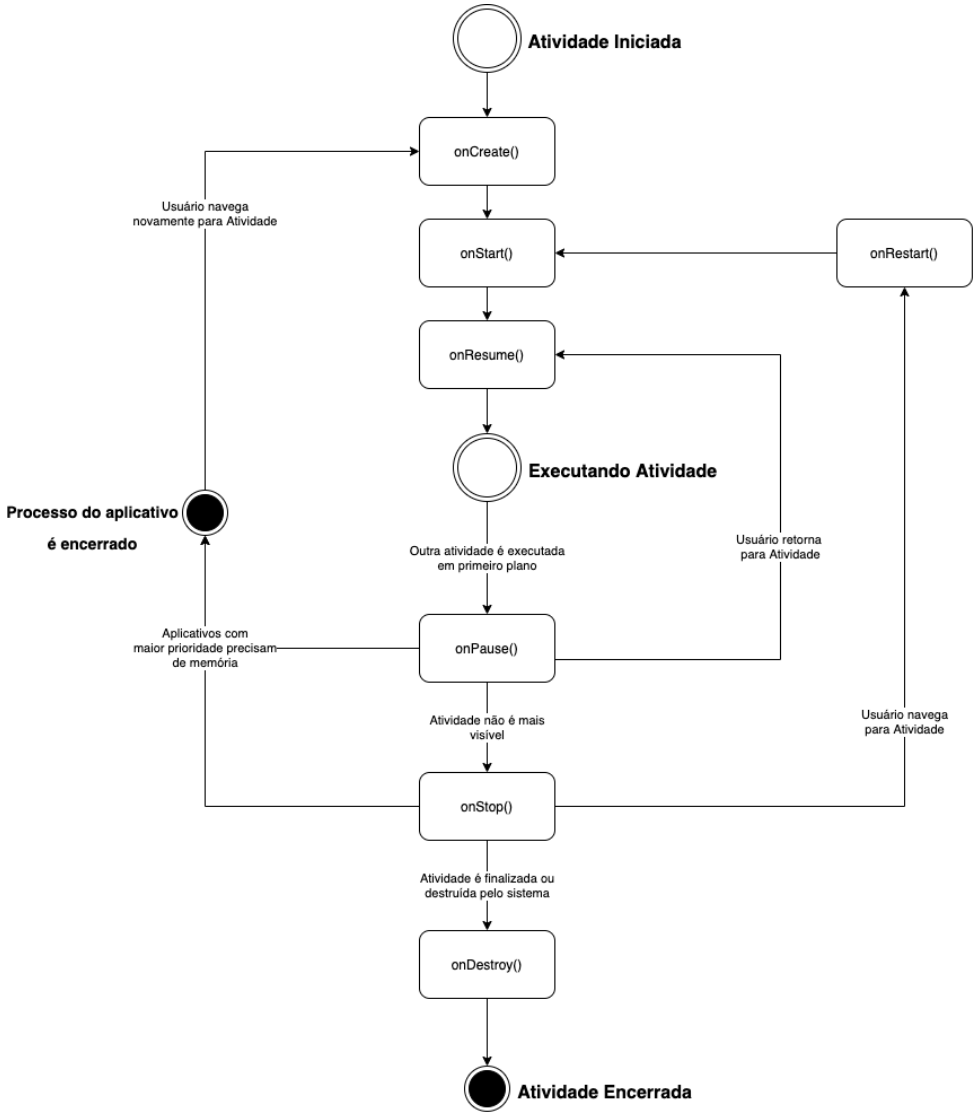


Figura 2.6: Ciclos de vida de uma atividade. Adaptado de [5].

que devem ser adotados de modo a possibilitar um desenvolvimento de aplicativos de qualidade, robustos, testáveis e de fácil manutenção. Alguns elementos importantes são recomendados e podem ser utilizados para atender a esses critérios. Destacamos alguns:

- **Linguagem de programação Kotlin**: segundo [28] a linguagem de programação Kotlin oferece uma sintaxe expressiva, um sistema de tipos robusto e intuitivo e o suporte de um ótimo conjunto de ferramentas, juntamente com uma interoperabilidade natural com códigos, bibliotecas e frameworks Java. Pode ser compilada para bytecode Java, portanto você pode ser usada em todos os lugares em que Java é utilizada. Possui um compilador eficiente e uma pequena biblioteca-padrão, Kotlin praticamente não impõe nenhuma sobrecarga em tempo de execução.

- **Componentes com reconhecimento de ciclo de vida de outros componentes:** executam ações em resposta a uma mudança no status do ciclo de vida de outro componente, como atividades e fragmentos. Esses componentes ajudam a produzir códigos mais organizados e, normalmente, mais leves e mais fáceis de manter.
- **Livedata:** uma classe armazenadora de dados observáveis. Diferente de um observável comum, o LiveData conta com reconhecimento de ciclo de vida, ou seja, ele respeita o ciclo de vida de outros componentes do aplicativo, como atividades, fragmentos ou serviços. Esse reconhecimento garante que o LiveData atualize apenas os observadores de componente do aplicativo que estão em um estado ativo no ciclo de vida. Por exemplo, ao girar a tela do dispositivo, o ciclo de vida da página apresentada é observado e o dado não é carregado novamente.
- **ViewModel:** uma classe ViewModel foi projetada para armazenar e gerenciar dados relacionados à UI considerando o ciclo de vida. A classe ViewModel permite que os dados recuperados de um servidor, por exemplo, sobrevivam às mudanças de configuração, como a rotação da tela e não requisitem novamente aquelas informações ao servidor.

2.5 NoSQL - Banco de dados não relacionais

Bancos de dados NoSQL são criados para modelos de dados específicos e têm esquemas flexíveis para a criação de aplicativos modernos. Os bancos de dados NoSQL são reconhecidos por sua facilidade de desenvolvimento, funcionalidade e performance em escala [29].

Bancos de dados NoSQL usam uma variedade de modelos de dados para acessar e gerenciar os dados. Esses tipos de banco de dados são otimizados especificamente para aplicativos que exigem modelos de grande volume de dados, baixa latência e flexibilidade. Esses requisitos são atendidos com a ausência de algumas restrições de consistência de dados dos outros bancos. Como no exemplo demonstrado em [29] de esquema para um bando de dados simples:

- Em um banco de dados relacional, um registro de livro é normalmente disfarçado (ou "normalizado") e armazenado em tabelas separadas, e os relacionamentos são definidos por restrições de chave primária e externa. Neste exemplo, a tabela Livros têm colunas para ISBN, Título do livro e Número da edição, a tabela Autores têm colunas para AuthorID e Nome do autor e, finalmente, a tabela Author-ISBN tem colunas para AuthorID e ISBN. O modelo relacional é projetado para permitir que o banco de dados imponha a integridade referencial entre as tabelas no banco

de dados, normalizadas para reduzir a redundância e geralmente otimizadas para armazenamento.

- Em um banco de dados NoSQL, um registro de livro é normalmente armazenado como um documento JSON. Para cada livro, o item, o ISBN, o Título do livro, o Número de edição, o Nome do autor e o AuthorID são armazenados como atributos em um único documento. Neste modelo, os dados são otimizados para desenvolvimento intuitivo e escalabilidade horizontal.

Os banco de dados não relacionais propõem enfraquecer o requisito de consistência de dados, atribuído em bancos de dados relacionais [30], focando em melhorar a disponibilidade e a tolerância a criar mais partições. Esse novo conceito considera um cenário que provê transações distribuídas, tolerância a falhas de consistência, e replicação otimista em um sistema distribuído. Os NoSQL são ideais para muitos aplicativos para dispositivos móveis, *Web* e jogos, que exigem bancos de dados flexíveis, com esquemas flexíveis que permitem um desenvolvimento rápido e iterativo, escaláveis, projetados para serem escalados horizontalmente usando clusters distribuídos de hardware, em vez de escalá-los verticalmente adicionando servidores caros e robustos, de alta performance, otimizado para modelos de dados específicos e padrões de acesso que permitem maior performance, e funcionais, fornecendo APIs e tipos de dados altamente funcionais criados especificamente para cada um de seus respectivos modelos de dados, para proporcionar boas experiências aos usuários.

Para o cenário deste trabalho, em que se espera as mesmas características acima citadas, como flexibilidade de esquemas, escalabilidade, ausente de gastos com servidores, funcionalidade através de acesso de dados por APIs e de alta perfomance com alta disponibilidade, a abordagem NoSQL tornou-se uma forte candidata à ser utilizada [31].

Um banco de dados de chave-valor é um tipo de banco de dados não relacional que usa um método de chave-valor simples para armazenar dados. Um banco de dados de chave-valor armazena dados como um conjunto de pares de chave-valor em que uma chave funciona como um identificador exclusivo. A chave e os valores podem ser qualquer coisa, desde objetos simples até objetos compostos complexos [29]. Esses bancos de dados de chave-valor são altamente particionáveis e permitem escalabilidade horizontal que outros tipos de bancos de dados não conseguem alcançar.

O modelo apresenta a proposta que permite que o desenvolvedor efetue a persistência de dados totalmente livre de definições de estrutura para esquema, o que representa uma grande vantagem por ser considerado bastante simples permite a visualização do banco de dados como uma grande tabela *hash*. A informação do conteúdo é armazenada e um índice em forma de um tipo primitivo de dado é usado para mapeá-lo. O conteúdo pode ser armazenado em qualquer formato, seja uma string, inteiro, matriz, ou objeto. Pelo

fato de ter uma chave de indexação para mapear um valor, a inserção e a recuperação se torna mais fácil e com interface simplificada. Na maioria dos casos a interface segue dois principais métodos (inserir e recuperar) que seguem o padrão a seguir [32]:

- Put(chave, valor): método que insere um registro
- Get(chave): método que recupera um registro

2.6 Firebase

O Firebase é uma plataforma móvel do Google que funciona conhecidamente hoje como *Backend as a Service*, ou seja, "*backend* como um serviço" [33]. O que seria o lado servidor de uma aplicação totalmente em nuvem.

Essa plataforma auxilia desenvolvedores no desenvolvimento de, especialmente, aplicativos, por dispor de já de algumas funcionalidades comuns à *backends* com fácil integração e entendimento. Uma das principais funcionalidades utilizadas é a *Firebase Auth* que oferece diversos métodos de autenticação, inclusive e-mail/senha, provedores de terceiros, como o Google ou Facebook, ou o uso direto do seu sistema de contas. Nesse sentido, a interface é criada no próprio aplicativo e através de APIs do Firebase os dados são persitidos e cadastrados adequadamente em nuvem. Outra funcionalidade que se destaca é o *Realtime Database*, um banco de dados que proporciona uma solução eficiente e de baixa latência para aplicativos. Trata-se de um banco de dados não relacional, NoSQL, representado por uma árvore JSON em que os dados inseridos via APIs da plataforma estão armazenados nos nodos. Um grande benefício do Firebase Realtime Database é que ele já possui um sistema de sincronização instantânea implementado, fazendo com que, caso ocorra uma modificação no banco, todos os aplicativos que tenham a referência daquele item, sejam automaticamente atualizados, ao invés de trabalhar com requisição e resposta normalmente utilizado em outros bancos.

2.7 Direcionamento do trabalho

Tomando como base, nesta seção, todos os tópicos estudados e descritos nesse capítulo, e relacionando-os como uma cadeia de necessidades, por exemplo, vimos que o jornalismo participativo se destaca e evolui com maior representatividade no ecossistema de mídias sociais, mas que mídias sociais, por existirem várias e cada um em seu formato, requerem, como forma de facilitar e monitorar, além de outros benefícios, aplicações de gerenciamento de mídias sociais, e ainda que essas mídias sociais se integram às plataformas desenvolvedoras das mídias sociais através de APIs. Nesse sentido, como forma de

agregação à esse cadeia torna-se útil desenvolver uma aplicação para dispositivos móveis capaz de gerenciar mídias sociais personalizadas para uma redação jornalística.

Capítulo 3

Metodologia

Neste capítulo são descritos quais os procedimentos foram feitos para implementar o trabalho proposto: implementar um Minimum Viable Product (MVP) como uma prova de conceito de um gerenciador de mídias sociais como um aplicativo para *mobiles* descrito no objetivo geral. O método de pesquisa utilizado foi o Design Science Research (DSR) [7] e a metodologia utilizada para produção e desenvolvimento do aplicativo foi a metodologia ágil, com *framework* Scrum [6].

A DSR em [7] é descrita como uma abordagem com objetivos de estudar, pesquisar e investigar o artificial e seu comportamento. É uma metodologia que busca consolidar conhecimento sobre projetos e sistemas que já existem e desenvolver soluções para melhorá-los, criar novos artefatos que trazem benefícios à comunidade visando preencher a lacuna existente entre pesquisa e prática.

Quanto à metodologia ágil utilizada, segundo [34], é descrita como uma metodologia que reúne conjuntos de práticas e métodos de desenvolvimento, criados e desenvolvidos ao longo das últimas décadas, que têm por objetivo tornar o desenvolvimento de software rápido, com custo controlável e melhorar a qualidade do software. A *framework Scrum* compõe a metodologia ágil e seu objetivo é entregar a maior qualidade de software possível dentro de uma série de pequenos intervalos de tempo fixo, chamados Sprints, que tipicamente duram menos de um mês[35]. Sendo, o objetivo do sprint entregar o máximo valor de negócio possível no menor tempo.

Com base nisso, e em experiências práticas através da disciplina Engenharia de Software cursada na Universidade de Brasília (UnB), fez-se muito conveniente reunir o uso dessas metodologias como orientação para os estudos e desenvolvimento deste trabalho, uma vez que o projeto visa incrementar funcionalidades específicas para um jornal com base em artefatos já existentes e visa o desenvolvimento de um software com uma produção de qualidade e em um tempo limitado.

Partindo-se das metodologias acima citadas, DSR e metodologia ágil com a framework

Scrum, foram definidas algumas etapas. Para a metodologia ágil, adotamos alguns passos e adaptamos algumas estratégias do *Scrum*, ilustrados pela Figura 3.1, que funcionam de maneira incremental e iterativa. Sua forma incremental se dá de forma a incrementar o produto, ou seja, a cada encontro ou reunião com os envolvidos no trabalho algo novo era apresentado de forma que somados, ao final do projeto geraram o aplicativo. E, a forma iterativa acontece repetindo determinadas ações resultando em ações constantes para não se parar o desenvolvimento. Alguns termos utilizados pelo *Scrum* e ilustrados em Figura 3.1 são listados abaixo:

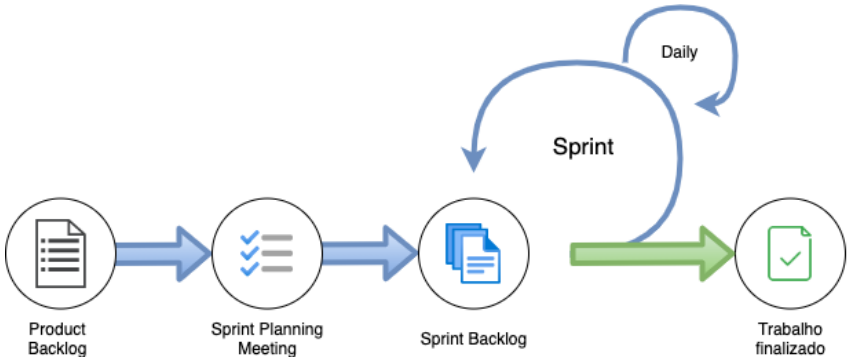


Figura 3.1: Representação visual dos principais artefatos da *Scrum* e seu relacionamento com a *Sprint*. Adaptado de [6].

- *Product Backlog*: é definido por aquilo que é necessário para o resultado final baseado no conhecimento [36];
- *Sprint Planning Meeting*: cada sprint é precedida por uma reunião de planejamento (*Sprint Planning*), onde as tarefas para a *sprint* são identificadas;
- *Sprint Backlog*: lista de itens selecionados na *Sprint Planning* que serão realizados durante a próxima *Sprint*;
- *Sprint*: unidade básica de intervalos de tempo fixo de desenvolvimento;
- *Daily*: reunião de reporte de *status* que ocorre rapidamente e diariamente.

Para a metodologia DSR, a Figura 3.2 descreve cada etapa e suas respectivas saídas. A seguir serão descritas as etapas elaboradas para o desenvolvimento do projeto:

1. Definição do problema;
2. Revisão da literatura;
3. Pesquisa de ferramentas;

4. Pesquisa sobre funcionamento de publicação de notícias em redes sociais;
5. Modelagem da aplicação;
6. Implementação da aplicação.

A primeira etapa apontada, item 1, tem uma abordagem aprofundada no Capítulo 1, onde destaca como os padrões e normas das redações jornalísticas foram afetados com o aumento exponencial das mídias sociais. Nesse sentido, foram feitos estudos e entrevistas, para se entender e discutir os problemas e os ganhos da utilização das mídias sociais, com alunos jornalistas da Universidade de Brasília (UnB). Esses estudos, eram feitos diariamente, mas não reportados diariamente, como propõe a metodologia Scrum, os reportes eram feitos semanalmente em reuniões de fechamento da Sprint. Essa etapa, para a *framework Scrum* foi considerada como trabalho finalizado quando foram coletados insumos necessários para se iniciar pesquisas e implementação do artefato, gerando um *Product Backlog* para próxima etapa.

O item 2 trata da revisao da literatura seguido pelo item 3, que e a pesquisa de ferramentas, e ambos podem ser vistos com mais detalhes no Capítulo 2. Essa etapa nasce do *Product Backlog* gerado pelos estudos feitos no item 1, feito com foco em estudar o relacionamento entre jornalismo e leitores e sua evolucao atraves da internet e das redes sociais, entendendo possiveis dificuldades existentes em gerenciar varias redes sociais que se tornaram foco do relacionamento entre jornal e leitor. Nesse capitulo foram tambem pesquisadas ferramentas de gerenciamento de redes sociais ja existentes para que se pudesse entender os beneficios dispostos por elas e comparar sua utilizacao no cenario de uma redacao jornalistica, entendendo o conveniente de sua utilização no contexto e personalizando pontos validos que pudessem ser usados no jornalismo. Alem disso, para que fosse possivel entender como funciona o processo de recuperacao e integracao de dados das mídias sociais com gerenciadores já existentes, foi necessario conhecer com mais detalhes os tipos de interfaces de programacao disponibilizadas pelas proprias plataformas de mídias sociais que fornecem dados e informações para os gerenciadores. Toda a revisão foi feita baseada no *Product Backlog* proposto nessa etapa, seguindo as reuniões e reportes semanais assim como no item 1, e geraram material e propostas para o *Product Backlog* do próprio MVP proposto.

Para o item 4, uma entrevista foi realizada com uma aluna da Faculdade de Comunicação da Universidade de Brasília e pode ser vista em Apêndice B.

A partir dos itens 1, 2, 3 e 4 descritos, foi possível formar o *Product backlog* do MVP proposto. Sendo o product backlog final uma lista de prioridades que foi constantemente atualizada e aos poucos concluída. Inicia-se a partir desse *Sprint backlog* algumas *sprints* que contemplam o item 5, onde um *design* em forma de protótipo navegável foi feito uma

Sprint e apresentado e aprovado em uma *Sprint Review* (representada na figura pelo fim de cada ciclo de *Sprint*) pelos alunos e professores coordenadores. Ao ser aprovado o que marcou o fechamento de uma Sprint, novas Sprints com tópicos também do *product backlog* se iniciavam com planejamento, através de *Sprint Planning Meetings*, de implementar e desenvolver o MVP, descrita pelo item 6. As funcionalidades do MVP eram propostas separadamentes nas reuniões de planejamento, realizadas e apresentadas ao final da Sprint nas reuniões de fechamento. E se repetia até que completasse todas as funcionalidades propostas no *Product Backlog* de maneira iterativa e incremental, como propõe a metodologia.

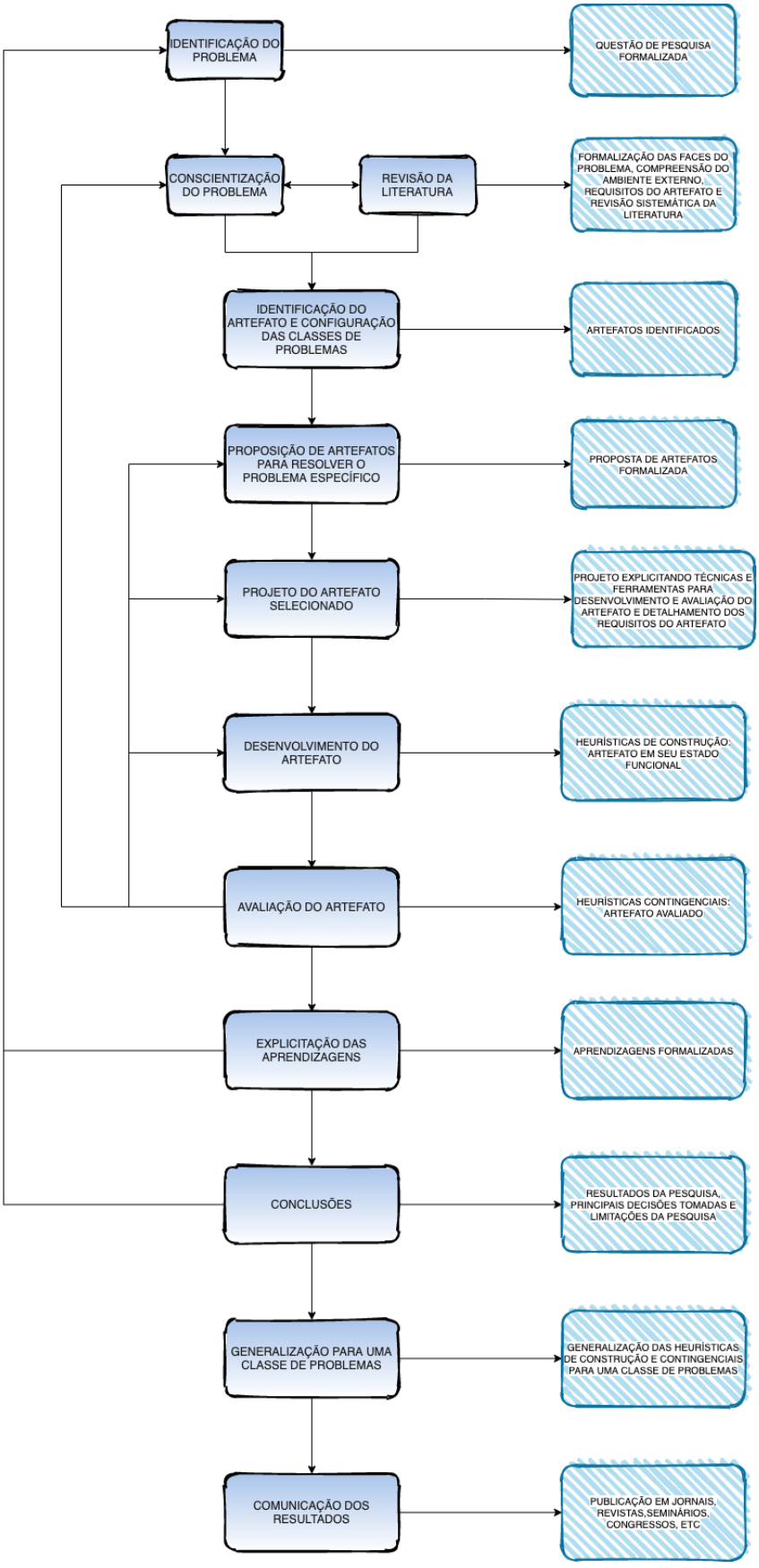


Figura 3.2: Ilustração das etapas da metodologia DSR. Adaptado de [7].

Capítulo 4

Implementação

Neste capítulo é abordada a implementação do artefato proposto nos objetivos gerais deste trabalho, bem como histórias de usuário, casos de uso, recursos, ferramentas, *frameworks* e arquitetura utilizados.

Como abordado anteriormente, esse projeto representa uma possibilidade como prova de conceito para um gerenciador de conteúdos de mídias sociais (Social Media Management System (SMMS)) de um jornal, planejado para agregar à produção de notícias e auxiliar no monitoramento de divulgação dessas notícias, bem como possibilitar o aumento de acesso e integração para com seus leitores.

As possibilidades de divulgação de notícias, de modo simultâneo, a mais de uma plataforma ou rede social de destino, contando com uma funcionalidade de análise e monitoramento de engajamento de usuários permitem apontar a relevância das notícias, ampliando seu alcance, comentando-as e até mesmo apontando outras fontes. Neste sentido, o artefato não tem foco necessariamente em produzir notícias, mas fornecer meios de divulgação de elementos que podem ser noticiados com foco no jornal em mídias sociais. Todas as publicações feitas através da plataforma são armazenadas em base de dados e relacionadas através de anotações que marcam termos importantes da notícia, contribuindo para que posteriormente na construção de novas publicações elas possam ser consultadas usando essas anotações como fonte de busca para postagens associadas à esse termo.

O artefato também conta com um sistema de gerencimento de conteúdo baseado no trabalho [37] com o objetivo de explorar seus recursos de tal modo que incremente a capacidade da redação jornalística e forneça auxílio na busca por notícias e textos já publicados. Esse comportamento parte, assim como [37], de um sistema baseado em anotações para busca e relacionamento semântico entre textos.

Ao final deste Capítulo ficarão mais claras as razões para as decisões de implementação, arquitetura e funcionamento do artefato. O repositório de desenvolvimento do trabalho

pode ser encontrado no github [1].

4.1 Levantamento de Requisitos e Casos de Uso

O SMMS não está ligado diretamente ao leitor, sendo um sistema interno ao processo de divulgação de notícias com foco em jornalistas.

O levantamento de requisitos parte de estudos e observações das necessidades e principais dificuldades que os jornalistas e autores de notícias de um jornal têm em formas de histórias de usuário, que é uma forma de expressar esses requisitos através da metodologia ágil, utilizada para o desenvolvimento deste trabalho e apresentada no Capítulo 3, e que compõe especialmente os itens 1, 3, 4 e 5 descritos pela metodologia Design Science Research (DSR) apresentada, com descrições curtas de funcionalidade contadas a partir da perspectiva de um usuário [38], que aqui é representado pelo jornalista.

[39] sugere a escrita de uma história de usuário como se estivesse na pele do usuário que deseja a história. O uso na perspectiva de primeira pessoa torna a história mais compreensível. Nesse caso, propõe-se identificar o papel que está assumindo dentro da história no seguinte formato: *"Como um tipo de usuário eu quero capacidade ou funcionalidade de modo que o valor do negócio ou benefício"*.

Com base nisso e na entrevista com jornalista apresentada no Apêndice 2, as seguintes histórias de usuário foram escritas para identificar cada funcionalidade necessária no trabalho:

- **Funcionalidade: Login único em plataforma SMMS**
  - **Descrição:** Como jornalista gostaria de logar em uma única plataforma para divulgar e distribuir notícias em mídias sociais da minha escolha sem precisar me autenticar também nas mídias em que desejo publicar as notícias.
- **Funcionalidade: Visualizar a quantidade de leitores nas mídias sociais**
  - **Descrição:** Como jornalista quero acompanhar a quantidade de leitores das notícias nas mídias sociais para analisar qualitiva e quantitativamente as minhas publicações.
- **Funcionalidade: Visualizar quantidade de curtidas, comentários compartilhamentos das publicações**

[1] https://github.com/victoriagoularte/smms

  - **Descrição:** Como jornalista quero acompanhar a quantidade de curtidas e compartilhamentos das notícias na mídias sociais para analisar qualitiva e quantitativamente o engajamento em minhas publicações.

- **Funcionalidade: Visualizar notícia com maior quantidade de curtidas e comentários**

  - **Descrição:** Como jornalista quero acessar a notícia com maior quantidade de curtidas e comentários em determinado período de tempo para entender onde está o maior engajamento.

- **Funcionalidade: Publicar a notícia em uma ou mais plataformas de mídias sociais**

  - **Descrição:** Como jornalista quero preencher um formulário único de publicação de notícia com a possibilidade de anotar termos que considero relevantes e que se relacionam com outras publicações, além de possibilitar selecionar o genêro da notícia, selecionar foto e escolher em quais mídias sociais gostaria de publicar para não precisar fazê-lo para cada mídia social separadamente.

- **Funcionalidade: Transformar os termos importantes anotados em *hashtags* nas mídias sociais**

  - **Descrição:** Como jornalista quero preencher um formulário único de publicação de notícia com a possibilidade de anotar termos que considero relevantes e que se relacionam com outras publicações para que eles se tornem *hashtags* ou *hiperlinks* que nas mídias sociais redirecionam o usuário para uma página com outras publicações relacionadas ao mesmo tema.

- **Agendar divulgação de notícia**

  - **Descrição:** Como jornalista quero selecionar o dia e hora para distribuição da notícia para que ela esteja disponível para o público em horários previamente definidos e que considero adequados à publicação.

- **Visualizar notícias agendadas**

  - **Descrição:** Como jornalista quero visualizar as notícias agendadas para acompanhar o cronograma das minhas postagens.

- **Editar notícia agendada**

Tabela 4.1: Requisitos funcionais.

	Requisito	Descrição
RF01	Cadastrar jornalistas	Cadastro necessário para acesso ao aplicativo gerenciador de mídias socias. O cadastro permitirá que o usuário acesse à análise das postagens nas mídias e à publicação de notícias.
RF02	Publicar notícia	Publicação da notícia em uma ou mais mídias sociais desejadas, dentre elas *Facebook* e *Instagram*, com possibilidade de agendamento para publicação em momento posterior.
RF03	Analisar notícia	Dispor de informações relevantes quanto ao engajamento da notícia publicada nas mídias sociais através do aplicativo, com números métricas e gráficos de comparação.

  - **Descrição:** Como jornalista quero editar notícias agendadas para ajustá-las corretamente ao cronograma, se necessário.

- **Excluir notícia agendada**

  - **Descrição:** Como jornalista quero excluir notícias agendadas para que a publicação não ocorra futuramente, pois poderá haver algum erro na postagem ou ela poderá ter sido invalidada.

Além disso, pesquisas sobre aplicativos com finalidades semelhantes foram realizadas. Essas informações geraram insumos necessários para entender a proposta e desenvolvimento do artefato em forma de um aplicativo *mobile*, que possui também uma ferramenta auxiliar de análise de dados *web*, e podem ser representadas por requisitos funcionais e não funcionais. Os requisitos listados em Tabela 4.1 e Tabela 4.2 foram organizados em casos de uso e representados através do diagrama Figura 4.2.

Nos casos de uso apresentados, a *persona* do autor ou jornalista tem a função de criar, revisar e publicar notícias em mídias sociais através do sistema que é uma aplicação *mobile*. Essa função acarreta tarefas de anotar publicações que marcam determinado assunto ou tema, classificar a categoria da publicação, anexar uma imagem que ilustra a notícia, agendar a publicação, se necessário, para momento futuro, e apontar em que mídia social essa notícia deve ser divulgada. A informação produzida a partir daí pode ser consumida pelos *leitores* ou seguidores das notícias nas mídias sociais às quais a notícia foi então publicada e eles podem facilmente acessá-las através do sistema de busca por *hiperlinks* ou *hashtags* existentes também em cada mídia social atualmente. Depois de publicada a notícia, seus dados publicados são persistidos em uma segunda plataforma, representada pelo *Sistema de Gerenciamento de Dados*, uma ferramenta de gerenciamento

Tabela 4.2: Requisitos não funcionais.

	Requisito	**Descrição**
RNF01	Classificar categoria da notícia	Classificação de categoria para criar relacionamento entre notícias como editorias para melhor monitoramento e busca no processo de análise.
RNF02	Anotar a notícia	Marcação da notícia com termo relevate sobre o assunto através do sistema de *hashtags* para relacionar publicações feitas e faciliar a busca por notícias com mesmo termo anotado.
RNF03	Anexar ilustração à notícia	Possibilitar agregar à notícia uma imagem que ilustra o tema tratado.
RNF04	Agendar publicação de notícia	Possibilitar o agendamento de uma publicação para momento futuro.
RNF05	Consulta de notícia	Fornecer mecanismos de consulta à notícias que já foram publicadas através de campos de busca.

e de análise onde também são administrados os autores ou jornalistas cadastrados para utilizar o aplicativo gerenciador de mídias sociais. Nessa ferramenta, o jornalista com permissões para acessá-la é capaz de alterar, cadastrar e excluir novos usuários, além de realizar consultas e análises relacionadas ao aplicativo e às publicações feitas através do aplicativo.

Esses dados persistidos nessa ferramenta são consumidos novamente pelo mesmo sistema de publicação, ou aplicativo, onde o autor ou jornalista podem publicar notícias previamente agendadas, acompanhar e analisar o alcance da publicação, pesquisar por notícias relacionadas à determinado assunto, anotação ou categoria já publicados.

Baseando-se nessas explicações, casos de uso e histórias de usuário, cumprindo o item 5 descrito na metodologia como "Modelagem da aplicação", foi utilizada uma ferramenta, chamada MarvelApp [40], para modelar o então aplicativo *mobile* descrito e proposto, modelagem essa que foi avaliada em uma reunião de *Sprint* por jornalista e aprovada. O protótipo navegável pode ser acessado no MarvelApp [2]. O fluxo simples da modelagem é representado através de um fluxograma em Figura 4.3. No fluxograma, as telas enumeradas são descritas abaixo:

- Tela 1: Tela de entrada do aplicativo, onde usuário pode se cadastrar para utilizar o aplicativo e efetuar o *login* na plataforma;

[2]https://marvelapp.com/prototype/9a31i66/screen/61152645

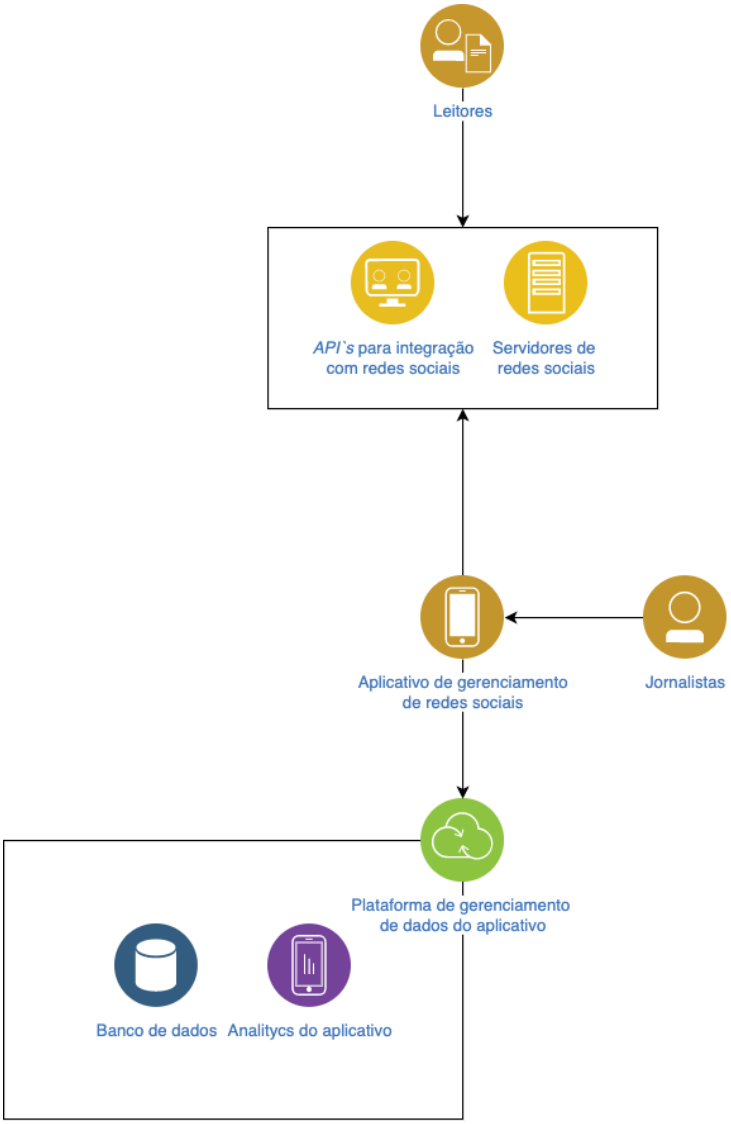


Figura 4.1: Diagrama com a arquitetura geral da aplicação. (Fonte: Autora).

- Tela 2: Tela de permissão e acesso ao *Facebook* e *Instagram*, de onde o aplicativo SMMS consumirá dados e informações;
- Tela 3: Tela principal do aplicativo. Apresenta duas abas de acesso. Na primeira aba, à qual o usuário é inicialmente posicionado contém informações e monitoramento para as mídias sociais ministradas. A partir dessa tela, pode-se acessar as telas 4, 5 e 8.
- Tela 4: Tela onde são mostrados gráficos que representam picos e quedas de engajamento relacionados à curtidas e comentários de publicações com filtros de tempo diários, mensais e anuais.

Figura 4.2: Diagrama de casos de uso (Fonte: Autora).

- Tela 5: Tela acessada pela segunda aba da tela principal. Tela que realiza a publicação de notícias em mídias sociais selecionadas. Nessa tela é apresentado um formulário ao usuário com capacidade de incluir aquivos multimídia e termos importantes, ou hashtags. Aqui o usuário também pode selecionar se deseja agendar a publicação para momento futuro e acessar também as publicações agendadas.
- Tela 6: Tela de agenda de publicações. Aqui são listadas todas as publicações pendentes e agendadas com todas as suas informações de publicação.
- Tela 7: Tela de agendamento de publicação. Nessa tela o usuário é capaz de programar o dia e hora para publicação da notícia.
- Tela 8: Tela de consulta de notícias já publicadas. Nessa tela o usuário pode buscar

por notícias através de título, *hashtags* e outros filtros. As notícias recuperadas são apresentadas em forma de lista com todas as suas informações de publicação.

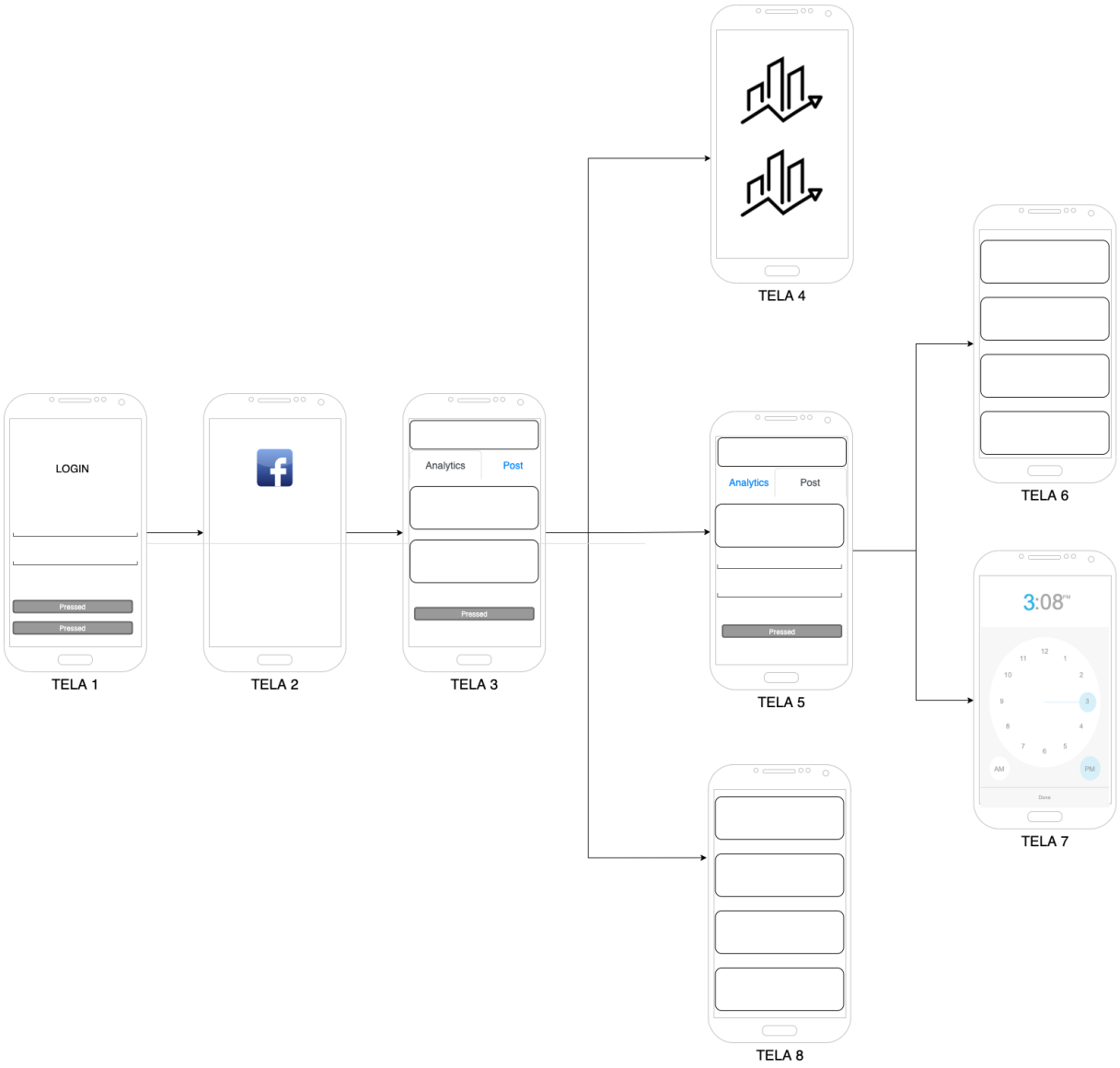


Figura 4.3: Fluxo de navegação de aplicativo *mobile* como MVP(Fonte: Autora).

Em suma, o principal foco de desenvolvimento aqui descrito é a aplicação *mobile* para SMMS, que conta com uma ferramenta *web* auxiliar que persiste e armazena todos os dados de publicações realizadas através do gerenciador e possibilita a análise e gerenciamento de conteúdo que tem como base e inspiração o módulo Content Management System (CMS) de [37] e abrange uma parte analítica de conhecimento no próprio aplicativo que também descreve pontos semelhantes ao ()KMS de [1]. Portanto esses módulos são, de certa forma, implementados no projeto para alcançar o objetivo proposto.

4.2 Arquitetura

No *framework* SMMS proposto, as funcionalidades de gerenciamento se comunicam com os módulos que gerenciam os conteúdos, baseados em Content Management System (CMS) e Knowledge Management System (KMS). Os dados capturados e processados em SMMS são persistidos em nuvem e podem ser modificados em CMS através de uma ferramenta *web*, o *Firebase* que será melhor explicado posteriormente, e KMS através de uma interface de usuário na própria aplicação onde pode-se inferir conhecimento e análises feitas sobre o conteúdos publicados.

A comunicação feita entre os módulos, em todos os sentidos, é através do padrão Representational State Transfer (REST), uma abstração da World Wide Web Consortium (W3C) como protocolo, utilizando a biblioteca *Retrofit* [41] como um *REST Client* no aplicativo. Essa troca de informações pode ser representada pela Figura 4.4.

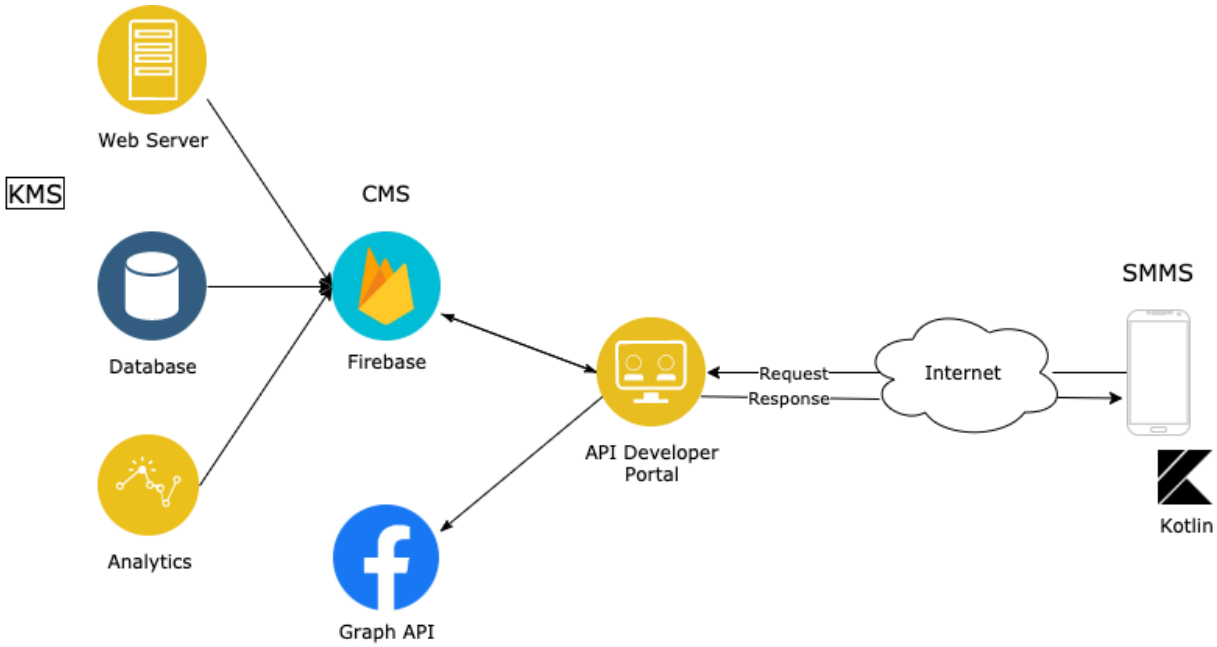


Figura 4.4: Comunicação entre módulos.

Para identificação dos dados transitados entre os módulos é utilizado o formato JavaScript Object Notation (JSON) [42], e os módulos CMS e KMS funcionam como um API REST enquanto o módulo SMMS funciona como um *REST Client*. A biblioteca brevemente citada, a *Retrofit* [41], é utilizada para suporte ao desenvolvimento de modo a abstrair e facilitar requisições transformando a Application Programming Interface (API) de requisições HTTP em uma interface Java.

As requisições HTTP de acesso direto às mídias sociais da empresa *Facebook*, que são as mídias *Facebook* e *Instagram*, são feitas através da *Graph API* [23], sendo a principal

forma de inserir e retirar dados da plataforma do Facebook, disponibilizada para que outros aplicativos possam consultar dados programaticamente, postar novas histórias, gerenciar anúncios, fazer upload de fotos e realizar outras tarefas. Por outro lado, tem-se também requisições HTTP à ferramenta *Firebase*, que atua como a interface dos módulos CMS e KMS pois é utilizado como banco de dados na nuvem NoSQL para persistir informações de publicaçoes feitas no módulo SMMS e forncecer análise e conhecimento do conteúdo persistido e do acesso ao aplicativo, sendo possível acessá-lo diretamento do dispositivo móvel ou de um navegador Web, sem um servidor de aplicativos.

O acesso aos dados do SMMS são feitas através dessas interfaces citadas que representam CMS e KMS. Desta forma, a autenticação no módulo e manipulação direta dos dados é encarregada apenas àqueles que têm cadastro realizado no aplicativo gerenciador das mídias sociais. As funcionalidades para esses usuários são as descritas e ilustradas em Figura 4.2.

4.2.1 Arquitetura do Aplicativo

O aplicativo foi desenvolvido com base no padrão de projeto Model-View-ViewModel (MVVM), seguindo a separação de princípios (Separation of Concerns (SoC)) [43], um princípio de design para separar um programa de computador em seções distintas, de modo que cada seção trate de uma preocupação separada. O MVVM possibilita a divisão do projeto em camadas muito bem definidas e o principal objetivo é separar as lógicas e regras de negócio da lógica da interface do usuário. Seu conceito e aplicação são muito bem abordados em [8]. E, mais a fundo, com a finalidade de garantir melhor independência entre as camadas, na *ViewModel* que representa a camada com regras de apresentação e regras de negócio, foi acrescentada uma subcamada exclusivamente para aplicação das regras de negócio do aplicativo, que chamamos de *Interactor*, mantendo a *ViewModel* em sua estrutua original apenas as regras de apresentação [44]. Transformando assim o padrão em Model-View-ViewModel-Interactor (MVVMi).

Abaixo uma breve explicação sobre cada camada apontada em Figura 4.5:

- **Model**
  - contém a conexão com banco de dados, representado na imagem de [44] em Figura 4.5 pela *framework Room*, no entanto usamos no projeto apenas o objeto *SharedPreferences* do Android que é um arquivo que contém pares de chave-valor e fornece métodos simples para leitura e gravação de dados no dispositivo;
  - tem a lógica necessária para processar os dados no banco de dados, no Shared-Preferences ou outra fonte, como dados de origem remota ou *webservices*;

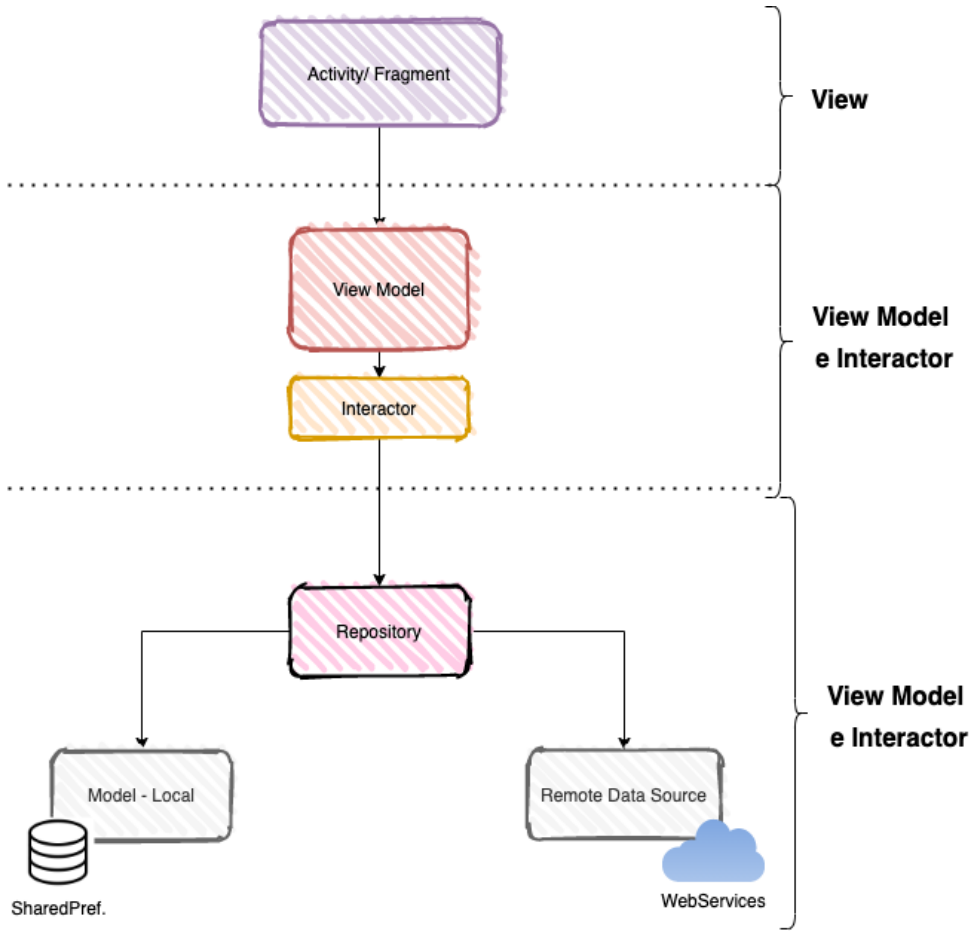


Figura 4.5: Padrão de projeto MVVMi. Adaptado de [8].

  - processa os dados obtidos na fonte e os mapeia na forma necessária como objetos de negócios, em classes que chamamos de "repositórios", para que as camadas ViewModel e Interactor possam utilizar adequadamente.

- **Interactor**
  - camada que acessa objetos de negócios da Model, valida e repassa à ViewModel;
  - funciona como um apêndice auxiliar para ViewModel, fazendo as verificações das regras de negócio e validações de lógica de modo a deixar a ViewModel apenas com responsabilidade de validações de lógica de apresentação.

- **ViewModel**
  - contém as propriedades que definem o comportamento da View a partir dos dados que essa camada recebe da View ou através de retornos da Interactor;

- fornece informações da View à Interactor para validações de dados e verificações de regras de negócio. Assim que são feitas essas validação pela Interactor, recebe um retorno e avalia um comportamento que será enviado para a View;
- utiliza, nesse projeto, o *databinding* para notificar mudanças aos observadores na View. Sendo essa uma técnica que liga fontes de dados entre um provedor de conteúdo e seu consumidor e mantendo os valores sincronizados. Ou seja, os dados da ViewModel enviados e processados na Interactor e Model, podendo ser considerados os provedores de conteúdos, são acessados e observados pela View, que consome essse conteúdo, para que ações e estados na interface sejam reproduzidos a partir dos valores obtidos.

- **View**
  - responsável por definir a estrutura, layout e aparência do que será exibido na tela e formatação da interface com usuário, conhecidas especificamente na plataforma Android como *Atividade* ou *Fragmento*;
  - processa os dados obtidos na UI para serem disponibilizados de forma adequada para manuseio das ViewModels.

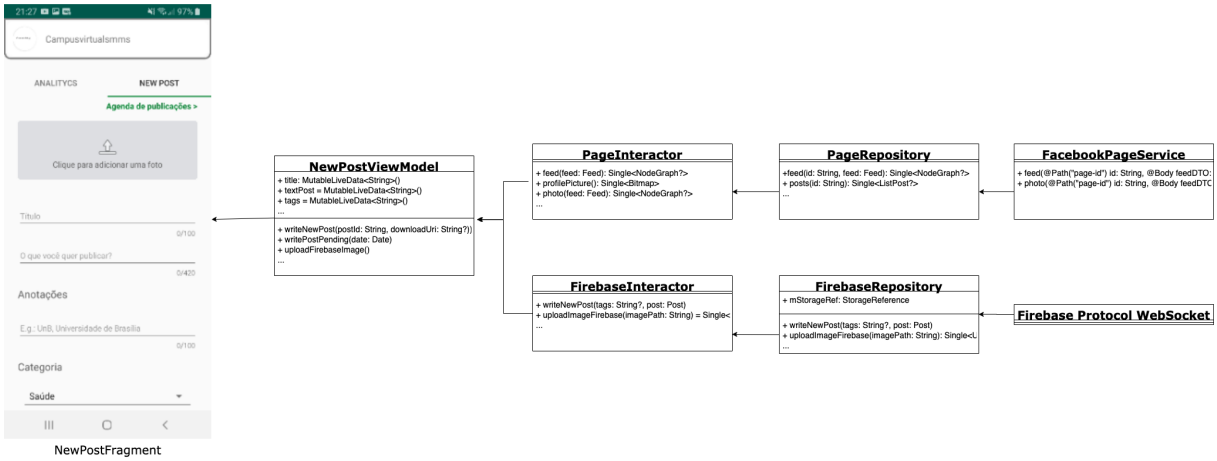


Figura 4.6: Representação das camadas da arquitetura no projeto.

4.3 Desenvolvimento do artefato

A elaboração do artefato seguiu os seguintes passos:

1. Módulos e Bibliotecas;

2. Arquitetura da persistência de dados;
3. Modelagem dos processos de publicação da notícia;
4. Modelagem dos mecanismos de análise das notícias;
5. Publicação automática da notícia;
6. Mecanismo de busca da notícia;

4.3.1 Módulos e Bibliotecas

Uma visão geral dos principais módulos e bibliotecas utilizados para o desenvolvimento do artefato e para melhor entedimento da arquitetura aplicada.

A Tabela 4.3 lista as mais importantes bibliotecas necerrárias importadas no aplicativo. São pacotes específicos para aplicações *mobile* Android.

4.3.2 Arquitetura da persistência de dados

Para a modelagem dos dados neste trabalho os focos foram o cadastro dos usuários na aplicação, e a armazenagem dos dados de uma pubicação.

Analisando esse cenário, alguns pontos foram levados em consideração, como uma manipulação e estruturação descomplicada e flexível, uma vez que entidades, atributos e objetos de notícias podem surgir e ter a necessidade de registro a qualquer momento. Também foram levadas em consideração a velocidade para se realizar consultas e atualizações dos dados da notícia, a performance com volume de dados grandes e a disponibilidade dos dados em tempo real. Partindo-se disso, optamos por utilizar o banco de dados da ferramenta *Firebase* [45], um Banco de Dados não relacional, ou seja, não utiliza SQL e relacionamentos entre entidades, no caso do Firebase o sistema utilizado é o JavaScript Object Notation (JSON).

O formato de banco de dados no *Firebase* é o NoSQL [46], um formato de banco de dados que não possui como padrão o sistema de tabelas e relacionamentos entre dados, basicamente tratando cada informação como um nó de um tronco que pode ser considerada a raiz do banco de dados. Abaixo uma ilustração que representa uma árvore, como o SMMS como seu tronco principal e as raízes como seus nós, onde cada raiz pode ter outras sub-raízes. Cada nó está identificado a seguir.

raiz de agregação tenha diversas propriedades de coleção filhas

smms raiz do banco de dados com agregação de sub-raízes que são considerados os objetos ou entidades;

posts nó subraiz de SMMS e raiz de uma coleção de folhas com as propriedades:

Tabela 4.3: Bibliotecas utilizadas.

Biblioteca	**Descrição**
Architecture Componentes	conjunto de bibliotecas que auxilia na construção de *apps* robustos, testáveis e de fácil manutenção. Foco em gerenciamento do ciclo de vida dos componentes de UI, evita vazamento de memória, responsável pela existência das ViewModels, dentre outros.
Material Design	Uma biblioteca abrangente para design visual, oferecendo recursos que ajudam no desenvolvimento de estilos, temas, animações, etc.
RX Libraries	conjunto de bibliotecas que permitem representar de forma declarativa, qualquer operação como um fluxo assíncrono de dados, que pode ser criado por qualquer *thread*, e consumido por múltiplos objetos em (opcionalmente) *threads* diferentes.
MPAndroidChart Library	Biblioteca que auxilia na construção de gráficos para compor a parte visual de análise e engajamento nas mídias sociais
Retrofit Library	é uma populares bibliotecas de *HTTP Client* para Android e Java. Um dos princípios do Retrofit é a simplicidade, que permite que não nos preocupemos com toda a complexidade de criar uma conexão *Web Service*, uma vez que grande parte de sua correspondente lógica é abstraída. Com essa proposta, se é possível simplesmente implementar algumas interfaces, e anotar alguns métodos.
Map Struct Mapper Library	é usada para mapear entre os tipos de objetos remotos fornecidos através dos *Web Services* para objetos de domínio que serão adequados para utilização nas camadas de lógica e visualização do aplicativo. A biblioteca basicamente reduz replicação de código para *parse* de objetos.
Picasso	permite o carregamento de imagens sem complicações no aplicativo via URL, imagens essas que compõem publicações feitas através do aplicativo SMMS.
Firebase Google Libraries	bibliotecas para vincular o Firebase ao *framework* desenvolvido. Possuem métodos e funções de accesso ao banco de dados da nuvem, onde as informações do aplicativo são persistidas, possui o meio de autenticação e cadastro feitos pelos usuários do aplicativo.
Facebook SDK	disponibiliza componentes visuais padrões com *links* de integração e acesso às mídias sociais. Contém a Graph API, uma API baseada em HTTP que usamos no aplicativo para consultar dados de forma programática, publicar conteúdos, gerenciar anúncios, carregar fotos e realizar outras tarefas.
Hilt	uma biblioteca de injeção de dependência para Android que reduz a injeção manual de código de texto clichê no projeto. Injeção de Dependências é um tipo de Inversão de Controle e significa que uma classe não mais é responsável por criar ou buscar os objetos dos quais depende. Isso serve para desacoplar as classes, evitando dependência direta entre elas.

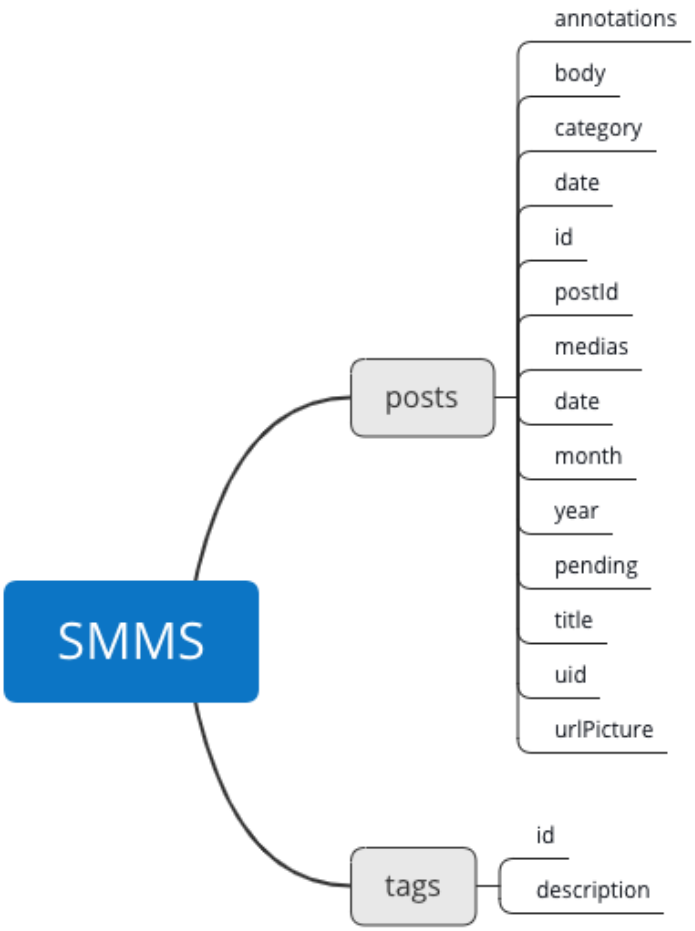


Figura 4.7: Representação do banco de dados NoSQL.

- **annotations:** coleção de tags, ou marcadores, que representam as *hashtags* nas mídias sociais. É um termo associado a um tópico da notícia para associar a publicação a outras notícias e para facilitar a busca por*posts* marcados com a mesma anotação;
- **body:** é o corpo da notícia, é o texto que irá compor a legenda da pubicação na rede social;
- **category:** é a categoria à que a notícia se refere, poderiam ser consideradas aqui as editorias do jornal para classificar a categoria da notícia, por exemplo;
- **date, month, year:** são dia, mês e ano em qua o *post* foi publicado. Se for uma data futura trata-se de um post com publicação pendente agendada no aplicativo para a data em questão.

- **id:** identificador único do *post* no firebase. Cada publicação tem o seu próprio identificador.
- **postId:** identificador do *post* na base de dados das mídias sociais *Facebook* e *Instagram*;
- **medias:** as mídias sociais às quais o *post* foi publicado, podendo elas ser, em primeiro momento no *feed:* do *Facebook* ou *Instagram*, ou nas histórias do *Instagram*.
- **pending**: indicador se o *post* está pendente de publicação, ou seja, se é um *post* agendado
- **title:** um título para a publicação
- **uid:** identificador do usuário que realizou a publicação
- **urlPicture:** *link* da imagem associada ao *post*

tags nó subraiz de SMMS e raiz de uma coleção de folhas com as propriedades dos marcadores de termos importantes de um*post* que também chamamos de *annotations* anteriormente. Possui as propriedades:

- **id:** identificador único das *tags*, ou anotações, feita nos *posts*. Cada *tag* é única mas pode ser usada mais de uma vez em vários *posts*. Classificando uma relação de uma para vários, ou *many-to-one*.
- **description**: é a própria descrição do termo

4.3.3 Modelagem dos processos de publicação da notícia

Esta seção consta com a modelagem do processo de publicação da notícia na *framework* SMMS. A publicação se dá com a integração entre duas plataformas, o *Firebase* e a *GraphAPI*, anteriormente citados. Através dessa integração é possível fazermos uma publicação coerente e adequada a cada mídia social de destino, podendo ser o *Facebook* ou *Instagram*, que selecionamos chamando microserviços[3] de postagens da *GraphAPI*, e é possível que administremos as informações persistidas no *Firebase*, salvando aquelas que são necessárias para consultas futuras através de métodos e funções abstraídas na biblioteca do *Firebase* com acesso aos serviços da plataforma.

É importante ressaltar aqui que em todas as requisições feitas através da GraphAPI é exigido um *token* de sessão de login na rede social, ou seja, o usuário da rede social

[3]Microsserviços são uma abordagem arquitetônica e organizacional do desenvolvimento de software na qual o software consiste em pequenos serviços independentes que se comunicam usando APIs bem definidas.

precisa estar logado para que os microserviços tenham permissão para acessar as informações corretamente. Desse modo, o que o aplicativo do framework SMMS faz, assim que o jornalista se autentica na framework, é solicitar autorização de acesso às informações do usuário em suas mídias sociais. Assim que concedida a autorização, um token é devolvido pela GraphAPI ao aplicativo SMMS que salva essa chave localmente em *cache* no celular utilizando o arquivo *shared preferences* do Android para que então toda requisição posterior possa usá-lo. Caso esse token salvo expire depois de determinado tempo, é solicitado ao usuário que autorize o acesso novamente.

Para auxiliar e entender como funcionavam as requisições através da GraphAPI, que tem uma documentação muito extensa e complexa, utilizamos a ferramenta disponibilizada para desenvolvedores pela própria Facebook, chamada de Explorador da GraphAPI [47]. Com essa ferramenta foi possível e menos custoso o mapeamento das rotas utilizadas e dos objetos de retorno recebidos na framework SMMS.

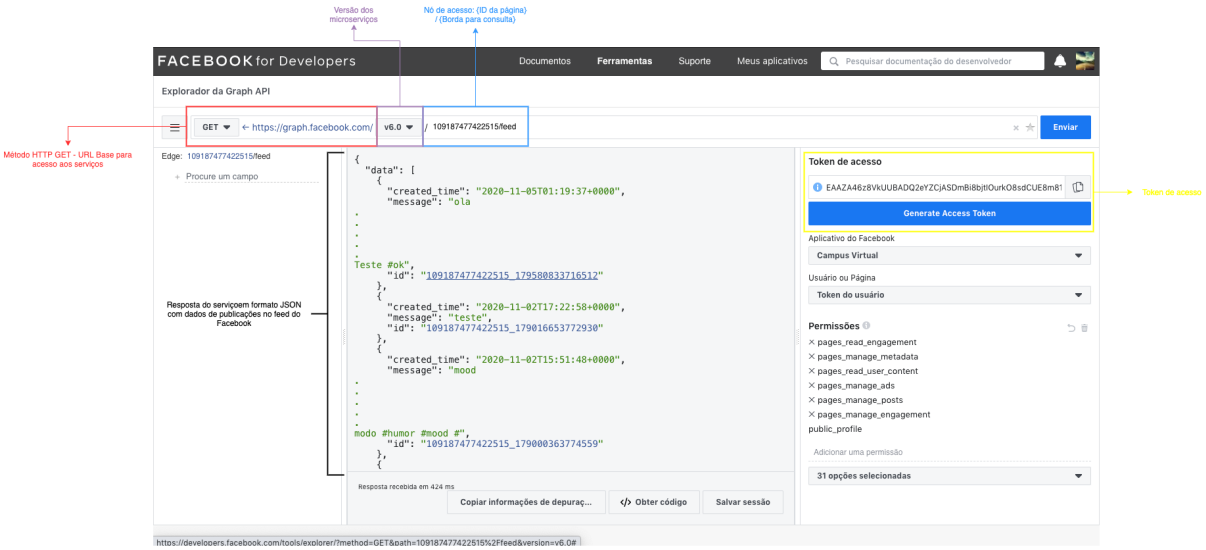


Figura 4.8: Explorador da GraphAPI.

4.3.4 Modelagem dos mecanismos de análise das notícias

Assim como na modelagem dos processos de publicação da notícia, a modelagem dos mecanismos de análise, que chamamos de *Analytics* no aplicativo, é feita partindo da integração com o Firebase e GraphAPI com um token de sessão válido. A maioria de suas requisições são GET, para resgatar informações, remanejá-las e compilá-las de modo que o jornalista possa analisar o engajamento e monitorar situações relacionadas à conta nas mídias sociais.

Para informações referentes ao usuário nas mídias sociais, como quantidade de seguidores, foto do feed, nome da página do jornal na rede social, dentre outros, utilizamos como principal parâmetro o identificador do usuário ou da página na plataforma Facebook. Enquanto que para informações relacionadas às publicações em si como quantidade de curtidas, quantidade de compartilhamentos, quantidade de publicações, dentre outros, utilizamos como principal parâmetro os identificadores dos *posts* na rede social.

4.3.5 Publicação automática da notícia

Com a conclusão da etapa anterior que nos disponibiliza requisições POST para publicar a notícia informando parâmetros como legenda, URL de uma imagem, *hashtags*, etc, em cada rede social específica, nos fez capaz de trabalhar com variações dessas requisições de modo a permitir a seleção de um ou mais destinos de rede social, sendo do formato: publicação no feed do Facebook, publicação no feed do Instagram ou publicação nos *stories* do Instagram. Desse modo, sendo de total responsabilidade da framework SMMS o momento de solicitar a publicação através do microsserviço, o torna capaz de solicitar a publicação para esses três formatos citados simultaneamente de forma automática sem que o usuário se preocupe em fazer algo a mais nas plataformas de destino na notícia selecionadas. Além disso, com o poder de publicar a notícia através dessas requisições quando solicitado, o aplicativo é capaz de agendar uma notícia para momento futuro, caso o usuário deseje, reservando as informações do *post* no Firebase marcando-as como publicação pendente e notificando automaticamente através de notificações do aplicativo no momento em que o usuário gostaria que fosse publicado. Nesse momento, o usuário confirma a publicação do *post* pendente e o aplicativo publica a notícia requisitando os serviços POST da Graph API para as mídias sociais previamente selecionadas.

4.3.6 Mecanismo de busca da notícia

Trata-se de uma funcionalidade presente na parte de *Analytics* do aplicativo. Partindo do princípio de que todas as publicações feitas através do aplicativo da framework foram persistidas no banco de dados do Firebase através da intgração, então a busca é feita também através dessa integração utilizando métodos de consulta da plataforma. A busca é feita com os parâmetos que são preenchidos pelo usuário: título, anotação (*tag* ou *hashtag*) e trecho da notícia, sendo todos eles campos opicionais, ou seja, se quiser buscar uma notícia apenas por uma anotação, a consulta na base de dados será feita buscando a coleção de notícias que possuem essa anotação solicitada. Se mais de um parâmetro é preenchido as buscas são feitas separadamente, cada campo realiza uma busca e ao final é feita uma intersecção entre as coleções resultantes de cada busca trazendo apenas os elementos

Tabela 4.4: Coleção de publicações recuperadas pela busca pelo título: "UnB".

Coleção de publicações recuperadas pela busca pelo título: "UnB"
annotations : ["description": "unb", "description":"universidade"]
body : "A Universidade de Brasília se destaca..."
category : "Social"
date : "04/11/2020"
id : "1abcd"
media : ["facebook"]
pending : false
title : UnB premiada em x
uid : "aGvWc9WhG"

Tabela 4.5: Coleção de publicações recuperadas pela busca pela anotação, ou *hashtag*: "universidade".

Coleção de publicações recuperadas pela busca pela anotação, ou *hashtag*: "universidade"	
annotations : ["description": "saude", "description":"universidade"]	**annotations** : ["description":"universidade"]
body : "Cuidados com saúde mental em tempo de pandemia..."	**body** : "A Universidade de Brasília se destaca..."
category : "Saúde"	**category** : "Social"
date : "05/10/2020"	**date** : "04/11/2020"
id : "fheurh2"	**id** : "1abcd"
media : ["facebook"]	**media** : ["facebook"]
pending : false	**pending** : false
title : Saúde Mental e Pandemia	**title** : UnB premiada em x
uid : "aGvWc9WhG"	**uid** : "aGvWc9WhG"

comuns presentes em todas coleções recuperadas. O algoritmo, em pesudocódigo, que representa a Busca por Notícias pode ser visto em Apêndice A.

Para melhor compreensão, imagine que uma pesquisa foi feita preenchendo os campos título com "UnB", anotação com "universidade"e trecho da notícia com "Brasília". Em primeiro momento, as seguintes coleções de notícias, representadas pelas tabelas Tabela 4.4, Tabela 4.5, Tabela 4.6 mostram os objetos e suas devidas propriedades que foram recuperadas no Firebase através dos serviços de busca para cada item separadamente.

Feito isso, em segundo momento temos a lógica de intersecção aplicada às listas recuperadas e, nesse cenário, temos uma única notícia comum à todas as coleções, que está identificada pelo id "1abcd". Portanto, o resultado final da busca pode ser representado por Figura 4.7

Tabela 4.6: Coleção de publicações recuperadas pela busca pelo trecho "Brasília".

Coleção de publicações recuperadas pela busca pela anotação, ou *hashtag*: "universidade"	
annotations : ["description": "ciencia", "description":"saude"] **body** : "Pesquisa realizada em Brasília..." **category** : "Saúde" **date** : "20/11/2020" **id** : "23tn33" **media** : ["facebook", "instagram"] **pending** : true **title** : Pesquisa **uid** : "aGvWc9WhG"	**annotations** : ["description":"universidade"] **body** : "A Universidade de Brasília se destaca..." **category** : "Social" **date** : "04/11/2020" **id** : "1abcd" **media** : ["facebook"] **pending** : false **title** : UnB premiada em x **uid** : "aGvWc9WhG"

Tabela 4.7: Resultado da busca.

Coleção de publicações recuperadas pela busca pelo título: "UnB", anotação: "universidade"e trecho "Brasília"
annotations : ["description": "unb", "description":"universidade"] **body** : "A Universidade de Brasília se destaca..." **category** : "Social" **date** : "04/11/2020" **id** : "1abcd" **media** : ["facebook"] **pending** : false **title** : UnB premiada em x **uid** : "aGvWc9WhG"

4.4 Funcionamento

À princípio, antes de iniciar o desenvolvimento do aplicativo, um protótipo navegável foi feito com o intuito de criar uma interface agradável, com uma boa experiência para os usuários, para que fosse o mais fácil, simples, intuitivo e eficiente para o propósito. Além de fornecer mais agilidade do momento de desenvolver de fato. O protótipo navegável foi feito em uma ferramenta de design chamado Marvel App [40]. O link para o protótipo navegável descrito pode ser acessado nesse link [4]. Uma documentação de como configurar e um passo-a-passo para a utilização do artefato desenvolvido pode ser encontrada no link [5]. O artefato é composto pelo então aplicativo que completa o tetraedro baseado em [1], e tem funcionalidades como *analytics* para monitoramento das notícias publicadas nas mídias sociais, publicação de notícias nas mídias socias, busca pela notícia e agendamento da notícia. Um cadastro também deve ser feito para que tenha acesso ao aplicativo. O usuário cadastrado será persistido e pode ser gerenciado no *Firebase* como mostra Figura 4.11.

Um pré-requisito para que o aplicativo funcione corretamente é que o usuário tenha acesso à conta no Facebook da página do jornal onde as notícias serão publicadas. Depois disso, o usuário deve se cadastrar na plataforma como mostra a Figura 4.9 e Figura 4.10 preenchendo um e-mail e senha e clicando no botão "Cadastrar". Com um retorno de sucesso, pode prosseguir na mesma página clicando no botão "Entrar"que irá redirecionar, no primeiro acesso, para a tela de permissões do Facebook onde será resgatado o token de sessão para chamadas posteriores que se comunicam com a GraphAPI do Facebook.

Feito isso, o usuário se depara, primeiramente, com a tela de análise das notícias no aplicativo, parte que chamamos de *Analytics*. Nessa tela, algumas chamadas à GraphAPI do Facebook são feitas de modo assíncrono, para que cada campo possa ser mostrado à medida que seu valor for recuperado sem bloquear todos os campos e carregá-los todos de uma vez.

A tela está ilustrada pela Figura 4.12.Para essa imagem, em modo de depuração do aplicativo, interceptamos as chamadas HTTP e alteramos os valores para que ficassem mais próximos da realidade, uma vez que o usuário usado para testes da aplicação tem poucos usuários seguidores (leitores) e interações.

As chamadas feitas à GraphAPI nessa tela foram:

accounts: informa o ID do usuário do *Facebook* e recebe como retorno o ID da sua página de publicações e o nome da página de publicações.

```
@GET("{id}/accounts")
```

[4] https://marvelapp.com/prototype/9a31i66/screen/61152645
[5] https://github.com/victoriagoularte/smms

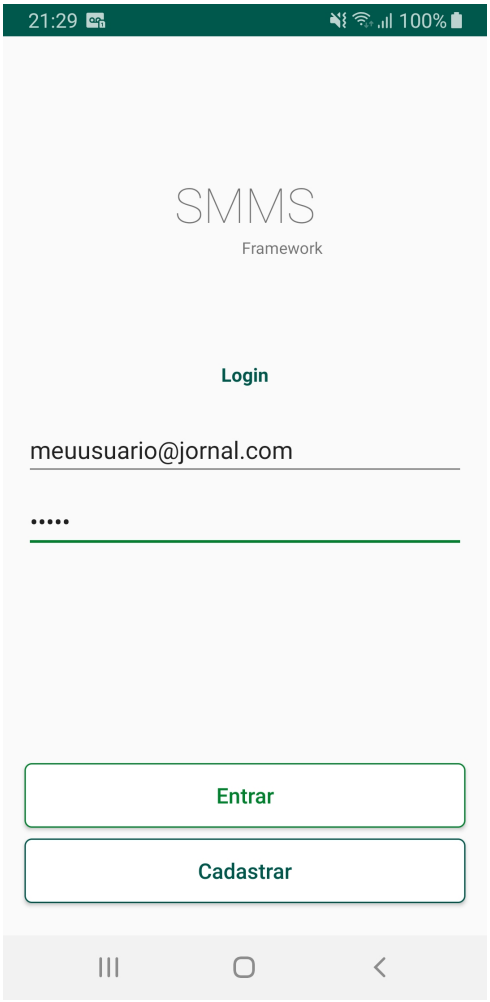


Figura 4.9: Cadastro e Login no *app* com usuário fictício.

picture: informa o ID da página recebida em **accounts**, o tamanho desejado e recebe como retorno a imagem usada no perfil da página.

```
@GET("{page-id}/picture?height=200&width=200&type=normal")
```

friends: informa o ID do usuário e recebe como retorno o número de seguidores no Facebook.

```
@GET("{id}/friends")
```

page likes: informa ID da página recebida e recebe como retorno o número de usuários que curtiram a página no Facebook

```
@GET("{page-id}/picture?height=200&width=200&type=normal")
```

Figura 4.10: Concessão de permissões ao *Facebook*.

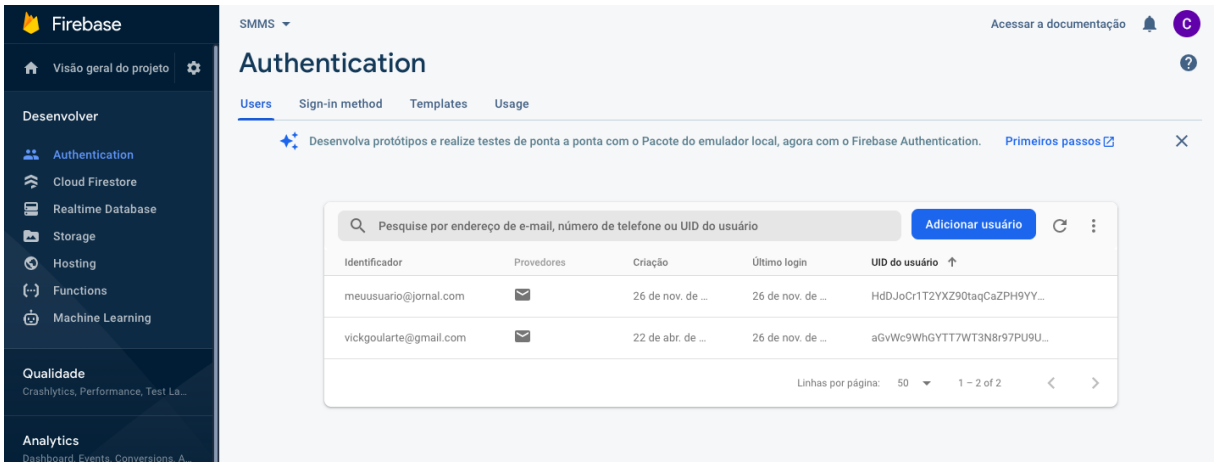


Figura 4.11: Usuário persistido no *Firebase*.

Figura 4.12: Tela de análise das notícias no aplicativo.

page likes: informa ID da página recebida e recebe como retorno o número de usuários que curtiram a página no Facebook

```
@GET("{page-id}/picture?height=200&width=200&type=normal")
```

Instagram Account: recupera informações da conta do Instagram, como o ID do usuário no Instagram e quantidade de seguidores, através do ID da página que é vinculada.

```
@GET("{page-id}?fields=instagram_business_account")
```

posts: recupera *posts* realizados na página informando o ID da página.

```
@GET("{page-id}/posts")
```

insights: recupera o número de engajamentos da página, sendo eles definidos por comentários, curtidas ou impressões. Para que isso seja feito é necessário informar a métrica, ou tipo de engajamento, desejado e o ID do post recuperado em **posts**

```
@GET("{page-post-id}/{metric}?summary=total_count")
```

É importante ressaltar que para todos os casos citados acima houve desenvolvimento e trabalho em lógicas e regras negociais com algoritmos de filtragem de dados para determinados períodos de tempo, podendo comparar a quantidade de interações realizadas e o engajamento naquele período, além de o mapaeamento de cada retorno recebido e o controle de armazentamento desses dados para chamadas consequentes que necessitam de informações recuperadas como parâmetro.

Em Figura 4.12, pode-se observar um *link* descrito como "Mais informações". Ao clicar nesse link, o usuário é redirecionado para uma tela de gráfico de engajamentos onde tem uma visão mais detalhada sobre a quantidade de posts realizados pela quantidade de interações em cada post naquele período filtrado, veja em 4.13, um exemplo:

Ainda na tela de *Analytics* observa-se um botão de "Pesquisar". Ao selecionar esse botão o usuário é redirecionado para a tela de busca por notícias onde pode escolher o tipo de busca desejada, sendo por título, anotação (ou *hashtag*) ou trecho da publicação, ilustrado em 4.14(a). Os resultados da busca possuem informações de data de publicação, data de agendamento para posts que ainda não foram publicados, plataforma em que foi realizada a publicação (Facebook ou Instagram), título, imagem, conteúdo, dentre outros, como mostra 4.14(b).

A busca de notícias é feita nos dados persistidos no Firebase, funciona como um provedor de conteúdo, completando algo como o CMS do tetraedro [48], através das funções disponibilizadas pela própria SDK (ou bibliotaca de dependência) que integra o aplicativo ao Firebase. As funções de busca são feitas informando a referência do banco de dados do aplicativo na plataforma, a folha de busca desejada no grafo do banco de dados e o parâmetro de busca desejado como por exemplo, em Figura 4.14(b) o título como "Teste". O trecho abaixo implementado no desenvolvimento descreve de maneira simplificada como isso acontece.

1. recupera referência de banco de dados do aplicativo partindo da raiz "post"

```
val database = FirebaseDatabase.getInstance().reference
    val titleRef = database.child("posts")
```

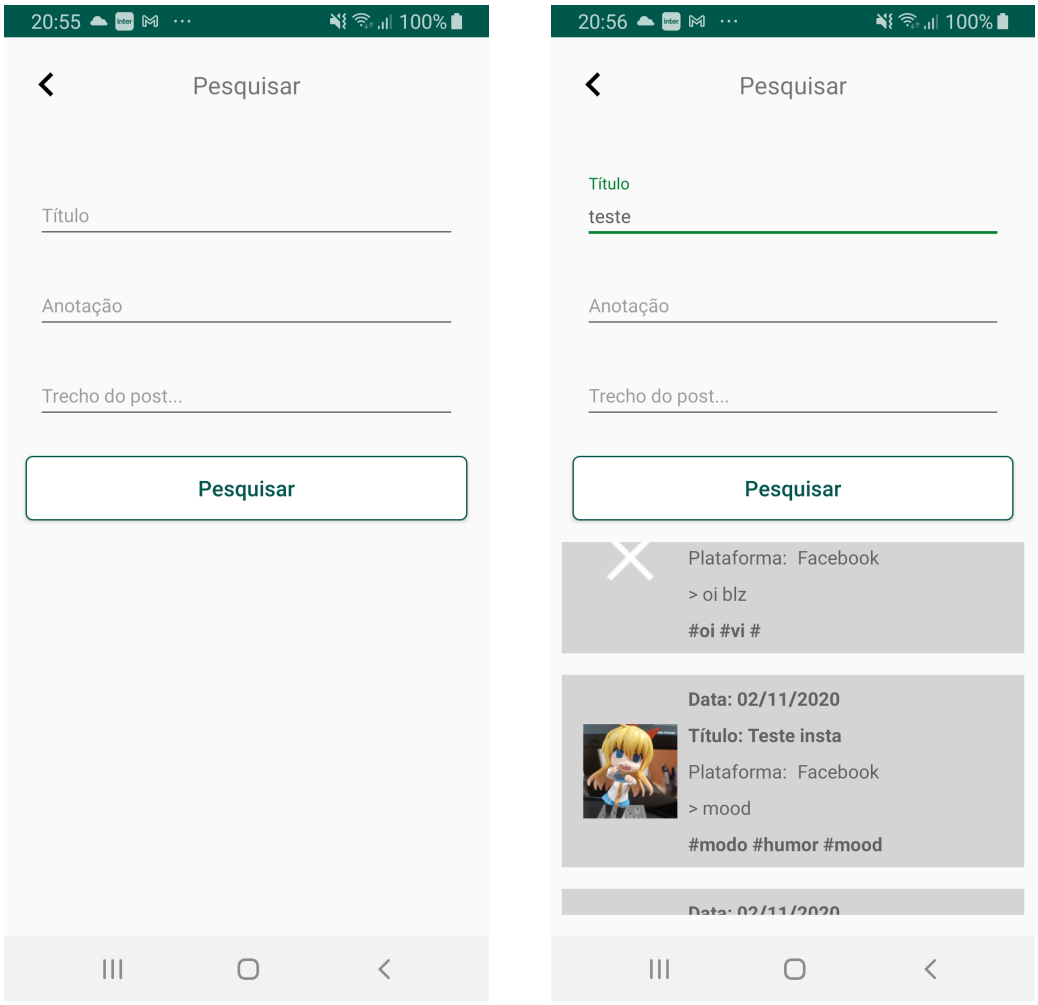
2. cria listener iterativo que recupera os dados e a partir da referência

```
titleRef.addValueEventListener(object : ValueEventListener {
```

(a) Gráficos de engajamento de publicações.

(b) Gráficos de engajamento de publicações sinalizando post com clique no número do post que descreve seu conteúdo.

Figura 4.13: Gráficos de engajamento das publicações.



(a) Tela de busca por notícia.

(b) Resultado de busca realizada por título "Teste".

Figura 4.14: Pesquisar notícia

```
                override fun onDataChange(snapshot: DataSnapshot) {
                 // trabalha com dados recuperados
                }
        }
```

3. compara dado passado para busca com dado recuperado no processo de busca e, se forem iguais, adiciona na lista de resultado da busca.

```
        if (post != null && !post.title.isNullOrEmpty()){
            if (post.title!!.contains(title, true)) {
                listPost.add(post)
            }
        }
```

O banco de dados também pode ser gerenciado e acessado pela plataforma do Firebase. Seu grafo é apresentado como em Figura 4.15.

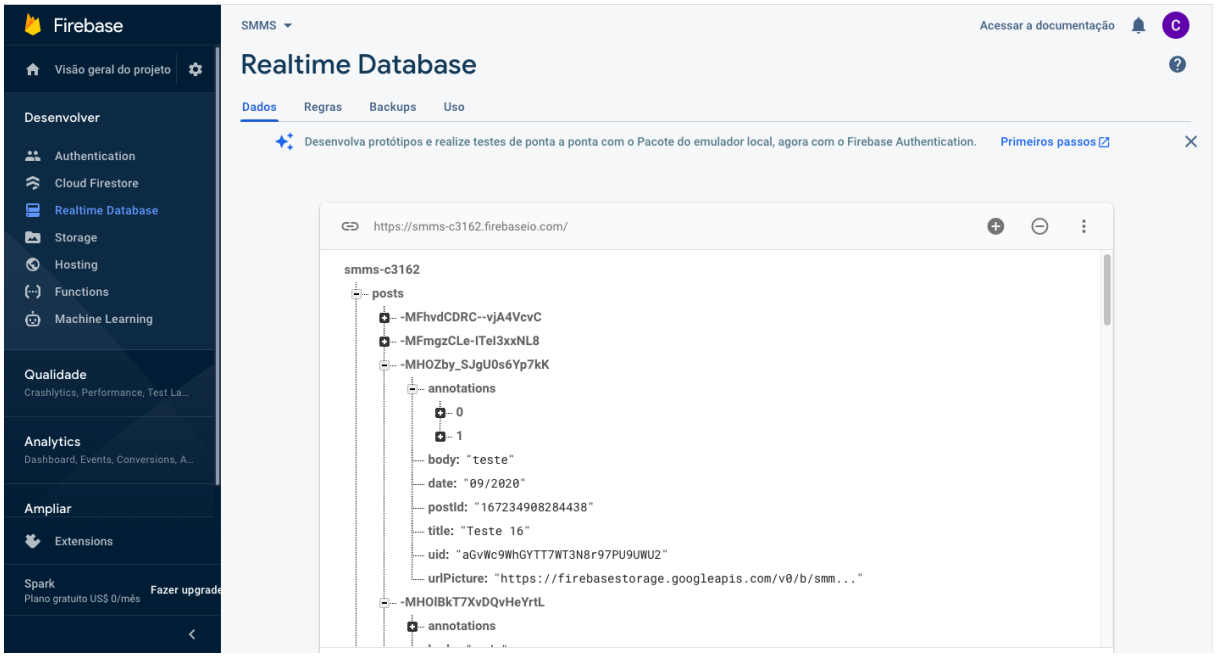
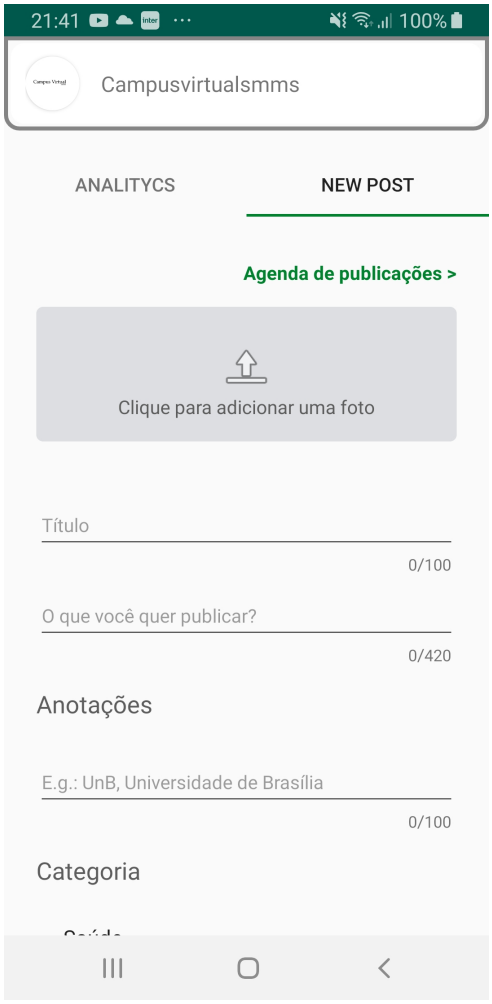


Figura 4.15: Representação do Banco de Dados na plataforma Firebase..

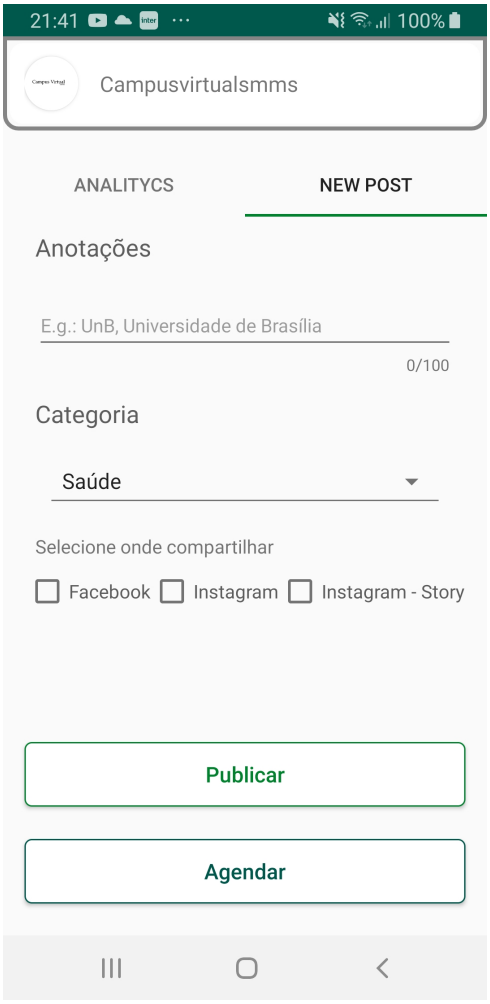
Por fim, temos a tela de publicação da notícia, localizada na aba "New Post". Representada pela 4.16.

As integrações com GraphAPI nessa tela são descritas abaixo:

feed: realiza publicação no Facebook informando o ID da página e no corpo da requisição um objeto com a mensagem da publicação. Esse serviço é chamado quando o usuário opta por não postar foto, apenas o texto.



(a) Apresentação superior da tela.



(b) Apresentação inferior da tela.

Figura 4.16: Publicação de novo post.

```
@POST("{page-id}/feed")
```

photo: realiza publicação no Facebook informando o ID da página e no corpo da requisição um objeto com a mensagem e url de foto da publicação.

```
@POST("{page-id}/photos")
```

Instagram media: realiza publicação no Instagram informando o ID da página do Instagram recuperado em **Instagram Account**, e no corpo da requisição um objeto com a mensagem e url de foto da publicação.

```
@POST("{ig-user-id}/media")
```

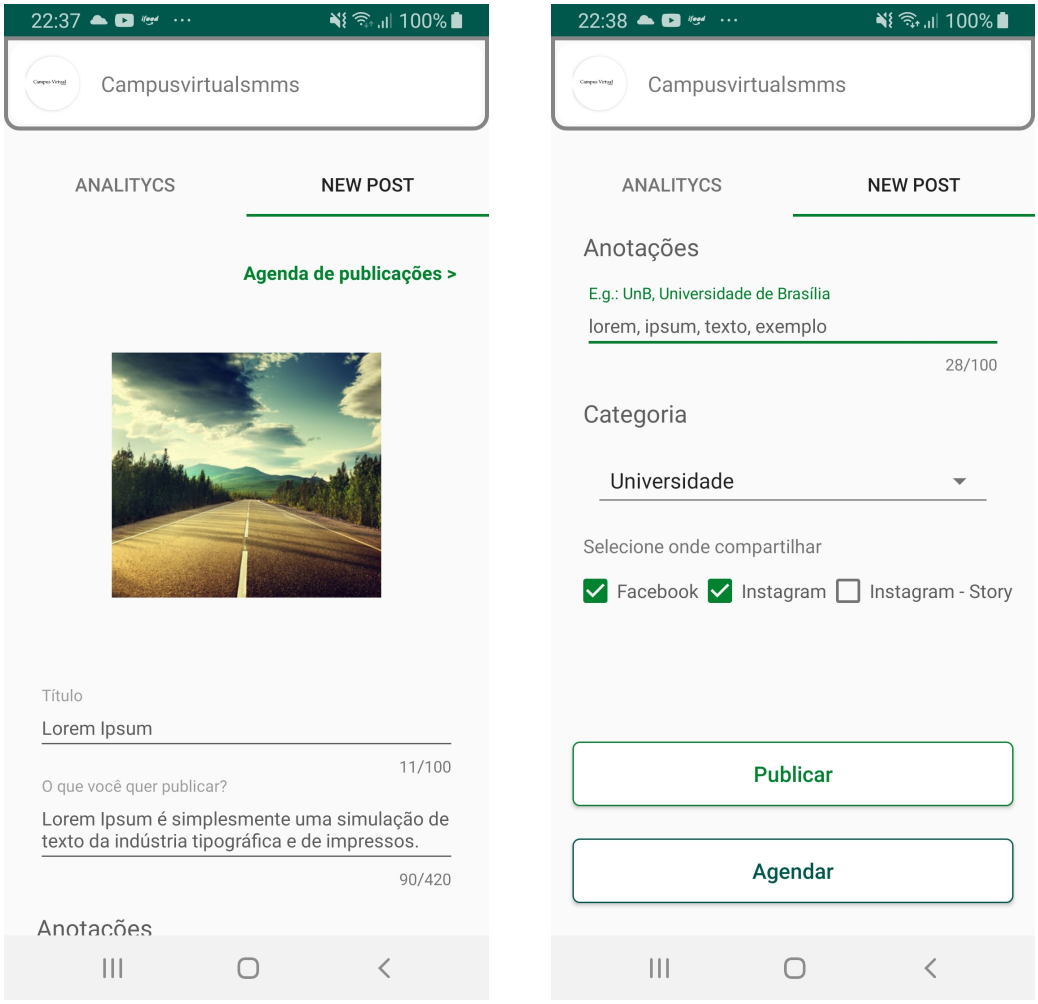
Para que a publicação seja feita no formato de mídias sociais e passado para as APIs listadas acima, um trabalho de concatenação e formatação entre as informações é feito na camada de interação (*Interactor*) do desenvolvimento do artefato. Por exemplo, ao publicar uma notícia o usuário seleciona uma foto de seu dispositivo e preenche os campos da tela com os seguintes valores:

- Título: Lorem Ipsum
- O que você quer publicar: Lorem Ipsum é simplesmente uma simulação de texto da indústria tipográfica e de impressos.
- Anotações: lorem, ipsum, texto, exemplo (assim mesmo, com vírgulas em palavras únicas isoladas como termos chaves que marcam a notícia)
- Categoria: Universidade

E depois de preencher com esses valores o usuário seleciona as plataformas onde gostaria de compartilhar a notícia: Facebook e Instagram. Representado em 4.17

Após clicar em "Publicar"e as informações forem concatenadas e formatadas uma publicação é feita automaticamente no Facebook e a mensagem é apresantada: "Publicado com sucesso no Facebook, aguarde enquanto redirecionamos para o Instagram! O seu texto foi copiado na área de transferência.". Uma vez que a GraphAPI depreciou a publicação automática nessa rede. Então, a tela do Instagram é aberta com a foto que ele havia selecionado e basta, no campo de legenda para a foto, selecionar a opção "Colar"do sistem que a mensagem digitada no aplicativo SMMS virá formatada devidamente. Por fim, a publicação pode ser vista em 4.19.

Observe que as anotações se transformam em *hashtags* com formatação usual e outros campos como Categoria e Título não aparecem na publicação final. Esses campos são usados para identificar e classificar as notícias postadas, são salvos no banco de dados do



(a) Apresentação superior da tela.

(b) Apresentação inferior da tela.

Figura 4.17: Publicação com dados preenchidos de novo post.

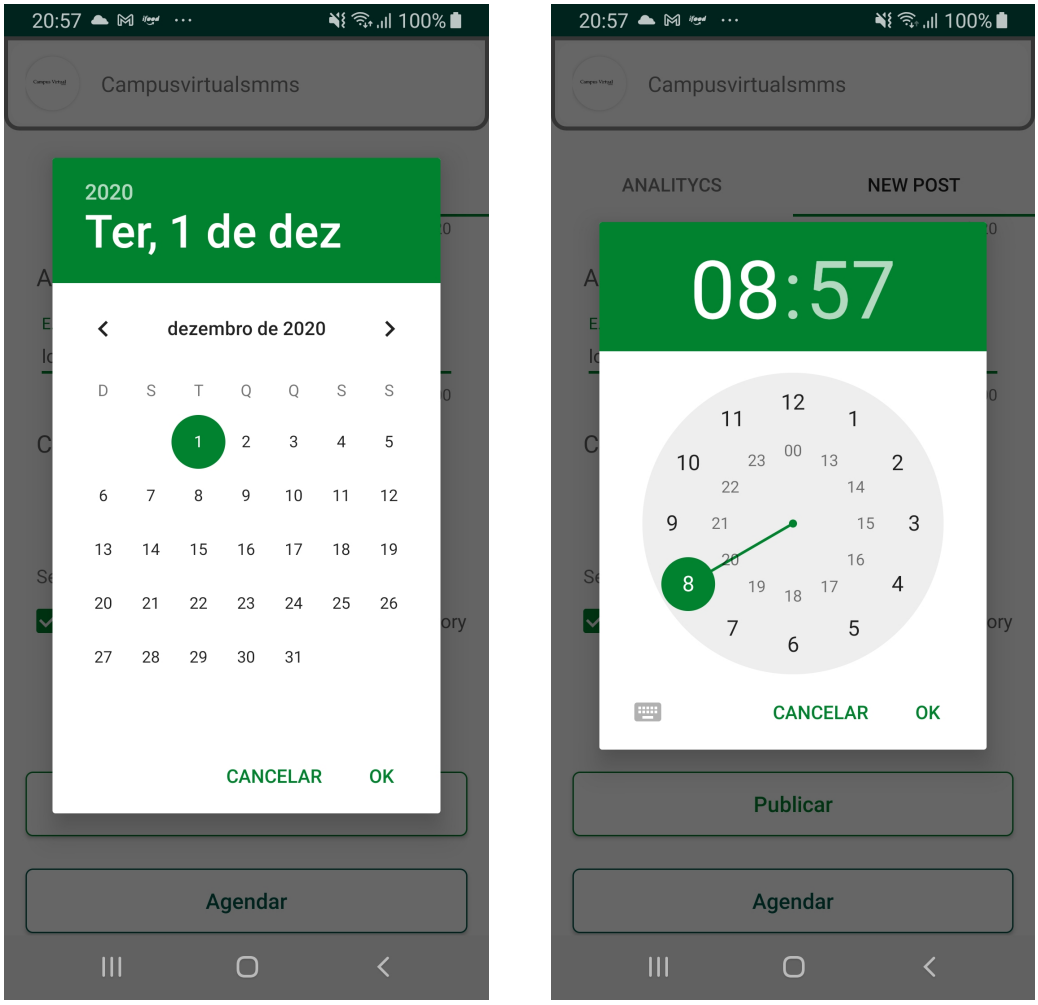
(a) Resultado da publicação no Facebook.

(b) Resultado da publicação no Instagram.

Figura 4.18: Resultado da publicação nas mídias sociais.

Firebase assim que a publicação é feita nas mídias sociais. Para a persistencia no Firebase os dados não há formatação. seus valores e chaves originais são mantidos como mostrado em Figura 4.15.

Outro diferencial do aplicativo SMMS é o agendamento de notícias. Essa funcionalidade também se encontra na página "New Post"e o usuário preenche os campos exatamente como faz para uma publicação comum selecionando ao final as plataformas às quais deseja efetuar a postagem. Para que o agendamento seja feito, ele clica no botão "Agendar". Feito isso, um componente chamado *dialog* aparecerá para seleção da data de publicação desejada, veja em 4.19(a), e após escolher a data, o próximo passo é selecinar o horário para publicação, 4.19(b). E a notícia é salva então no Firebase com status *pendente*. A Figura 4.20 ilustra como essa publicação é marcada no Firebase.



(a) Seleção de data para publicação.

(b) Seleção de hora para publicação.

Figura 4.19: Agendamento de publicação da notícia.

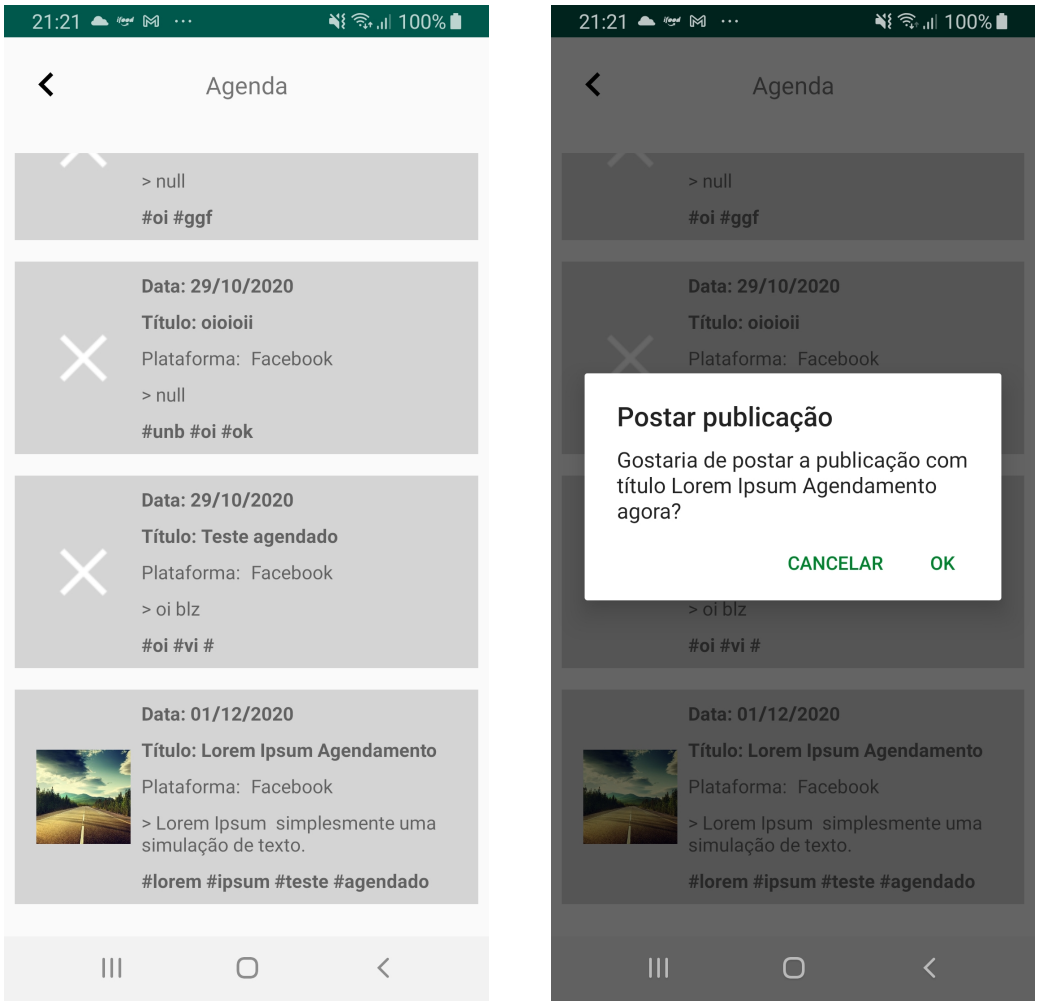
As publicações agendadas podem ser consultadas e publicadas antes de seu agendamento caso seja de interesse do usuário. Isso pode ser feito, ainda na tela "New

Figura 4.20: Publicação com marcação de agendamento, data e hora de postagem no Firebase.

Post"clicando no *link* "Agenda de publicações"localizado na parte superior da página que irá redirecionar para uma nova tela com título de "Agenda"onde uma consulta no banco de dados do Firebase é feita recuperando todas as publicações marcadas como pendentes. Essa consulta é feita de modo semelhante à descrita na Busca por Notícias, na parte de *Analytics*. A tela é vista como em 4.21(a), e caso o usuário deseje adiantar a publicação da notícia, basta selecioná-la e confirmar a publicação, ilustrado em 4.21(b). E, caso o usuário realize esse adiantamento da publicação, seu status é atualizado no Firebase retirando sua marcação como pendente e as chamadas à GraphAPI com as devidas formatações para postagem nas mídias sociais são feitas.

No caso de o usuário não realizar o adiantamento da publicação, uma notificação é enviada em seu dispositivo como um lembrete na data e hora marcadas por ele no processo de agendamento, como em Figura 4.22. Assim, o usuário clica nessa notificação e é direcionado para a tela de Agenda de Publicações (4.21) onde manualmente deve pubicar a notícia.

A opção por deixar essa última parte manual se deu pelo fato de que talvez uma notícia agendada pode se tornar não verdadeira até o momento de sua publicação comprometendo a integridade da página do jornal nas mídias sociais. Nesse sentido, o agendamento funcionando como um lembrete de publicação manual, o usuário deve, de certa forma,



(a) Tela de agenda de publicações pendentes.

(b) Adiantamento de publicação pendente.

Figura 4.21: Agenda de publicações pendentes.

revisar o que irá publicar.

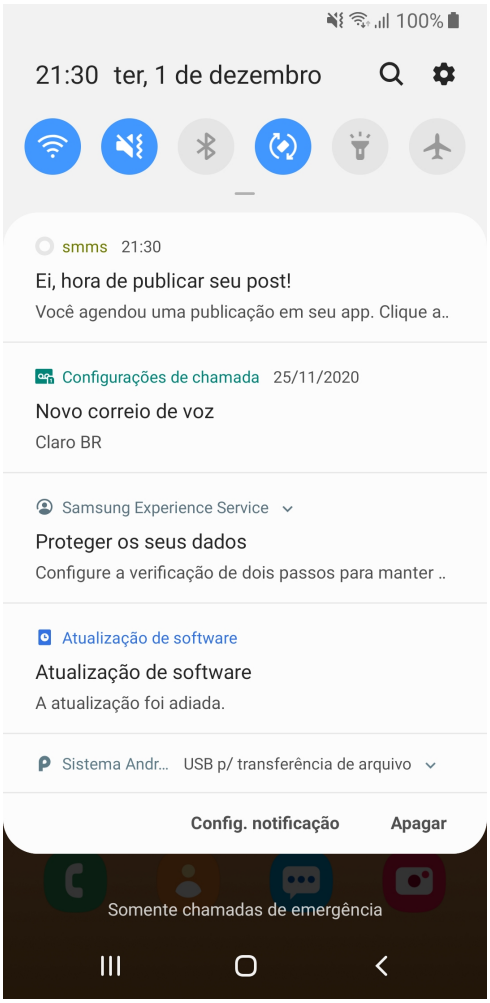


Figura 4.22: Notificação como alerta para publicação de notícia.

Capítulo 5

Conclusão

Os objetivos de implementação de prova de conceito foram alcançados. Usando apenas ferramentas de código aberto e estudando melhores formas de aplicá-las foi possível concluir o desenvolvimento do artefato, e foi possível também atender o objetivo geral proposto.

Para isso, estratégias de interação com jornalista utilizando a metodologia ágil com reuniões semanais na UnB possibilitou o entendimento e visão sobre como é feito o gerenciamento de mídias sociais no jornal, além de estudos e revisão de artigos sobre trabalhos e projetos relacionados à gerenciadores de mídias sociais. Que eram objetivos propostos especificamente ao trabalho como meio de se alcançar o objetivo geral.

Além disso, propostas com *design* e modelagem do artefato em formas de protótipo navegável realizados com ferramentas específicas para isso, auxiliaram na orientação da implementação do MVP, um aplicativo *mobile*, de forma a facilitar o desenvolvimento e apoiar como guia de funcionamento do aplicativo proposto. O protótipo aqui descrito pode ser acessado e melhor compreendido no Capítulo 4.

Um passo importante na pesquisa e desenvolvimento do projeto foi a personalização do modo de gerenciar as mídias sociais especificamente para redações jornalísticas, podendo o trabalho, ao final de tudo, ser realmente utilizado por uma organização jornalística como suporte à produção, monitoramento e análise das mídias sociais em seu contexto.

O MVP implementa funcionalidades básicas e fundamentais para um Social Media Management System (SMMS) com a capacidade de recuperar informações e elaborar cálculos a partir delas, com a capacidade de publicar e agendar publicações, e ainda foi desenvolvido com embasamento computacional e padrões de projeto que o fazem flexível para que, a qualquer momento, um desenvolvedor consiga facilmente adicionar novas funcionalidades que sejam convenientes ao jornalismo, além da fácil manutenção e gerenciamento do aplicativo em si.

A validação do funcionamento da ferramenta pode ser feita comparando os resultados obtidos e mostrados no aplicativo SMMS com os resultados reais das mídias sociais que forneceram essas informações. Ou, no caso de publicações feitas através do aplicativo SMMS pode-se ver o resultado conferindo no *feed* da rede social escolhida para publicação se ela foi realmente postada.

5.0.1 Contribuição

O artefato visa auxiliar, de fato, a um jornal no gerenciamento de suas publicações nas mídias sociais. Contribui com elaboração e compilação de informações e dados que podem ser utilizados como embasamento para o jornalismo para estratégias de publicação e para alcançar mais leitores. Além disso, contribui pelo fato de ser totalmente gratuita, ao contrário da maioria das ferramentas já existentes para o mesmo fim, como demonstrado em Capítulo 2, além do suporte técnico que pode ser assumido também pelos próprios alunos da universidade e segurança de concentrar todas as informações e publicações em uma plataforma confiável e não de terceiros que tem o poder de modificar e tomar posse dessas informações.

5.1 Avaliação

O trabalho resulta em um aplicativo funcional, com funcionalidades básicas demonstradas em forma de um produto mínimo viável, ou MVP, com os objetivos propostos alcançados. Sua interface com o usuário foi desenvolvida cuidadosamente seguindo padrões de outros aplicativos e tentando deixá-lo o mais próximo de mídias sociais, porém, sem conhecer algumas propostas do trabalho, o usuário pode ficar confuso ao ter que informar certos dados para uma publicação que ele normalmente não precisa, por exemplo.

O funcionamento e demonstração do protótipo, no escopo deste trabalho demonstra a sua viabilidade. A avaliação por parte de profissionais e jornalistas se fez inviável no tempo corrente de desenvolvimento do projeto mas pode ser feita e conduzida em um cenário experimental ou em um contexto real. Para isso, existirá a necessidade de comunicação entre o pesquisador, usuários e pessoas da organização na qual o artefato será testado. O resultado da avaliação poderá ser obtido através de uma nota sugerida pelos usuários sobre o aplicativo, como comumente é feito para esse tipo de aplicação, onde será possível extrair limitações do artefato e suas condições de utilização, ou seja, a relação do artefato produzido com o ambiente externo em que será utilizado, ao qual foi especificado durante a conscientização do problema [7]. Nesse sentido, essa etapa entra para Trabalhos Futuros, na seção seguinte.

5.2 Trabalhos futuros

Este trabalho foi realizado com intuito e previsão também de colaborar e fazer parte da implementação do *framework* para redação jornalística [1] compondo sua proposta de colaboração [49] utilizando o SMMS. Os resultados alcançados foram concretos, como todo trabalho que envolve uma implementação, mas, por ser um trabalho inicial, há muito que possa ser feito para melhorar o gerenciamento sobre as mídias sociais utilizadas pelos jornais.

5.2.1 Integração com mais mídias sociais

Devido à complexidade e tempo de integração para consumir os dados disponíveis pelas plataformas de mídias sociais, só foi possível realizar a integração do aplicativo SMMS com as principais mídias que são o *Facebook* e o *Instagram* através da Graph API. Aumentar a integração do jornal com mais mídias como *Twitte, WhatsApp, YouTube, dentre outros*, e até o próprio aplicativo do jornal universitário citado ao longo do trabalho [14], traria muito mais comodidade e suporte ao gerenciamento das redações jornalísticas.

5.2.2 Desenvolvimento para mais sistemas operacionais

O aplicativo desenvolvido é compatível apenas com dispositivos que possuem sistema operacional Android. Desenvolver para outros sistemas operacionais abrange o acesso dos jornalistas ao artefato de modo que possam gerenciar sem limitações de sistemas operacionais.

5.2.3 Integração com outros processos da Framework proposta em [1]

Utilizar métodos de anotações semântica e ontologias na publicação de notícias em mídias sociais utilizando ferramentas como a proposta em [37] resultariam em publicações mais embasadas em relacionamentos com outras publicações e consequentemente mais acurácia nas informações publicadas.

Referências

[1] Edison Ishikawa, Benedito Medeiros Neto, George Ghinea: *Newsroom 3.0: Managing technological and media convergence in contemporary newsrooms.* Relatório Ténico, 2, 3, 14, 15, 2018. ix, 34, 47, 65

[2] universoufesautor: *Jornalismo participativo: o leitor como elemento ativo na produção jornalística.* https://universo.ufes.br/blog/2018/06/jornalismo-participativo/, 2018. x, 1, 6

[3] Patel, Neil: *9 aplicativos de marketing de conteúdo essenciais para aumentar as visitas do seu blog.* https://neilpatel.com/br/blog/9-apps-essenciais-de-marketing-de-conteudo-que-irao-aumentar-o-trafego-do-seu-blog/, 2018. x, 6, 8, 9

[4] Sharma, Shivam Dutt: *Facebook graph api | python.* https://medium.com/analytics-vidhya/facebook-graph-api-python-3c8bab8a5a2a, 2020. x, 12

[5] Google, Inc.: *Arquitetura da plataforma.* https://developer.android.com/guide/platform?hl=pt-br. x, 15, 17

[6] Jeff Sutherland, J.J. Sutherland: *SCRUM: a arte de fazer o dobro do trabalho na metade do tempo*, volume 1. Editora Sextante, 2019. x, 22, 23

[7] Aline Dresch, José Antonio Valle Antunes Júnior, Daniel Pacheco Lacerda: *Design science research: método de pesquisa para avanço da ciência e tecnologia.* bookman, Porto Alegre, BR, 2015. x, 22, 26, 64

[8] Google, Inc: *Guia para a arquitetura do app.* https://developer.android.com/jetpack/guide. x, 16, 36, 37

[9] saasworthy: *Social media management software.* https://www.saasworthy.com/list/social-media-management-software, 2020. xii, 8, 9

[10] Alejandro, Jennifer: *Journalism in the age of social media.* Reuters Institute Fellowship Paper, 5:1–47, 2010. 1

[11] Lima, Alline Laís Silva *et al.*: *Perfil das redes sociais no jornalismo público: um estudo sobre a aplicação do manual de orientação para atuação em mídias sociais no ifpe.* 2018. 1

[12] Schultz, Tanjev: *Interactive Options in Online Journalism: a Content Analysis of 100 U.S. Newspapers.* Journal of Computer-Mediated Communication, 5(1), setembro 1999, ISSN 1083-6101. https://doi.org/10.1111/j.1083-6101.1999.tb00331.x, JCMC513. 1

[13] Edison Ishikawa, Benedito Medeiros Neto e: *Jour 3.0: A newsroom framework for journalists.* Relatório Técnico, Departamento de Ciência da Computação, 1, 2017. 1

[14] Ana Clara Alves, Ana Clara Cabeceira, Ana Luísa Santos Anny West Felipe Alves Carla Moura Jessica Cardoso Joana Diniz João Pedro Lima Karine Teles Maria Ferreira Matheus Marques Raquel Ribeiro Silvana Sousa Thales Alves e Wanessa Alves.: *Relatório técnico do conselho editorial do campus multiplataforma.* 2019. 1, 65

[15] Wazlawick, Raul Sidnei: *Metodologia de Pesquisa para Ciencia da Computacao.*, volume 4. Elsevier, 2009. 3

[16] Silva, Letícia Flávia da: *Webjornalismo colaborativo ou culto ao amador? - biblioteca online de ciências da comunicação (bocc).* http://www.bocc.ubi.pt/pag/silva-leticia-webjornalismo-colaborativo-ou-culto-ao-amador.pdf, 2011. 4

[17] Paiva, Ariane Parente: *A interatividade no jornalismo online para o conteúdo das notícias. O perfil interativo dos jornais de língua portuguesa: Folha de São Paulo (Brasil) e Público (Portugal).* Tese de Doutoramento, Faculdade de Ciências Sociais e Humanas, Universidade Nova de Lisboa, 2013. 4

[18] Fonseca, Virginia, Lindemann Cristiane Webjornalismo participativo: repensando algumas questões técnicas e teóricas: *Type of nosql databases and its comparison with relational databases.* Revista FAMECOS: mídia, cultura e tecnologia, 34:86–94, 2007. 4

[19] Aroso, Inês Mendes Moreira: *As redes sociais como ferramentas de jornalismo participativo nos meios de comunicação regionais: um estudo de caso.* Biblioteca online de cirências da comunicação, 2013. 5

[20] Rodrigues, Catarina: *Redes sociais e práticas que se impõem ao jornalismo.* Congresso Internacional de Comunicação, 3:368–377, 2010. 5

[21] Benetti, Rodolfo: *Redes sociais: em qual delas minha empresa deve investir?* https://www.organicadigital.com/blog/redes-sociais-para-empresas/, 2020. 6

[22] AL-Qurishi, Muhammad, Mabrook Alrakhami, Majed Alrubaian, Atif Alamri e Mohammed Al-Hougbany: *Online social network management systems: State of the art.* Procedia Computer Science, 73, julho 2015. 9

[23] Facebook, Inc: *Graph api.* https://developers.facebook.com/docs/graph-api/. 11, 35

[24] Albright, Jonathan: *The graph api: Key points in the facebook and cambridge analytica debacle.* https://medium.com/tow-center/the-graph-api-key-points-in-the-facebook-and-cambridge-analytica-debacle-b69fe692d747, 2018. 11

[25] Lima, Cíntia Caldas Barcelas de: *Aplicativos móveis de interesse público: limites e possibilidades para a cidadania no brasil*. Congresso Internacional de Comunicação, 2017. 12

[26] Luquetti B Silva, Daniel Facciolo Pires, Silvio Carvalho Neto: *Desenvolvimento de aplicações para dispositivos móveis: Tipos e exemplo de aplicação na plataforma ios*, 2015. `http://www.lbd.dcc.ufmg.br/colecoes/wicsi/2015/004.pdf`. 14

[27] Inc, Google: *Android - a plataforma que redefine o impossível*. https://www.android.com/, 2020. 14

[28] Dmitry Jemerov, Svetlana Isakova: *Kotlin em ação*. Novatec Editora, 2017. 17

[29] Simpson, Luke: *O que são bancos de dados nosql?* https://aws.amazon.com/pt/nosql/, 2017. 18, 19

[30] Rees, Robert: *Nosql, no problem - an introduction to nosql databases*, 2010. 19

[31] Bernadette Farias Lóscio, Hélio Rodrigues de Oliveira, Jonas César de Sousa Pontes: *Nosql no desenvolvimento de aplicações web colaborativas*. 2011. 19

[32] Toth, Renato Molina: *Abordagem nosql – uma real alternativa*. Universidade Federal de São Carlos – Campus Sorocaba, 2015. 20

[33] Monroe, Martin: *The gospel of mbaas*. 20

[34] Fowler, Martin: *UML Essencial. Um Breve Guia Para a Linguagem-Padrão de Modelagem Para Objetos*. Bookman, 2004. 22

[35] Beedle, Mike, Martine Devos, Ken Schwaber e Jeff Sutherland: *Scrum: An extension pattern language for hyperproductive software development*. dezembro 1998. 22

[36] Pontes, Thiago Bessa e Daniel Dias Branco Arthaud: *Metodologias Ágeis para o desenvolvimento de softwares*. Ciência e Sustentabilidade, 4(2):173–213, mar. 2019. `//periodicos.ufca.edu.br/ojs/index.php/cienciasustentabilidade/article/view/314`. 23

[37] Deus, Vitor Silva de: *Anotacao semi-automatica baseada em ontologia, busca e relacionamento semantico entre textos: Proposta para um sistema de gerenciamento de conteudo*. Universidade de Brasília, 2018. 27, 34, 65

[38] Longo, Hugo e Madalena Silva: *A utilização de histórias de usuários no levantamento de requisitos Ágeis*. julho 2014. 28

[39] Cohn, Mike: *User Stories Applied: For Agile Software Development*. janeiro 2004. 28

[40] Marvel, Inc: *Marvel app*. https://marvelapp.com/. 31, 47

[41] Square, Inc: *Retrofit*. https://square.github.io/retrofit/, 2013. 35

[42] *Json. javascript object notation*. http://json.org/. 35

[43] Laplante, Philip A.: *What Every Engineer Should Know about Software Engineering*, volume 1. CRC Press, 2007. 36

[44] Simpson, Luke: *Clean architecture with mvvmi, architecture components rx-java.* https://medium.com/@thereallukesimpson/clean-architecture-with-mvvmi-architecture-components-rxjava-8c5093337b43, 2017. 36

[45] Google, Inc: *Firebase.* https://firebase.google.com/. 39

[46] Ameya Nayak, Anil Poriya e Dikshay Poojary: *Type of nosql databases and its comparison with relational databases.* International Journal of Applied Information Systems (IJAIS), 5(4):16–19, 2013. 39

[47] Facebook, Inc: *Explorador da graph api.* https://developers.facebook.com/tools/explorer/. 43

[48] Neto, Benedito Medeiros: *Uma proposta de modelo para um framework semântico de ambiente colaborativo para gestão de informação em redação jornalística.* Proposta de pos-doutorament, 1, 2016. 51

[49] GHINEA, G; MEDEIROS NETO, B.; BRANDÃO M. D. F. R.; ISHIKAWA E.: *The communication, coordination, cooperation, and connection dimensions, when using framework and collaborative systems in the newsroom – a case study in the bbc london.* 65

Apêndice A

Algoritmo de Busca por Notícia

A.1 Algoritmo para Busca de Notícias

O algoritmo, em pseudocódigo, está representado abaixo.

Algorithm 1 Buscar notícias

```
1: function SEARCH(String title, String body, String annotation)
2:     if title == NULL then
3:         listPostsByTitle ← emptyList()
4:     else
5:         listPostsByTitle ← firebaseService.findByTitle(title)
6:     end if
7:     if body == NULL then
8:         listPostsByBody ← emptyList()
9:     else
10:        listPostsByBody ← firebaseService.findByBody(body)
11:    end if
12:    if annotation == NULL then
13:        listPostsByAnnotation ← emptyList()
14:    else
15:        listPostsByAnnotation ← firebaseService.findByAnnotation(annotation)
16:    end if
17:    return listPostsByTitle AND listPostsByBody AND listPostsByAnnotation
18: end function
```

Apêndice B

Entrevista

B.1 Entrevista com aluna de mestrado da Faculdade de Comunicação da UnB - Marília

Esta entrevista foi realizada em 23/10/2019. Marília, em 2021, ocupa a posição de Copywriter na Mindvalley. Em Kuala Lumpur, Malásia.

P: O quão interessante é um Gerenciador de Mídias Sociais para o jornal universitário?

R: Muito. Porque quando você vai para o mercado de trabalho, a idéia é exatamente preparar as pessoas para o mercado para que façam algo o mais profissional possível. E com certeza você vai usar um gerenciador de mídias sociais. Seja um para uma instituição ou para um jornal independente. Porque fazer a programação da notícia em si, é muito simples, mas quando você começa a raciocionar em cima dos resultados, ver o *analytics* e começar a agir em cima de dados, faz muito mais sentido. Então é bem interessante que eles comecem a trabalhar com isso.

P: Quais mídias são utilizadas como meio de divulgação da notícia pelo jornal universitário?

R: Atualmente usam o Facebook, Instagram, Twitter YouTube. Além disso, eles tem o site, o app e o podcast.

P: No contexto do aplicativo SMMS quais mídias seriam interessantes incluir e centralizar para o jornal universitário?

R: Descartaria o YouTube. Mas Instagram, Facebook e Twitter seriam ótimos.

P: Em níveis de prioridade, por questão de tempo e complexidade da integração com as plataformas de redes sociais, quais seriam as mais importantes para o aplicativo SMMS no seu ponto de vista?

R: Atualmente, o Facebook é muito importante, mas o Twitter tem assumido uma posição de muita importância também. Contudo, acho um erro que o Facebook seja mais utilizado que o Instagram, porque o Instagram é uma tendência.

P: No aplicativo SMMS o que seria importante para acompanhamento do Facebook?

R: Concentrar o processo de *Analytics* do Facebook e Instagram por hoje serem processos muito diferentes, por que o Instagram e o Facebook agem de maneiras bem diferentes mesmo. Por exemplo, no Facebook, para você ter sucesso, você precisa de compartilhamentos, então o compartilhamento se torna mais importante que a quantidade de *likes* na página. Os *likes* na página seriam uma consequência, não o foco. O que chamamo de *Analytics de vaidade*. Já no Instagram, a prioridade é caracterizada pelas "menções". Quando outra pessoa fala do seu perfil na página dela.

P: Ter no aplicativo um ranking representando quais publicações tiveram mais engajamento seria interessante?

R: Sim. Tomando como classificadores a quantidade de *likes*, depois de interações e equacionar esses fatores para inferir o engajamento das pessoas é muito interessante.

P: Centralizar o analytics e o gerenciador de mídias sociais em um único painel como o aplicativo SMMS é benéfico?

R: Facilitaria com certeza. Já acessando e compartilhando tudo no mesmo lugar. Teríamos contras como, de qualquer maneira teriamos de entrar em cada mídia específica para envio de mensagens nos chats das plataformas. Mas um ponto que seria muito interessante pra nós, é o compartilhamento de *stories* no Instagram.

P: Formatar as notícias antes de publicá-las, de forma a adequá-las para as mídias sociais é útil ou seria descartado?

R: É interessante sim, porque processos de menções no Instagram e no Facebook são bem diferentes, por exemplo. E, no Instagram, as hashtags são muito mais importantes do que no Facebook.

P: Agendar o dia e horário de uma publicação é interessante no aplicativo?

R: Sim. Muito.

P: É importante diferenciar publicações por editorias?

R: Seria legal. Não tinha pensado nisso. Mas é muito interessante, porque o público muda a cada semestre e conseguiriam entender como está o engajamento de cada categoria. Isso para mim é um diferencial.

Interrompemos a entrevista em seguida porque começaria uma nova aula na sala em que estávamos.

P: No seu ponto de vista, o que poderiam ser aplicados para os formatadores de hashtags e palavras chaves de uma publicação?

R: Tenho uma sugestão: fixar as *hashtags* que sempre usamos, em todas as publicações. Por exemplo, sempre adicionamos no Instagram as *hashtags*: unb, campus, campusonline, jornaluniversitario. Com a opção de excluí-las, caso o publicador não queira utilizá-las na publicação.

Apêndice C

Guia de Instalação e Utilização do MVP

C.1 Pré-Requisito

Para que as etapas posteriores sejam possíveis é necessário que o celular utilizado tenha sistema operacional Android.

C.2 Etapas

Foram feitos dois tutoriais de instalação. O primeiro contempla desde a configuração do projeto com seu código fonte até a instalação e uso do aplicativo. Para isso, fiz alguns vídeos para auxiliar todo esse processo e pode ser acessado em [1]. O segundo tutorial descreve um caminho mais rápido e simples, onde não é necessária a configuração do projeto desde seu código fonte e pode ser feita seguindo os passos abaixo:

1. Acesse diretamente de seu celular e faça o download do arquivo .APK disponibilizado em [2].
2. Quando concluído o download, selecione o arquivo baixado em seu celular e clique em "Instalar".
   (a) para alguns celulares, talvez seja necessário autorizar o download do arquivo nas permissões do aparelho.
3. Ao final da instalação, pelo fato do *app* não ser publicado na loja talvez emita uma mensagem de não verificação do *app*. Basta confirmar a autorização de execução.

[1] https://drive.google.com/drive/folders/1EUhvLwBRVyaLvP45b1KAtJJQR7lDEGwT?usp=sharing
[2] https://drive.google.com/file/d/1goq0zL72xnOGofaq7O-1YP8OK4CdrwGY/view?usp=sharing

4. Pronto! O app está pronto para ser utilizado. Para entender como usá-lo, acesse [3] e assista **a partir do vídeo 3** ou, se preferir, leia a seção 4.4 da monografia **Gerenciador de mídias sociais para redação jornalística**.

C.3 Dados importantes

Usuário utilizado para simular uma conta nas mídias sociais de uma redação jornalística. No Facebook e Instagram.

- usuário: campusvirtualsmms@gmail.com
- senha: Campus@123

[3] https://drive.google.com/drive/folders/1EUhvLwBRVyaLvP45b1KAtJJQR7lDEGwT?usp=sharing

Apêndice D

Configuração do sistema

D.1 Configuração e instalação do aplicativo SMMS

- Baixar e instalar o programa Android Studio, disponível em [1]
- no terminal de seu computador, em um diretório de sua escolha, clone o projeto através do comando: *git clone https://github.com/victoriagoularte/smms*
- importar o projeto clonado no Android Studio através dos passos:
  - abra o Android Studio
  - clique em "File", em seguida clique em "Open"e busque pelo projeto no diretório escolhido em sua máquina
  - aguarde enquanto é feita a sincronização e enquanto baixa as dependências
- para executar o aplicativo no celular, siga os passos abaixo:
  1. Conceda acessos de desenvolvedor no seu dispositivo Android selecionando Configurações - Sobre o telefone - Informações de software - selecione 7 a seção: Número de compilação
  2. conecte seu dispositvo através de um cabo USB ao computador
  3. selecione o ícone mostrado em Figura D.1

Figura D.1: Selecionar ícone apontado para executar.

[1] https://developer.android.com/studio?gclsrc=aw.dsgclid=Cj0KCQiAzsz-BRCCARIsANotFgPWQ8hBPlhv06LuMVl3z3wLepYir05kIM1ful-xqxx3EqOrpLspo7UaApIREALw_wcB

D.2 Testes e acesso ao aplicativo

- inicialmente, é necessário se logar no *Facebook* com a conta usada neste trabalho:
  - usuário: campusvirtualsmms@gmail.com
  - senha: Campus@123
- depois de logado no *Facebook*, é necessário se cadastrar no aplicativo SMMS (se já possuir cadastro, pule para o próximo passo) com um usuário à sua escolha. Utilizando um e-mail e senha e selecionando o botão "Cadastrar".
- preencha seu usuário e senha escolhidos para o aplicativo SMMS, clique em "Entrar"
- uma tela de permissões do Facebook será mostrada. Conceda as autorizações. E terá acesso a todos os recursos do app SMMS.

D.3 Sistema administrador Firebase

- acesse o painel da ferramenta em *https://firebase.google.com/*
- efetue o Login com usário e senhas:
  - usuário: campusvirtualsmms@gmail.com
  - senha: Campus@123
- para acessar a lista e gerenciar os usuários cadastrados para utilizar o aplicativo SMMS selecione a opção: **Ir para o console**, situada no canto superior esquerdo da página
  - selecione o projeto: smms
  - na próxima tela será mostrado o Dashboard com análises relacionadas ao aplicativo SMMS. Como a quantidade de usuários ativos, consumo de memória, dentre outros.
  - no menu lateral, à esquerda, selecione a opção "Authentication"que será possível administrar todos os usuários cadastrados.
- para acessar o banco de dados com os dados persistidos pelo aplicativo SMMS, também no menu lateral esquerdo, selecione a opção "Realtime Database"e o grafo com todas as informações de publicações será mostrado.