SINCRONISMO DE DADOS EM DIFERENTES DISPOSITIVOS

Ailton Vicente Junior [1]

Ronan Gomes Junior [1]

Bruno Zonovelli[2]

Resumo

O objetivo deste trabalho é realizar teste sobre a qualidade do sincronismo de dados entre os dispositivos móveis. O sincronismo permite a conexão de dispositivos com a nuvem (Computação em nuvem) e salvar os dados, para que fiquem em segurança, caso ocorra algum problema físico com o equipamento, e é uma tecnologia de fácil acesso, pois é possível em qualquer lugar para conseguir acessar o que deseja. Atualmente os sistemas em nuvem estão sendo bastante utilizados, porém, todos estão acostumados a realizar algumas tarefas e já salvar os dados instantaneamente, porém não é sempre que é possível estar conectado à internet. Então, a ideia do sincronismo em vários dispositivos, é salvar os dados na memória do dispositivo, e assim quando estiver online, o sincronismo do banco de dados será realizado. Para avaliar o sincronismo de dados, foi desenvolvido uma ferramenta com o intuito de analisar a qualidade do sincronismo realizado. Tempo de resposta, a qualidade de sincronismo, se é realmente automático ou não, e qual o consumo de dados utilizando dois diferentes bancos de dados.

Palavras-chave: Sincronismo, Dados, Dispositivos.

[1] Acadêmicos do curso de Sistemas de Informação - Centro UNIVERSO Juiz de Fora

[2]Doutor, professor do curso de Sistema de Informação - Centro UNIVERSO Juiz de Fora

1 - Introdução

Atualmente pode-se dizer que a internet está presente em todos os lugares, em todos os dispositivos, mas a história mostra que nem sempre foi assim. Na década de 40, os militares tinham suas tecnologias, porem eram todas centralizadas. Por isso no final da década de 50 e início da década de 60, durante a Guerra Fria, o departamento de defesa dos Estados Unidos financiou um projeto para que o processamento e armazenamento das informações nos grandes computadores não fossem mais centralizados, e com isso permitissem a troca de informações entre eles. Logo a informação e o processamento não se restringiam mais a uma única base (PANTOJA, 2000).

Nos anos 2000, a partir de estudos, foi desenvolvido os sistemas distribuídos, que tem como objetivo disponibilizar a informação em diferentes servidores e posteriormente apresentá-la para o usuário (COULOURIS,2013). As tecnologias utilizadas no passado não permitiam que o acesso à internet ou até mesmo a velocidade de transferência de informações através da rede fossem realizadas de maneira instantânea, como é realizado atualmente.

Apesar do avanço da tecnologia e da comunicação através da rede, o atraso na transferência de dados ainda pode ocorrer por diversos motivos, como por exemplo, a utilização de *softwares* inadequados para realizar tal tarefa, ou até mesmo *hardware* que possuem limitações. Essas limitações foram sendo amenizadas à medida que os microprocessadores começaram a ser utilizados, uma vez que várias ações podem ser realizadas ao mesmo instante.

O sistema centralizado ainda é muito utilizado nos tempos de hoje. De acordo com Marina Mantinez(2019), um ponto importante no sistema centralizado, é a segurança. Somente uma máquina servidora faz um alto poder de cópia de segurança e recuperação do servidor. Já de acordo com Pedro Teixeira (2019), um ponto importante em sistemas distribuídos, é a possibilidade de ter vários computadores simples em uma rede, formando juntos um super computador.

Uma das maneiras necessárias para se realizar a troca de informações entre diferentes dispositivos é a internet, e com o passar do tempo a internet se tornou algo comum e essencial

na sociedade, fazendo com que a partir dos anos 2000 as grandes empresas fizessem um grande investimento em sistemas distribuídos.

Assim como os computadores foram sendo atualizados de acordo com a evolução e as necessidades, os sistemas de banco de dados também (RACHID, 2017). Nos tópicos posteriores, será apresentado o banco de dados onde foram realizados os testes. Além disso será apresentado o processo de sincronismo e a verificação da realização do sincronismo entre os dispositivos de maneira correta.

Neste artigo será desenvolvido um sistema para testar o sincronismo de dados em diferentes dispositivos e nele é utilizado o sistema de dados em nuvem (*cloud)*. De acordo com Mário Rachid (2017), o criador do *cloud* (Computação em nuvem) foi o cientista John McCarthy no início da década de 1960. O armazenamento de dados em *cloud* pode ser feito em serviços de qualquer lugar do mundo, a qualquer hora, não há necessidade de ser instalado nenhum programa físico no computador que está sendo usado. O acesso a esses programas e serviços é feito diretamente pelo acesso à internet, o usuário possui um login e senha e com esses dados ele consegue manipular o seu banco de dados em nuvem.

2 - Revisão da literatura

O banco de dados é um sistema de gerenciamento de arquivos, que possuem dados armazenados e após criar um sistema utilizando esse banco de dados, esses dados são processados e transformados em informação.

O banco de dados em nuvem é um grande exemplo de sistemas distribuídos, no qual se tem vários computadores interligados e trocando os dados entre si, porém o cliente não sabe de onde vem essa informação gerada através do processamento desses dados (MARTINEZ, 2020). Um dos maiores problemas com o banco de dados em nuvem é a possível falta de internet, porque os aplicativos que usam este banco, automaticamente param de funcionar. A partir desse problema, estão surgindo novas tecnologias, que são capazes de armazenar na memória cache do dispositivo uma pequena parte do banco de dados.

2.1 - Cloud Firestore

O *cloud firestore (Cloud Firestore, 2020)* é uma tecnologia desenvolvida pela Google, que mantém os dados armazenados na nuvem, junto com uma certa parte offline. Fica muito mais fácil desenvolver aplicações voltadas para a nuvem, visto que, um dos problemas em se ter uma aplicação na nuvem, seria o dispositivo ficar sem acesso à internet, no qual ele não irá mais funcionar, até que a conexão seja estabelecida novamente.

Segundo Todd Kerpelman (2017), o *cloud firestore* armazena os seus dados através de documentos, que são organizados em coleções. Com ele, o aplicativo faz consultas para recuperar as informações, podendo ler só os documentos e retornar toda uma coleção para o seu dispositivo. Independentemente de como ele trouxer os dados, se ficar offline, o acesso ao aplicativo será de forma normal, podendo consultar, inserir e alterar informações igual se faz quando está online. Quando o *cloud firestore* retornar esta coleção, a mesma ficará armazenada na memória cache do dispositivo, e na ausência da rede de internet, ele faz a consulta na parte que está offline.

O aplicativo pode até ficar um longo período de tempo sem acessar a internet, porém os dados não estarão atualizados, pois quanto mais tempo ficar sem internet, mais desatualizadas estarão as informações, podendo realizar buscas e retornar informações ultrapassadas, dependendo do aplicativo. Quando o dispositivo se conecta à internet, automaticamente, a aplicação faz o sincronismo dos dados offline com os dados online, mantendo sempre as informações atualizadas (KERPELMAN, 2017).

Segundo Alex Dufetel (2019), o *Cloud Firestore* é um banco de dados *NoSql* (originalmente se referindo a "no SQL": "não SQL" ou "não relacional", posteriormente estendido para *Not Only SQL* - Não Somente SQL) hospedado na nuvem, no qual os aplicativos ou sistema podem acessar facilmente de acordo como os *SDKs (Software Development Kit)* nativos de si mesmos. Está também disponível em *SDK's*, nativas como Node.js, Java, Python e Go, bem como em *REST* e *RPC APIs.* O *Cloud Firestore* em si foi desenvolvido especialmente para as aplicações no IOS, Android e Java Script, fazendo com que somente os aparelhos móveis e as aplicações da *Web*, pudessem ter este recurso da Google ativo sem ter qualquer problema. Uma breve observação sobre a versão para a *web* é, se múltiplas abas no mesmo navegador forem abertas em uma mesma aplicação, somente a primeira página que foi aberta estará disponível o sincronismo de dados. As outras páginas não estarão com a persistência de dados ativa.

2.2 - Realtime Firebase Database

Como foi informado anteriormente, o *Cloud Firestore* é uma tecnologia que possibilita o usuário acessar determinado aplicativo ou página web, mesmo de forma offline. Similarmente existe o *Realtime Firebase Database (Firebase, 2020)*.

Realtime Firebase Database é um banco e dados NoSql hospedado em nuvem. Foi desenvolvido pela empresa Firebase, da Google. Este banco por ser usado para desenvolver aplicativo para diferentes plataformas, como IOS, Android ou JavaScript e sincronizar automaticamente todos os novos dados e atualizações (CLARK, 2020).

De acordo com Jessica Clark (2020), ele utiliza a tecnologia JSON para armazenamento de dados. Para isso o recurso permanece ativo utilizando o banco de dados no disco, porém com a ausência da internet, isso tudo se armazena na cache e quando é estabelecida a conexão com a internet, o sincronismo de dados é realizado.

O banco possui uma forma de autenticação, que é a regra de segurança do banco, onde por meio deste, os desenvolvedores determinam a estrutura de dados que os clientes podem ler (CLARK, 2020).

Para os desenvolvedores, é possível criar aplicativo atraentes enquanto utilizam o bando de dados. Já para os usuários, a utilização dos aplicativos sem ter acesso a internet, a persistência local é um recurso fundamental, pois com ele é aprimorado a experiência do usuário.

3 – Metodologias

A proposta do artigo é apresentar o funcionamento do sincronismo de dados em diferentes dispositivos, de forma *online* e *offline*, verificar se realmente essa solução é uma boa opção para os desenvolvedores, mostrando o nível de dificuldade para utilizar esse tipo de tecnologia. Será analisada a forma de sincronismo realizada pelo dispositivo, quanto tempo de resposta ocorre entre o aplicativo e o banco, a qualidade do mesmo, e se o consumo de dados é automático ou não.

Para isso, será desenvolvido um aplicativo na linguagem Kotlin (Kotlin, 2020), com a funcionalidade de criar notas rápidas, para que seja possível a realização dos testes. Esse aplicativo será desenvolvido utilizando os bancos de dados *Cloud Firestore* e o *Realtime Firebase Database*. A intenção é fazer uma comparação entre os dois serviços para verificar se existe diferença entre as duas soluções. Essa análise tem como objetivo a geração de relatório que ao final serão apresentados aos desenvolvedores, para que eles possam chegar a uma conclusão de qual solução mais confiável para utilização.

A métrica de tempo é feita pela média de acumulação do tempo do banco de dados (CPU + Espera) por sessões em primeiro plano do banco de dados durante um intervalo de tempo. Também conhecido como médio de sessões ativas.

Em tamanho do arquivo será medido o espaço máximo alocado no banco de dados em um determinado período de tempo. Dentro desta mesma métrica de tamanho do arquivo será medido o tempo de *upload* e de *download* por período de tempo em redes de dados móveis e redes Wifi.

4 - Resultados e discussão

Este trabalho possibilitou a compreensão e o entendimento de uma linguagem de programação e do conhecimento sobre dois diferentes bancos de dados e suas respectivas funcionalidades.

Kotlin foi a linguagem utilizada para o desenvolvimento do aplicativo de testes, é uma linguagem multiplataforma e multiparadigma. A linguagem chamou muita atenção dos desenvolvedores pois é compilado e executado em ambiente Java (Java, 2020), atualmente a empresa responsável pela linguagem Kotlin é a Google. Uma característica importante da linguagem é que ela é completamente interoperável com a linguagem de programação Java, ou seja, é possível utilizar código baseado em Java com o Kotlin ou usá-lo em código baseado em java.

Na ferramenta que foi desenvolvida é possível inserir um título e uma descrição e em seguida clicar no botão salvar para ser realizado o sincronismo direto com o banco de dados ou salvar na memória cache para posteriormente ser sincronizado. Foi colocado um contador

de tempo, que faz a contagem a partir do momento que o usuário clica no salvar e o dado é salvo no banco ou na memória cache.

Para definir a compreensão sobre o que foi apresentado, foi utilizado o aplicativo em dois diferentes dispositivos e testado o sincronismo de dados nos diferentes bancos o Cloud *Firestore e Realtime Firebase Database* e em diferentes situações de internet e sem ela. Cada teste foi realizado 10 vezes em cada dispositivo para assim conseguir as medias informadas. Logo a seguir será descrito a média dos testes realizados.

Foi realizado testes com internet e sem, no qual a média de tempo para salvar no banco de dados *Realtime Firebase Database*, utilizando a rede Wifi ficou em 0.0073s visto que por outro lado para salvar também por rede Wifi usando o *Cloud Firestore* ficou em 0.0027s. Os mesmos testes móveis, utilizando a rede de dados móveis nos dois dispositivos ficaram em 0.0048s para o banco *Realtime Firebase Database* e a média de 0.0096s para o *Cloud Firestore*.

Testes similares, porém, com os dois dispositivos sem internet, foi usado para testar se realmente salvava na memória cache do dispositivo no qual o banco *Realtime Firebase Database* ficou em 0.0038s e o mesmo teste de cache no banco *Cloud Firestore* ficou em 0.0025s. Todos esses testes e médias foram calculadas em segundos. Foi notado que os dispositivos se conectaram na internet e foi carregado o conteúdo automaticamente para o banco em nuvem.

A seguir será apresentado alguns gráficos relacionados aos testes realizados com as métricas de tempo para salvar nos bancos de dados, cada gráfico exibe um cenário de teste diferente do outro e nele é apresentado a média dos testes que foram realizados nos dois dispositivos e também foi realizado a média de desvio padrão, ela é utilizada para expressar o quanto um conjunto de dados se desvia da média de acordo com o que é mostrado nas linhas pretas de cada teste realizado.

Testes do *Cloud Firestore* e do *Realtime Firebase Database* em formas de gráfico de barras.

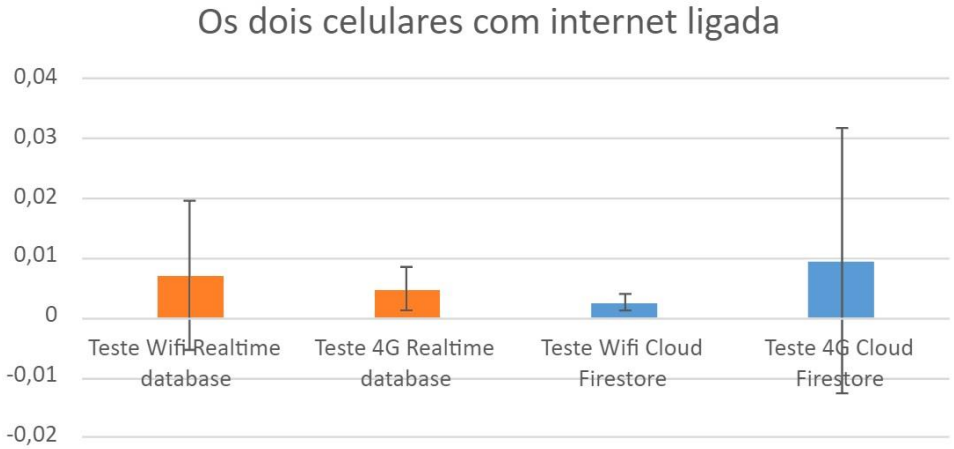

Figura 1: Gráfico com tempo gasto em rede wi-fi e rede de dados moveis nos diferentes bancos.

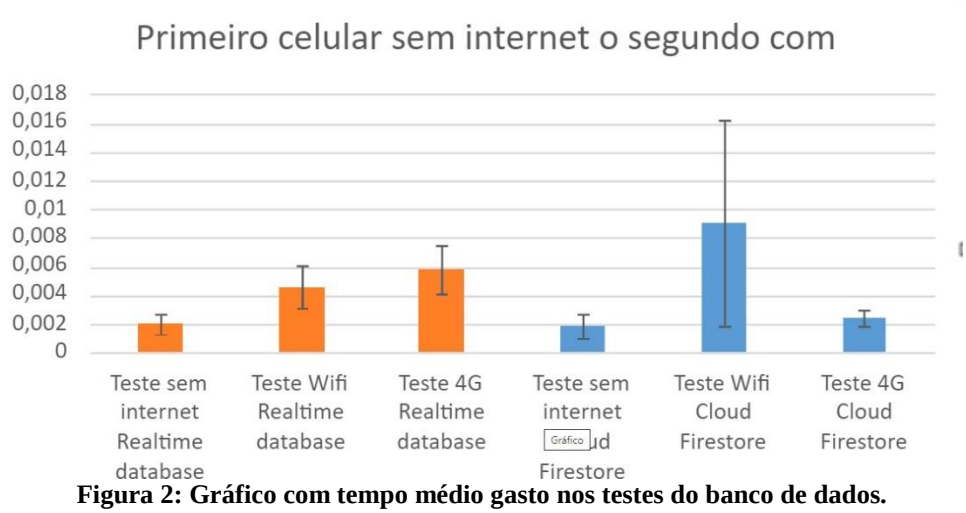

Figura 2: Gráfico com tempo médio gasto nos testes do banco de dados.

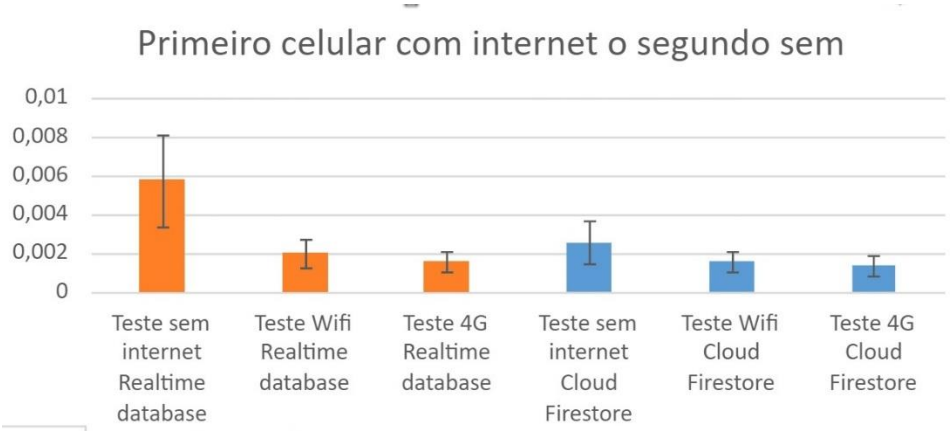

Figura 3: Gráfico de comparação de tempo gasto pelos bancos.

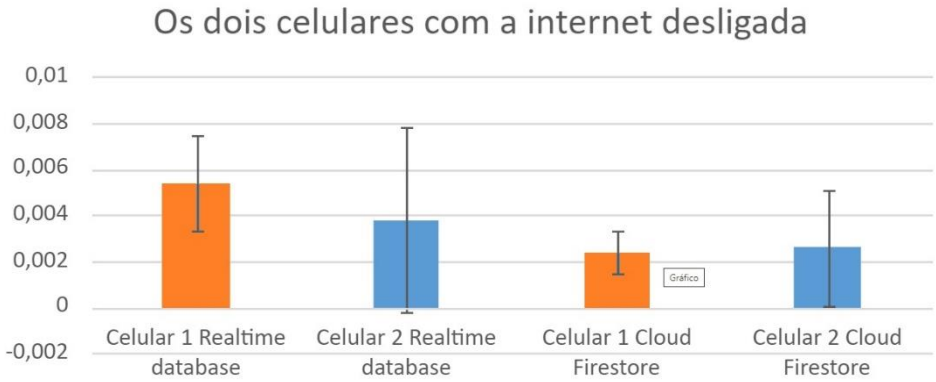

Figura 4: Gráfico com dados, dos diferentes bancos em diferentes celulares.

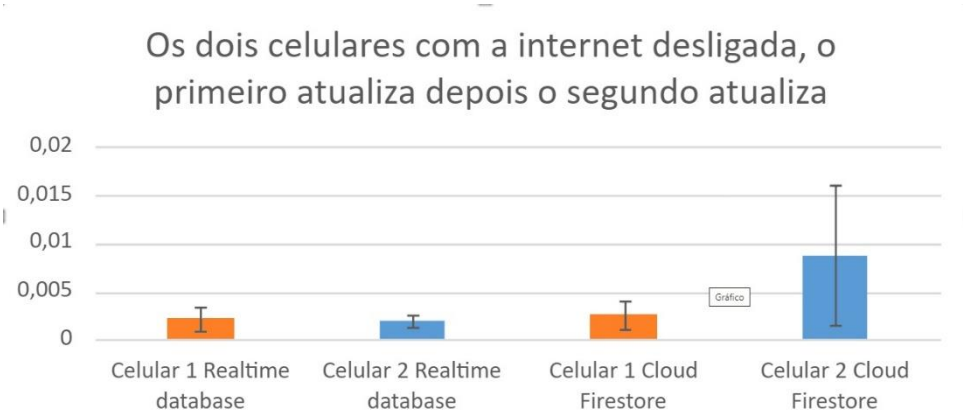

Figura 5: Gráfico com a diferença de tempo dos dois celulares.

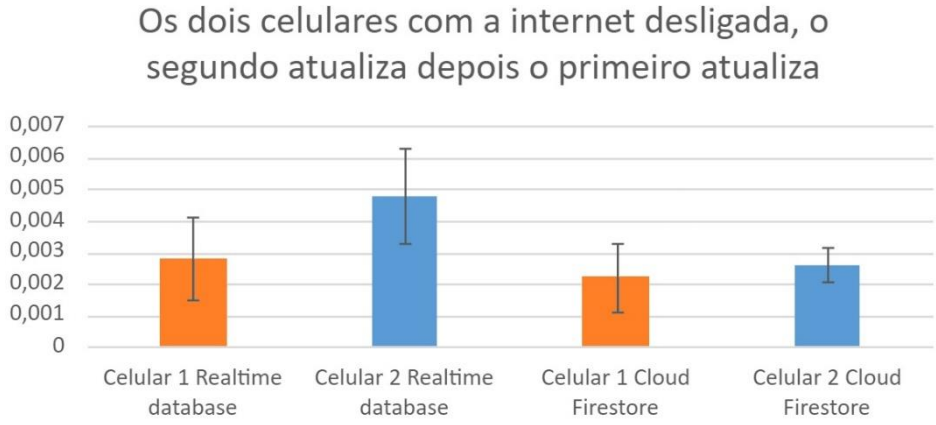

Figura 6: Gráfico com a apresentação do teste dos celulares sem internet e depois a ligando.

O desenvolvimento da aplicação com a conexão com banco *Cloud Firestore* teve um grau de complexidade maior que o outro pois ele é dividido em documentos e sub documentos. O grau de complexidade foi nesta parte de desenvolver a correlação entre os documentos dentro do banco de dados. Porem em questão de linhas e códigos, cada um ficou em cerca de 60 linhas.

5 - Considerações finais

O trabalho proposto tem como objetivo realizar testes sobre sincronismo de dados em diferentes dispositivos e testar para verificar a aplicabilidade do mesmo. Foi então realizado pesquisas e o desenvolvimento para testar alguns bancos que prometem fazer o sincronismo mesmo se faltar internet no dispositivo durante o uso do aplicativo.

Os testes foram realizados, e após a análise dos resultados, foi possível verificar que ambos têm resultados satisfatórios e bem parecidos entre si mesmo em diferentes situações, contudo, com o desenvolvimento foi analisado que o banco *Cloud Firestore* tem um maior grau de complexidade na hora do desenvolvimento, possibilitando um maior trabalho e maior demanda de tempo.

Com todos esses dados é possível decidir com mais clareza que ambos os bancos de dados, podem ser muito bem aplicados para diversas situações do dia a dia das pessoas. Garantido sempre os dados salvos de maneira segura.

Com os testes realizados a as análises feitas foi possível atingir o intuito inicial e concluímos que existem formas de sincronismos de dados e que realmente tem ótimos resultados.

REFERÊNCIAS

ALVES, Gustavo Furtado de Oliveira. A história dos bancos de dados. **Dicas de programação.** Online. 01 abr. 2013. Disponível em:<https://dicasdeprogramacao.com.br/a-historia-dos-bancos-de-dados/>. Acesso em 25 abr. 2020.

CLARK, Jessica. **Firebase Realtime Database vs. Firestore | Diferenças e Similaridades.** Online em 1 de outubro de 2020. Disponivel em :< https://blog.back4app.com/pt/firebase-realtime-database-vs-firestore-diferencas-e-similaridades/ >. Acesso em 1 out. 2020

Cloud Firestore; Disponível em:<https://firebase.google.com/docs/firestore> Acessado em: 30 abr. 2020.

COULOURIS, George. et al; **SISTEMAS DISTRIBUÍDOS Conceitos e Projeto**. 5° edição; Dados eletrônicos. – Porto Alegre : Bookman, 2013.

DUFETEL, Alex; **Apresentando o Cloud Firestore**: Nosso novo banco de dados de Documentos para Aplicativos.Online.18 out. 2019. Disponível em: <https://imasters.com.br/banco-de-dados/apresentando-o-cloud-firestore-nosso-novo-banco-de-dados-de-documentos-para-aplicativos>. Acesso em 25 abr. 2020.

Firebase Realtime Database, armazene e sincronize dados em tempo real; Disponovel em <https://firebase.google.com/products/realtime-database?hl=pt-br&gclid=CjwKCAiA7939BRBMEiwA-hX5Jx4CJUQ8sOLVLZ-4Bn5oAwYCjMA_6TN9iRiRDHFxe3e06IfDHPpPwRoCam8QAvD_BwE> Acessado em 30 abr de 2020.

Java Detalhes Técnicos do Java; Disponível em <https://www.oracle.com/br/java/technologies/> Acessado em 30 abr. De 2020.

KERPELMAN,Todd; **Cloud Firestore vs o Realtime Database:** Qual deles eu uso?. Online. 03 out. 2017. Disponível em:< https://firebase.googleblog.com/2017/10/cloud-firestore-for-rtdb-developers.html>. Acesso em:27 abr. 2020

Kotlin; Disponível em <https://developer.android.com/kotlin?hl=pt> Acessado em 30 abr de 2020.

MARTINEZ, Marina ;**Sistema de Informação Centralizado.** Online. [2006 a 2020]. Disponível em:<https://www.infoescola.com/informatica/sistema-de-informacao-centralizado/>. Acesso em: 27 abr. 2020.

OLIVEIRA, Evaldo; Disciplina de Sistemas Distribuidos - Material de apoio, <https://drive.google.com/file/d/0B8dL6ic8MJIkVk0zQ0t3ekZEb2M/view>. Acesso em 29 abr. 2020.

ORLANDI, Claudio; Planejando uma aplicação Offline First no React Native. **Rocketseat**. Blog, Online. 01 dez. 2019. Disponível em:<https://blog.rocketseat.com.br/offline-first-em-detalhes/> Acesso em 29 abr. 2020.

PANTOJA, Sônia; FERREIRA, Rosângela: Evolução da Internet no Brasil e no Mundo. Online. 24 jun. 2020. Disponivel em <https://www.faeterj-rio.edu.br/downloads/bbv/0032.pdf> Acesso em:24 jun de 2020.

RACHID, Mario; Uma breve história da Cloud Computing. Online. 20 jan. 2017. Disponivelaem<https://imasters.com.br/cloud/uma-breve-historia-da-cloud-computing>

Acesso em:30 abr. 2020.

SOUZA, Flávio ; MOREIRA, Leonardo ; MACHADO, **Java; Computação em Nuvem:** Conceitos, Tecnologias, Aplicações e Desafios. Online. 24 set. 2015. Disponível em:<https://www.researchgate.net/profile/Javam_Machado/publication/237644729_Computacao_em_Nuvem_Conceitos_Tecnologias_Aplicacoes_e_Desafios/links/56044f4308aea25fce3121f3.pdf>. Acesso em: 30 abr. 2020.

TEIREIRA, Pedro;Segurança em sistemas distribuídos. Revista **programar.** Online**.** 27 mai. 2007. Disponível em: <https://www.revista-programar.info/artigos/seguranca-em-sistemas-distribuidos/>. Acesso em 30 abr. 2020.