UNIVERSIDADE ESTADUAL PAULISTA "JÚLIO DE MESQUITA FILHO"
FACULDADE DE CIÊNCIAS - CAMPUS BAURU
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM CIÊNCIA DA COMPUTAÇÃO

HENRIQUE DE SOUZA SUMITOMO

DESENVOLVIMENTO DE APLICAÇÃO PARA AGENDAMENTO EM CLÍNICAS USANDO COMPUTAÇÃO EM NUVEM

BAURU
2017

HENRIQUE DE SOUZA SUMITOMO

DESENVOLVIMENTO DE APLICAÇÃO PARA AGENDAMENTO EM CLÍNICAS USANDO COMPUTAÇÃO EM NUVEM

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.
Orientador: Profa. Dra. Roberta Spolon

BAURU
2017

Henrique de Souza Sumitomo DESENVOLVIMENTO DE APLICAÇÃO PARA AGENDAMENTO EM CLÍNICAS USANDO COMPUTAÇÃO EM NUVEM/ Henrique de Souza Sumitomo. – Bauru, 2017- 45 p. : il. Orientador: Profa. Dra. Roberta Spolon
Trabalho de Conclusão de Curso – Universidade Estadual Paulista "Júlio de Mesquita Filho" Faculdade de Ciências
Ciência da Computação, 2017.
1. Computação em nuvem 2. Aplicações híbridas 3. NoSQL

Henrique de Souza Sumitomo

DESENVOLVIMENTO DE APLICAÇÃO PARA AGENDAMENTO EM CLÍNICAS USANDO COMPUTAÇÃO EM NUVEM

Trabalho de Conclusão de Curso do Curso de Bacharelado em Ciência da Computação da Universidade Estadual Paulista "Júlio de Mesquita Filho", Faculdade de Ciências, Campus Bauru.

Banca Examinadora

Profa. Dra. Roberta Spolon
Orientador

Universidade Estadual Paulista "Júlio de Mesquita Filho"
Departamento de computação
Faculdade de Ciências

Profa. Dra. Simone das Graças Domingues Prado

Universidade Estadual Paulista "Júlio de Mesquita Filho"
Departamento de computação
Faculdade de Ciências

Prof. Dr. Eduardo Martins Morgado

Universidade Estadual Paulista "Júlio de Mesquita Filho"
Departamento de computação
Faculdade de Ciências

Bauru, 08 de Dezembro de 2017.

Agradecimentos

Agradeço a minha família por todo o apoio, ajuda e incentivo. Especialmente aos meus pais, que sempre zelaram por mim estando sempre ao meu lado.

Aos meus amigos, por todo companherismo, diversão e momentos que ocorreram no decorrer de todos esses anos, mostrando que família não é apenas a de sangue.

A Profa. Dra. Roberta Spolon, não apenas pela orientação e pela liberdade dada para a execução do trabalho de conclusão de curso, mas também por me guiar e mostrar que mesmo não sendo o melhor dos alunos, é possível ir atrás de várias oportunidades que a área acadêmica proporciona, de que é posso ir além desde que eu queira e vá atrás.

Ao LTIA e ao Prof. Dr. Eduardo Morgado por proporcionar grandes oportunidades de conhecer um mundo e um ambiente que vão além do espaço universitário. Por toda experiência, risadas, e uma forma de aprendizado que valoriza o verdadeiro potencial dos alunos.

"Se os fatos não se adequam à teoria, mude os fatos."
(Albert Einstein)

Resumo

Atualmente, com o grande desenvolvimento e a popularização dos dispositivos móveis, a tecnologia distribuída tem feito parte do cotidiano da população mundial, modificando suas rotinas e formas de tomar decisões. Em diversos momentos, os dispositivos móveis são sinônimos de praticidade e conforto, uma vez que vem trazendo ao mundo moderno formas mais ágeis e práticas de se armazenar, compartilhar e visualizar informações e, em conjunto com a internet, hoje é possível acessar informação de qualquer lugar e a qualquer momento dentro de um dispositivo mobile. Em paralelo, as formas com que se armazenam informações e dados também vem se alterando nos últimos anos. Os custos e a complexidade para se ter e manter servidores locais têm tornado opções como a Computação em Nuvem mais atraentes, por trazer ao usuário uma solução em que não precisa se preocupar com sistemas operacionais, hardwares ou atualizações de software, onde os softwares e dados poderem ser acessados em qualquer lugar desde que haja acesso à Internet. É apresentado neste trabalho o desenvolvimento de uma aplicação móvel utilizando computação em nuvem para auxiliar o agendamento clínico, contendo os horários de consultas, históricos e informações de um determinado paciente. Para tanto, além da produção da aplicação, o relatório contempla também introduzir o conceito de computação em nuvem, formas para armazenamento de dados em uma aplicação mobile e a escolha de desenvolvimento de uma aplicação móvel híbrida.

Palavras-chave: Computação em Nuvem, Banco de dados, Dispositivos Móveis, Aplicação móvel híbrida.

Abstract

Today, with the great development and popularization of mobile devices, distributed technology has become part of the daily lives of the world population, changing their routines and the ways to make decisions. Sometimes, mobile devices are synonymous with practicality and comfort, since it has brought to the modern world more agile and practical ways of storing, sharing and visualizing information and, together with the internet, it is now possible to access information from any place and at any time within a mobile device. In parallel, the ways in which information and data are stored have also been changing in recent years. The costs and complexity of having and maintaining local servers have made options such as cloud computing more attractive by bringing the user a solution where they do not have to worry about operating systems, hardware, or software updates, where software and data can be accessed from anywhere as long as there is Internet access. With this assumption, it is proposed in this work the development of a mobile application using cloud computing to help the clinical scheduling, containing the schedules of consultations, histories and information of a given patient. To do so, in addition to the production of the application, this work also includes introducing the concept of cloud computing, ways to store data in a mobile application and the choice of development of a hybrid mobile application.

Keywords: Cloud Computing, Database, Mobile, Mobile Hybrid App.

Lista de ilustrações

Lista de tabelas

Lista de abreviaturas e siglas

TI	Tecnologia da Informação
SaaS	Software as a Service
PaaS	Platform as a Service
IaaS	Infrastructure as a service
CPU	Central processing unit
GPU	Graphics processing unit
SQL	Structured Query Language
NoSQL	Not Only SQL
ACID	Atomicity, Consistency, Isolation, Durability
RAM	Random Access Memory
DHT	Distributed hash table
SDK	Software development kit
iOS	iPhone Operating System
HTML	HyperText Markup Language
CSS	Cascading Style Sheets
NPM	Node Package Manager
CLI	Command-line interface
GPS	Global Positioning System
VM	Virtual Machine

Sumário

1 INTRODUÇÃO

Desde a década de 1960 já era imaginado a computação na forma de rede global (CANTU, 2011), mas apenas em 1997 o termo “computação em nuvem” foi utilizado pela primeira vez pelo professor de sistemas de informação, Ramnath Chellappa (CANTU, 2011).

O esqueleto para o que se conhece hoje como Computação em Nuvem, nasceu com os sistemas distribuídos, e além de ser uma coleção de computadores independentes entre si que se apresenta ao usuário como um sistema único e coerente (TANENBAUM; STEEN, 2007), a computação em nuvem também oferece um conjunto de serviços com capacidade de processamento, armazenamento, aplicações, plataformas e serviços disponibilizados na internet (TAURION, 2009). Desde então, diversos autores, organizações e instituições têm procurado definir a expressão "computação em nuvem". Bandyopadhyay (2009) define que a Computação em Nuvem é um modelo de serviço em TI no qual os serviços da computação (de hardware e software) são entregues sob demanda a consumidores conectados a uma rede, sob a forma de auto-serviço, independentemente de equipamentos e localização. Os recursos requeridos para prover os serviços com níveis de qualidade adequados são compartilhados, dinamicamente escaláveis, rapidamente disponibilizados, virtualizados e liberados com uma interação mínima com o provedor. Os consumidores pagam pelo uso dos serviços sem terem que incorrer em investimentos iniciais de porte, e os serviços em nuvem empregam sistemas de medição que dividem os recursos computacionais em blocos apropriados para a cobrança (BANDYOPADHYAY, 2009).

Dentre os serviços oferecidos pela Computação em Nuvem, encontra-se o armazenamento de dados online (ou em nuvem), que pode ser oferecido de forma gratuita ou paga por empresas que trabalham via Internet (ANDRADE et al., 2015). Mell e Grance (2009) destacam que a Computação em Nuvem pode ser dividida em três tipos de serviço: Software como serviço (SaaS), infraestrutura como serviço (IaaS) e plataforma como serviço (PaaS). Estes serviços proporcionam diversas vantagens e benefícios no uso pessoal ou empresarial, tais como: disponibilidade de arquivos em qualquer dispositivo com acesso à Internet, facilidade no compartilhamento com grupos, economia no consumo de recursos, menos problemas com os servidores etc (ARUTYNOV, 2012).

A redução de custos de implantação e operação de redes de mídia móvel ocorre de forma que os recursos do sistema podem ser alocados dinamicamente para atender a demanda de aplicativos em tempo real. Com armazenamento baseado na nuvem, uma organização pode comprar apenas a quantidade de armazenamento necessária e pagar uma tarifa mensal por esse armazenamento, em vez de ter que comprar um dispositivo físico caro que pode acarretar um custo inicial alto de capital, sem contar o custo adicional de alocação de espaço e energia. Esta

capacidade de "expandir conforme a necessidade" também ajudam a reduzir os custos iniciais (HONEY, 2015). Não apenas em armazenamento, mas também em relação a processamento, os recursos de computação (CPUs ou GPUs) em centros de processamento de dados podem ser instanciados em máquinas virtuais (VMs), cuja capacidade pode ser configurada dinamicamente para aplicações de mídia específicas (por exemplo, transcodificação, renderização, entre outros) (WEN et al., 2014).

Tendo a possibilidade de armazenamento e processamento de dados em um servidor online, uma aplicação utilizando computação em nuvem permite que uma grande diversidade de dispositivos com acesso a internet (tablets, celulares, notebooks e desktops), possam acessar e executar os recursos disponibilizados pela aplicação (VANDRESEN; MAGALHÃES, 2013), sendo assim, as informações podem ser acessadas de qualquer lugar, necessitando apenas de um dispositivo com internet.

Com a acessibilidade proporcionada pela computação em nuvem, é possível desenvolver aplicações para facilitar o acesso a determinadas informações como exemplo de aplicações que utilizam amplamente a arquitetura de computação em nuvem, é possível citar plataformas como o PayPal (PAYPAL, 2017), que oferece um serviço de transação de pagamentos e compras totalmente digital.

Essa acessibilidade não é vista atualmente em por exemplo, consultórios clínicos. A maioria dos consultórios clínicos possuem acesso a informação dos pacientes apenas por um computador local ou pelas fichas (cadastro, histórico) do respectivo paciente, utilizando arquivos físicos para armazenar tais informações onde muitas vezes os clínicos desejam verificar informações como por exemplo, quais consultas estão agendadas para os próximos dias e histórico ou informações sobre o paciente, em um ambiente fora do consultório clínico.

As formas físicas de armazenamento de informações como papéis e agendas demandam maior tempo na busca de informações, e o uso de softwares com infra-estruturas locais, embora possam facilitar a busca de informações, falham quanto à disponibilidade, visto que as informações são acessíveis de um único dispositivo, exigindo a presença do clínico no local onde este se encontra. Além disso, há também o custo e necessidade de manutenção dos equipamentos e servidores alocados para tal armazenamento.

Para facilitar a acessibilidade às informações, diminuir o gasto de tempo do clínico e os custos de manutenção, é possível aplicar computação em nuvem no desenvolvimento de aplicações móveis, uma vez que as informações não precisam estar mais armazenadas no computador do usuário ou em um servidor próximo. Esse conteúdo passa a ficar disponível na nuvem. Caberá ao fornecedor da aplicação todas as tarefas de desenvolvimento, armazenamento, manutenção, atualização, backup, escalonamento, etc (DILLON; WU; CHANG, 2010), tornando os serviços acessíveis ao usuário de maneira transparente, de modo que ele não precisa se preocupar com a infraestrutura do sistema e sim, com a utilização da aplicação. Ao utilizar-se uma aplicação móvel com computação em nuvem, o clínico terá acesso às

informações desejadas em qualquer lugar ou momento, desde que tenha um dispositivo com o aplicativo conectado a internet.

Tendo isto em vista, o presente trabalho apresenta uma aplicação que demonstra a aplicação dos conceitos de computação em nuvem, por meio de um sistema de agendamento de consultas médicas.

1.1 OBJETIVOS

1.1.1 Objetivo Geral

Desenvolver uma aplicação móvel utilizando computação em nuvem para auxiliar o agendamento clínico, consultas de históricos e informações do paciente.

1.1.2 Objetivos Específicos

- Realizar o levantamento de serviços de nuvem que poderão ser utilizados no desenvolvimento;
- Definir a arquitetura do sistema considerando o conceito de computação em nuvem;
- Definir os requisitos do sistema;
- Desenvolver a aplicação de agendamento de consultas;
- Realizar testes controlados da aplicação em ambiente real

1.1.3 Organização da Monografia

O presente trabalho divide-se em seções, sendo esta a primeira (Introdução). As demais seções estão dispostas na seguinte ordem:

- Seção 2 - Fundamentação Teórica: apresentação dos conceitos teóricos envolvidos no trabalho.
- Seção 3 - Metodologia: Descrição do método de pesquisa, desenvolvimento e ferramentas utilizadas.
- Seção 4 - Arquitetura do Projeto: Descrição da arquitetura do projeto.
- Seção 5 - Resultados: Apresentação dos resultados obtidos.
- Seção 6 - Conclusão: Considerações finais.

2 FUNDAMENTAÇÃO TEÓRICA

Nesta seção apresentam-se conceitos, fundamentos e teorias que dão suporte ao presente trabalho. Dentre eles, Computação em Nuvem, Banco de Dados Não Relacionais (NoSQL) e Aplicações Móveis Híbridas.

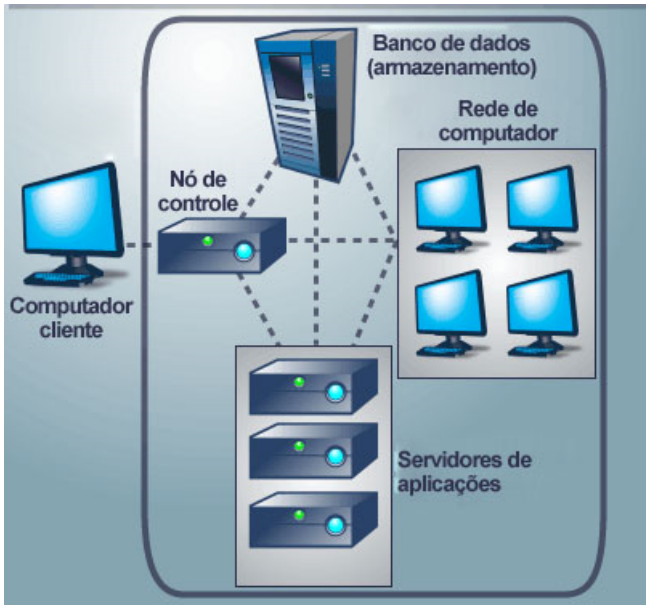
2.1 Computação em Nuvem

A computação em nuvem está se tornando uma das palavras chaves da indústria de TI. Surgindo da necessidade de construir infraestruturas de TI complexas, onde usuários não tem que realizar instalação, configuração e atualização de softwares, a nuvem é uma metáfora para a Internet ou infraestrutura de comunicação entre os componentes arquiteturais, baseada em uma abstração que oculta à complexidade de infraestrutura uma vez que, recursos de computação e hardware são propensos a ficarem obsoletos rapidamente (RUSCHEL; ZANOTTO; MOTA, 2010). Cada parte desta infraestrutura é provida como um serviço e, estes são normalmente alocados em centros de dados, utilizando hardware compartilhado para computação e armazenamento (BUYYA et al., 2009). Dentre várias definições propostas, o Instituto Nacional de Padrões e Tecnologia do Departamento de Comércio norte-americano definiu em 2011 que "Computação em nuvem é um modelo para permitir acesso ubíquo, conveniente e sob demanda via rede a um agrupamento compartilhado e configurável de recursos computacionais (por exemplo, redes, servidores, equipamentos de armazenamento, aplicações e serviços), que pode ser rapidamente fornecido e liberado com esforços mínimos de gerenciamento ou interação com o provedor de serviços."(OPUS, 2014).

A infraestrutura do ambiente de Computação em Nuvem geralmente é composta por um grande número de máquinas físicas ou nós físicos de baixo custo, conectados por meio de uma rede. Cada máquina física tem as mesmas configurações de software, mas pode ter variação na capacidade de hardware em termos de CPU, memória e armazenamento em disco (SOROR et al., 2010). A Figura 1 apresenta o funcionamento da computação em nuvem.

A Computação em Nuvem é uma evolução dos serviços e produtos de tecnologia da informação sob demanda, também chamada de Utility Computing (BRANTNER et al., 2008). O conceito de Utility Computing é baseado no pagamento na medida exata, pagando apenas o que é utilizado ou consumido. Seu modelo permite a aquisição de capacidade temporária de processamento e armazenamento de dados, potencializando a otimização da infraestrutura de hardware, software e serviços com redução os custos fixos por capacidade não utilizada (INTEL, 2017). Os usuários não precisam mais se preocupar com escalabilidade pois a Utility Computing fornece disponibilidade total para os usuários poderem ler e gravar dados a qualquer

Figura 1: Funcionamento da Computação em Nuvem.



Fonte: HowStuffWorks(2008).

tempo, sem serem nunca serem bloqueados (COUTINHO et al., 2013).

A partir disso, o modelo de computação em nuvem passou a ser composto por três características básicas, que são a virtualização, o compartilhamento dos recursos, a escalabilidade e a usabilidade (ANDRADE et al., 2015). Também foi desenvolvido com o objetivo de fornecer serviços de fácil acesso e baixo custo. Este modelo visa fornecer, basicamente, três benefícios. O primeiro benefício é reduzir o custo na aquisição e composição de toda infraestrutura requerida para atender as necessidades das empresas, podendo essa infraestrutura ser composta sob demanda e com recursos heterogêneos e de menor custo. O segundo é a flexibilidade que esse modelo oferece no que diz respeito à adição e substituição de recursos computacionais, podendo escalar tanto em nível de recursos de hardware quanto software para atender as necessidades das empresas e usuários. O último benefício é prover uma abstração e facilidade de acesso aos usuários destes serviços. Neste sentido, os usuários dos serviços não precisam conhecer aspectos de localização física e de entrega dos resultados destes serviços (SOUSA; MOREIRA; MACHADO, 2015).

Mell e Gance (2011) ampliaram o modelo da nuvem para cinco características essenciais: serviço sob demanda, amplo acesso à rede, agrupamento de recursos, elasticidade rápida e serviço medido.

Com o serviço sob demanda, o usuário pode adquirir unilateralmente recurso computacional, como tempo de processamento no servidor ou armazenamento na rede, na medida em que necessite sem precisar de interação com os provedores de cada serviço. O hardware e o software podem ser automaticamente reconfigurados de forma transparente para os usuários, podendo personalizar os ambientes computacionais de acordo com a necessidade.

O amplo acesso à rede encontra-se na disponibilidade de recursos por meio da internet, que devem ser acessíveis por meio de mecanismos padrões, permitindo seu uso por diferentes dispositivos e em qualquer localização, desde que tenha acesso a rede. A interface de acesso não obriga os usuários a mudar suas condições e ambientes de trabalho. Os recursos computacionais do provedor são organizados a partir de um agrupamento para servir múltiplos usuários, utilizando o modelo multi-inquilino (JACOBS; AUBACH, 2007) com recursos físicos e virtuais sendo arranjados e rearranjados dinamicamente conforme a demanda desses consumidores. Deve-se haver um senso de independência de localização, em que os usuários não precisam ter conhecimento da localização física dos recursos computacionais, podendo somente especificar a localização em um nível mais alto de abstração, como por exemplo país, estado ou centro de dados.

A elasticidade rápida encontra-se na forma com que os recursos são alocados, sendo de forma automática em alguns casos, permitindo a rápida adaptação caso haja a necessidade de escalar com o aumento da demanda, e liberados na retração dessa demanda. Para o consumidor, os recursos parecem ilimitados e podem ser adquiridos em qualquer quantidade a qualquer momento. Além disso, a virtualização auxilia a elasticidade rápida na Computação em Nuvem, criando várias instâncias de recursos requisitados utilizando um único recurso real (ABOULNAGA et al., 2009).

Por fim o serviço medido, onde serviços de computação em nuvem devem controlar e otimizar os recursos de maneira automática por meio de uma capacidade de medição. A automação é realizada em um nível de abstração apropriado para o tipo de serviço, tais como armazenamento, processamento, largura de banda e contas de usuário ativas, sendo possível monitorar, controlar e consultar o uso dos recursos, provendo a transparência para o consumidor e para o provedor de serviços.

Dentre o serviço medido, existem três modelos de serviços que definem um padrão arquitetural para soluções de computação em nuvem, sendo eles: Software como um Serviço (SaaS), Plataforma como um Serviço (PaaS) e Infraestrutura como um Serviço (IaaS). A Figura 2 mostra a diferença entre o que é gerenciado pelo usuário em cada tipo de serviço, desde um serviço que não utiliza nuvem até a maior abstração de um serviço de computação em nuvem.

Para os serviços de nuvem, primeiramente temos o de Infraestrutura como um Serviço (IaaS). Nesse modelo, o provedor disponibiliza toda a infraestrutura para o desenvolvimento, tornando mais fácil e acessível o fornecimento de recursos como rede, armazenamento, servidores, entre outros recursos que ajudam a evitar gastos e complexidade de comprar e gerenciar. Cada recurso é oferecido como um componente de serviço separado, deixando o usuário alugar o que for necessário pelo tempo que precisar. Dessa forma, o usuário terá controle sobre os sistemas operacionais, aplicativos implantados e eventualmente, componentes de rede, como firewalls. A Digital Ocean (OCEAN, 2017) e o Apache CloudStack (CLOUDSTACK, 2017)

são exemplos de provedores do serviço de IaaS.

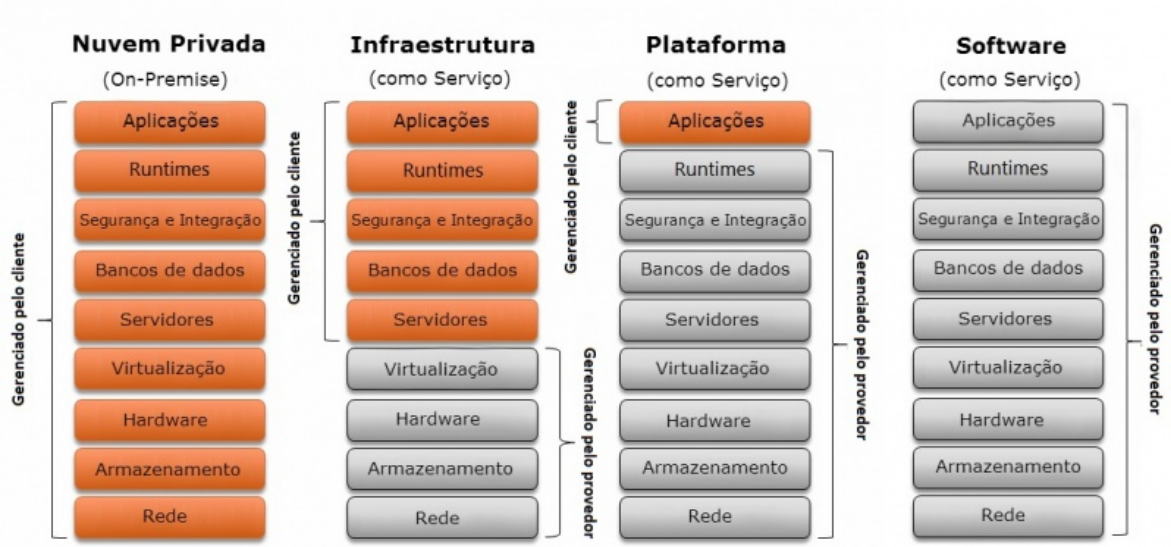
A Plataforma como um Serviço (PaaS) oferece não apenas a infraestrutura em baixo nível mas também a de alto nível de integração para implementar aplicações na nuvem. Fornece ao consumidor serviços e ferramentas para o desenvolvimento de aplicações, um ambiente escalável porém com restrições do tipo de software que se pode desenvolver, deixando-o isento de administrar, controlar ou se preocupar com qualquer infraestrutura básica, incluindo servidores e sistemas operacionais, sendo assim, uma forma de evitar gastos e a complexidade de comprar e gerenciar licenças de software, infraestrutura e middleware de aplicativo subjacente ou ferramentas de desenvolvimento entre outros recursos.. Como exemplo de PaaS, o Microsoft Azure (AZURE, 2017) e Firebase (FIREBASE, 2017a).

Por fim, o Software como um Serviço (SaaS), onde o usuário apenas consome as aplicações do fornecedor em uma infraestrutura da nuvem. Como o software está na Web, ele pode ser acessado pelos usuários de qualquer lugar permitindo maior integração entre unidades de uma mesma empresa ou outros serviços de software. Pode-se dizer que o SaaS representa os serviços de mais alto nível disponibilizados em uma nuvem, sendo estes aplicações completas apenas para uso, reduzindo custos como aquisição de licença de softwares. Google Docs (GOOGLEDOCS, 2017) é um exemplo de um SaaS.

Em relação a implementação de um serviço de computação em nuvem, existem quatro modelos de arquiteturas para nuvem: Privadas, comunidade, pública e híbrida (SHAMIM; SARKER; BAHAR, 2015).

Nuvens Privadas são caracterizadas por terem sua infraestrutura sendo utilizada exclusivamente para uma organização, sendo esta nuvem local ou remota e administrada pela

Figura 2: Quadro de Responsabilidades.



Fonte: IBGP (2017).

própria empresa ou por terceiros, utilizando de níveis de gerenciamento de redes, configuração dos provedores de serviço e utilização de tecnologias de autenticação e autorização para garantir as características de uma nuvem privada. O contrário ocorre em uma Nuvem Pública, onde a infraestrutura de nuvens é disponibilizada para o público em geral, sendo acessado por qualquer usuário que conheça a localização do serviço, sem restrições e técnicas para autenticação e autorização. Em uma Nuvem Comunidade, a infraestrutura é compartilhada por diversas organizações e suporta uma comunidade específica que partilha um mesmo objetivo. A Nuvem Híbrida combina duas ou mais nuvens, que podem ser do tipo privada, pública ou comunidade, e que continuam sendo entidades únicas, mas conectadas por meio de uma tecnologia padronizada que permite a portabilidade de dados e aplicações.

2.2 Banco de Dados Não Relacionais

Os bancos de dados não relacionais (NoSQL) surgiram como uma solução para a questão da escalabilidade no armazenamento e processamento de grandes volumes de dados na Web (CHANG et al., 2006). NoSQL é um termo usado para descrever bancos de dados não relacionais de alto desempenho utilizando diversos modelos de dados, incluindo documentos, gráficos, chave-valor e colunares. Bancos de dados NoSQL são amplamente reconhecidos pela facilidade de desenvolvimento, desempenho escalável, alta disponibilidade e resiliência (AWS, 2017b).

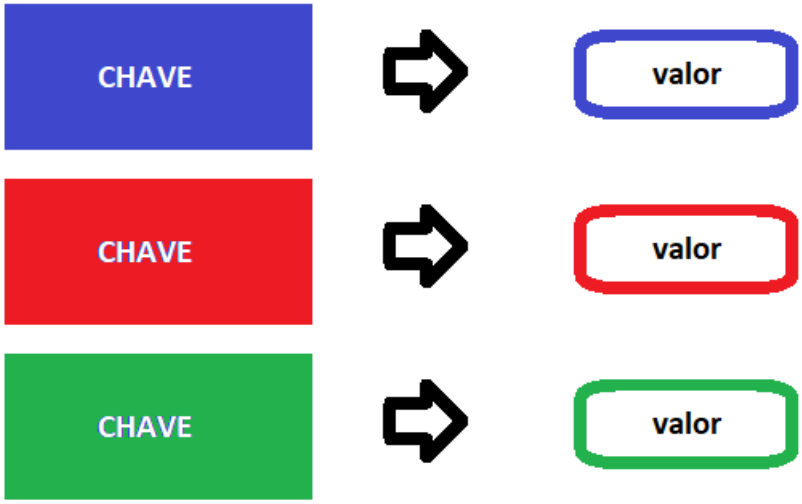
O modelo de banco de dados NoSQL pode ser caracterizado pela habilidade de escalar horizontalmente de forma simples entre muito servidores; poder replicar e distribuir dados de forma distribuída; uma interface de chamada simples; com um controle de concorrência um pouco mais fraco que as transações SQL que asseguram as propriedades de atomicidade, consistência, isolamento e durabilidade (ACID); ser eficiente em relação a distribuição de índices e memória RAM para o armazenamento de dados; e a habilidade de adicionar e salvar novos atributos de forma dinâmica aos registros de dados (CATTELL, 2011).

Baseado nessas características, pode-se definir diversas diferenças entre os modelos de banco de dados não relacionais dos modelos relacionais (SQL) como mostrado na Tabela 1.

Dentre as mais comuns formas de classificação de Banco de Dados Não Relacional, quatro delas se destacam, sendo estes orientados por Chave/Valor, Colunas, Grafos e Documentos.

Os banco de dados de esquema Chave/Valor trabalham com tabelas de hash distribuídos (DHT), mapeando chaves em valores contendo os dados distribuídos em múltiplos nós que mantêm informações sobre seus vizinhos. Nesse modelo, toda consulta ao banco de dados é feita apenas através de uma chave, que pode ou não ter algum valor relacionado como é mostrado na Figura 3. É muito usado na implementação de caches de sistemas ou acesso a informações que são alteradas em tempo real.

Figura 3: Banco de Dados Chave/Valor.



Fonte: Elaborado pelo autor.

Os bancos de dados orientados a documentos, são baseados no armazenamento de pares de chave-valor formando coleções. Pode-se definir como um agrupamento de atributos que não possui uma regra rígida para os tipos de cada atributo, tornando-os ótimas opções para dados semi estruturados como por exemplo, registro de eventos, análise web ou em tempo real e aplicativos de comércio eletrônico. Normalmente os documentos são salvos em um formato similar ao JSON mostrado na Figura 4, por ser um formato leve de transferência de dados, de simples leitura, tipado e com arquivo de tamanho reduzido.

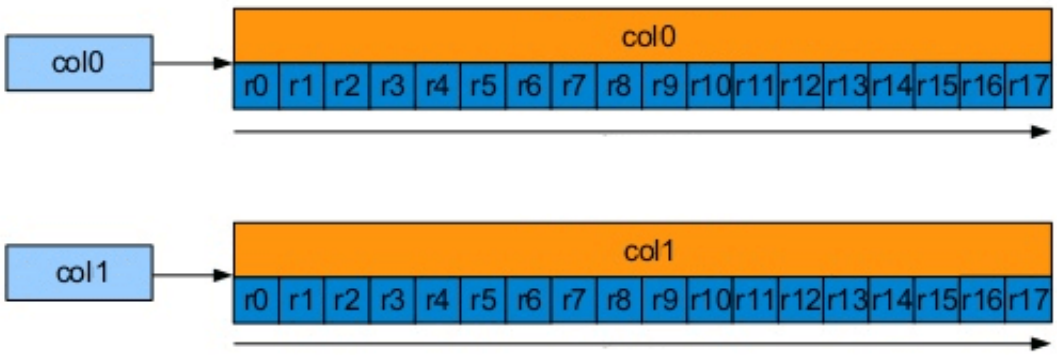
Figura 4: Banco de Dados orientado a documentos.

```
{
  "id": 55,
  "Pais": "Brasil",
  "Regiao":  "América do Sul",
  "Populacao": 201032714,
  "PrincipaisCidades": [
    {
      "NomeCidade": "São Paulo",
      "Populacao": 1182876,
    },
    {
      "NomeCidade": "Rio de Janeiro",
      "Populacao": 6323037,
    }
  ]
}
```

Fonte: imasters (2016).

Porém, os bancos de dados orientados a colunas já apresentam um modelo mais complexo que os de chave-valor, mudando o paradigma da orientação à colunas. Cada coluna é definida por uma tripla, contendo uma linha, uma coluna e um timestamp. Nesse modelo, é possível criar o encadeamento de colunas, chamadas de supercolunas como observado na Figura 5, e que sendo as colunas de uma mesma família, são armazenados no mesmo conjunto de arquivos, tornando-o eficaz para manter dados relacionados agrupados (OLIVEIRA, 2014). Podem ser utilizadas para registro de eventos, sistemas de gerenciamento de conteúdo e contadores.

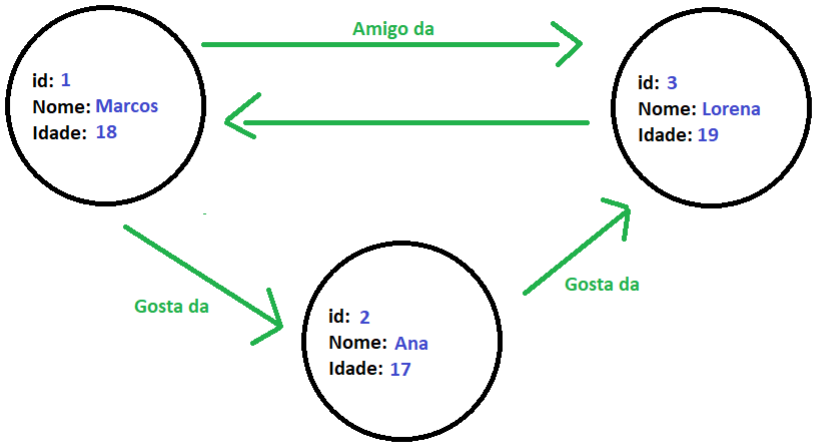
Figura 5: Banco de dados orientado a Colunas.



Fonte: imasters (2016).

Por fim, os bancos de dados de Grafos. Diferente dos modelos citados anteriormente que guardavam registros, o modelo orientado a Grafos tem como foco o armazenamento e relacionamento entre objetos. Não existem mais coleções, apenas vértices representando os objetos do sistema, sendo estes um agregados de atributos similar ao modelo de documentos; e arestas indicando o relacionamento entre eles como observado na Figura 6. São geralmente utilizados em sistemas que possuem dados conectados, roteamentos, envio e serviços baseados em localização, redes sociais e sistemas de recomendações.

Figura 6: Banco de dados orientado a grafos.



Fonte: Elaborado pelo autor.

2.3 Aplicações Móveis Híbridas

Antes de falar e definir o que são aplicações móveis híbridas, é preciso explicar o que são as aplicações nativas.

Uma aplicação móvel Nativa é aquela desenvolvida com as linguagens padrão do kit de desenvolvimento de software (SDK) do dispositivo o que facilita o acesso a funcionalidades do sistema operacional. No caso do Android, a linguagem padrão para desenvolver é o Java, e no caso do iOS, o Objective-C e mais recentemente o Swift (VASCONCELOS, 2017). Sua grande vantagem é a facilidade de otimizar o código por tratar diretamente com as bibliotecas do sistema operacional do dispositivo.

Porém, desenvolver aplicativos nativo necessita de desenvolvedores com conhecimentos mais específicos, aumentando o custo do projeto e, caso queira publicar a aplicação em um dispositivo com um sistema operacional diferente, será necessário refazer toda a aplicação adaptando-a para o sistema desejado. Por conta dessas razões, tem-se o desenvolvimento de aplicativos móveis híbridos.

As aplicações híbridas são desenvolvidas utilizando HTML, CSS e JavaScript, exigindo então apenas conhecimento de desenvolvimento web, diminuindo o custo de produção (CARLOS, 2016). Com esse tipo de desenvolvimento, é possível publicar aplicações independente da plataforma pois trabalha em conjunto com frameworks para facilitar a integração dos aplicativos híbridos com as funcionalidades do dispositivo desejado. Atualmente existem dois frameworks principais para integração, o Cordova e o Phonegap. Ambos possuem o mesmo propósito, que é de modo transparente para o desenvolvedor criar uma webview, um componente capaz de exibir páginas web sem que seja necessário abrir outro programa, para execução do HTML, CSS e JavaScript.

Tabela 1: Diferenças entre um banco de dados SQL e NoSQL. (AWS, 2017b)

	Banco de dados SQL	**Banco de dados NoSQL**
Modelo de dados	Normaliza dados em estruturas conhecidas como tabelas, que consistem em linhas e colunas. Um schema define estritamente as tabelas, colunas, índices, relações entre tabelas e outros elementos do banco de dados.	Não aplica um schema. Geralmente, uma chave de partição é usada para recuperar valores, conjuntos de colunas ou documentos semiestruturados JSON, XML ou outros que contenham atributos de itens relacionados.
Propriedades ACID	Sistemas de gerenciamento de bancos de dados relacionais (RDBMS) tradicionais são compatíveis com um conjunto de propriedades ACID: Atomicidade, Consistência, Isolamento e Durabilidade.	Geralmente troca-se algumas propriedades ACID por um modelo de dados mais flexível que escala horizontalmente tornando-o uma excelente opção em situações que precisam solucionar uma combinação de gargalos de desempenho, escalabilidade, complexidade operacional e custos crescentes de administração e suporte. Alguns autores sugerem o acrônimo "BASE": Basicamente disponível, "soft state", eventualmente consistente (CATTELL, 2011).
Desempenho	Depende do subsistema do disco. A otimização de consultas, índices e estrutura de tabela é necessária para alcançar máximo desempenho.	Geralmente é uma função do tamanho do cluster do hardware subjacente, da latência de rede e da aplicação que faz a chamada.
APIs	Mais fácil de aumentar a escala "verticalmente"com hardware mais rápido. Outros investimentos são necessários para tabelas relacionais para abranger um sistema distribuído.	Projetado para aumentar a escala "horizontalmente"usando clusters distribuídos de hardware de baixo custo para aumentar a transferência sem aumentar a latência.
Escala	As solicitações para armazenar e recuperar dados são comunicadas usando consultas compatíveis com "structured query language"(SQL). Essas consultas são analisadas e executadas por sistemas de gerenciamento de bancos de dados relacionais (RDBMS).	APIs baseadas em objetos permitem que desenvolvedores de aplicações armazenem e restaurem facilmente estruturas de dados na memória. As chaves de partição permitem que os aplicativos procurem pares de chave-valor, conjuntos de colunas ou documentos semiestruturados contendo objetos e atributos de aplicativos serializados.
Ferramentas	Os bancos de dados SQL normalmente oferecem um rico conjunto de ferramentas para simplificar o desenvolvimento de aplicações orientadas ao banco de dados.	Os bancos de dados NoSQL normalmente oferecem ferramentas para gerenciar clusters e escalabilidade. As aplicações são a interface principal com os dados subjacentes.

3 METODOLOGIA

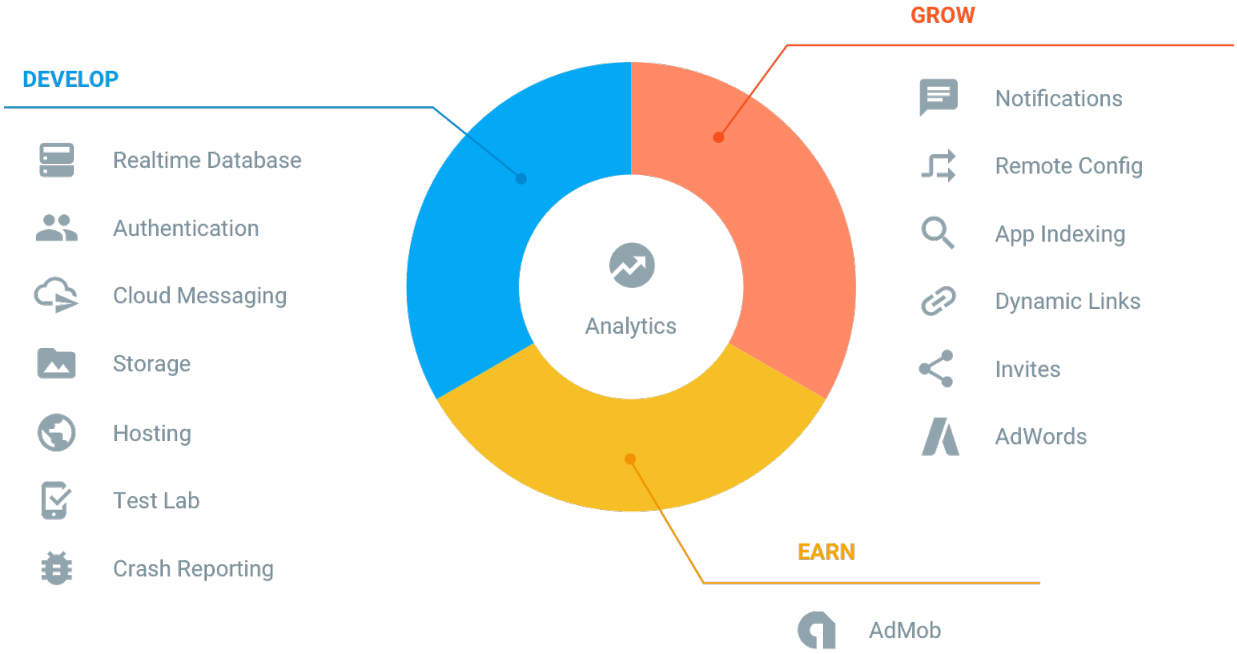
Neste capítulo estão descritos as ferramentas, metodologia utilizadas para o desenvolvimento do projeto e a apresentação do protótipo da aplicação móvel.

3.1 Tecnologias e Ferramentas Utilizadas

3.1.1 Firebase

Firebase é um serviço em nuvem para desenvolvedores, sendo um back-end completo para aplicações móveis e aplicações web, fornecendo ferramentas para desenvolver aplicativos de alta qualidade sem precisar gerenciar a infraestrutura, disponibilizando serviços para criação e testes de aplicativos, análises e interações com o público e também de desenvolvimento (FIREBASE, 2017a).

Figura 7: Serviços do Firebase.



Fonte: Firebase (2017a).

Dentre todos os produtos e serviços disponibilizados pela plataforma Firebase apresentados pela Figura 7, pode-se destacar para o desenvolvimento do projeto:

- **Realtime Database:** Armazenamento e sincronização de dados entre usuários e dispositivos em tempo real, utilizando um banco de dados NoSQL hospedado na nuvem.

Os dados atualizados são sincronizados em todos os dispositivos e além disso, os dados permanecem disponíveis caso seu aplicativo fique off-line, o que oferece uma ótima experiência do usuário, independentemente da sua conectividade de rede. Assim é possível desenvolver aplicativos sem precisar de servidores (FIREBASE, 2017b).

- **Autenticação:** Com esse serviço, é possível gerenciar os usuários de maneira simples e segura, utilizando diversos métodos de autenticação, inclusive e-mail/senha, provedores de terceiros, como Google e Facebook, ou uso direto do sistema de contas (FIREBASE, 2017c).
- **Cloud storage:** Serviço responsável por armazenar e compartilhar imagens, áudio, vídeo ou outro conteúdo gerado pelos usuários, simples e econômico criado para a escala do Google. Independentemente da qualidade da rede, esse serviço e capaz de pausar e retomar a transferência automaticamente conforme o aplicação perde e recupera a conexão móvel, economizando tempo e largura de banda dos usuários (FIREBASE, 2017d).

3.1.2 NodeJS E NPM

NodeJs é um interpretador de JavaScript open source implementado pelo Google, construído em cima da engine V8, que a linguagem JavaScript além do lado do browser, mas também pelo servidor (NODE, 2017). Sua arquitetura é conduzida por eventos assíncronos, projetado para criar aplicativos de rede escaláveis, utilizando um modelo de entrada e saída direcionada a evento não bloqueante tornando-se mais leve e eficiente, ideal para aplicações em tempo real com troca intensa de dados através de dispositivos distribuídos.

Além disso, conta com uma grande comunidade, utilizando-se de um repositório online para publicação de projetos abertos para Node.js. O NPM (Node Package Manager) é um utilitário de linha de comando que interage com esse repositório online, auxiliando no gerenciamento de pacotes, versão e dependências (NODEBR, 2017). Utilizando o NPM, é possível instalar pacotes a partir do console ou qualquer outra ferramenta CLI (Command-Line Interface).

Dentro de um projeto Node, encontra-se um arquivo chamado "package.json". Este arquivo armazena todas as dependências que o aplicativo possui e ao rodar o comando "npm install" na pasta raiz do projeto, todas as dependências listadas no arquivo "package.json" serão instaladas.

3.1.3 Ionic 3

Ionic é um framework open source para desenvolvimento de aplicativos móveis multiplataforma criado no final de 2013.É uma tecnologia que vem sendo empregada no desenvolvimento de aplicações web e mobile, especialmente pelo fato de ter como linguagens base o

HTML, CSS e JavaScript, essa tecnologia é comumente empregadas na construção do Front-end de soluções web. Por se tratar de um framework para desenvolvimento híbrido, o próprio Ionic se encarrega de aplicar as interfaces e padrões de dispositivos Android e iOS à aplicação. O Ionic traz consigo outros dois frameworks que simplificam o desenvolvimento do app, o Apache Cordova e o Angular, provendo uma solução de mais alto nível em termos de código e, consequentemente, de projeto (DEVMEDIA, 2017). A Figura 8 mostra exatametne o que framework Ionic contém e para quem é possível desenvolver.

Figura 8: Ionic, seus frameworks e para quais dispositivos desenvolve.

Fonte: DevMedia (2017).

Sua versão atual é o Ionic 3, e também traz consigo a versão do Angular 4 em conjunto com o Apache Cordova, permitindo aplicações menores, mais rápidas e também dá suporte a versões mais recentes do TypeScript, tornando-o compatível com o TypeScript 2.1 e 2.2 reduzindo assim os tempos de análise e compilação da aplicação. Também tornou-se possível utilizar "deep links", resolvendo o problema de que caso o usuário queira abrir um link externo, ele seria redirecionado ao navegador web do dispositivo. Utilizando o "deep link", os links passam a apontar para o conteúdo dentro do aplicativo sem precisar, por exemplo, abrir um navegador web. Com isso, facilitou a utilização de "lazy loading"na aplicação e a customização de cada página individual (IONIC, 2017). O mecanismo de "*lazy loading*" torna as entidades mais leves, pois suas associações são carregadas apenas no momento em que o método que disponibiliza o dado associativo é chamado. Portanto, o IONIC em conjunto de seus frameworks e diversos componentes tornam mais simples o desenvolvimento de uma aplicação híbrida.

3.1.3.1 Angular 4

O Angular é um framework originalmente JavaScript mantido pelo Google para o desenvolvimento de aplicações WEB. Ele se enquadra nos modelos MVC (Model-view-controller), MVP (Model-view-presenter) e MVVM (Model View View Model). O framework permite que o desenvolvedor utilize HTML como linguagem de modelo e permite estendê-lo para expressar

os componentes da sua aplicação (SILVA; VICENTE, 2014). A versão mais atual do Angular, é o Angular 4 onde desde a versão do Angular 2 que deixou de ser um framework JavaScript para ser TypeScript. O TypeScript é um superset (superconjunto) do JavaScript possibilitando a disposição de recursos que melhor suportam o uso de Programação Orientada a Objetos, oferecendo uma forma de contornar problemas como a escrita de classes de forma clara e a fraca tipagem de dados, adicionando funcionalidades que quando compiladas resultarão em código JavaScript novamente (JOSÉ, 2017). No exemplo da Figura 9 pode-se observar a criação de uma interface e uma classe em TypeScript:

Figura 9: Exemplo typescript.

```
interface IComponent{
    getId() : string;
}

class Button implements IComponent{
    id:string;
    getId():string{
        return this.id;
    }
}
```

Fonte: Schmitz (2015).

O código cria uma interface chamada IComponent e uma classe Button, essa classe implementa a interface e por isso o método getId() deve ser criado. Após compilado, irá gerar um código equivalente ao mostrado na Figura 10:

Figura 10: Saida em JavaScript.

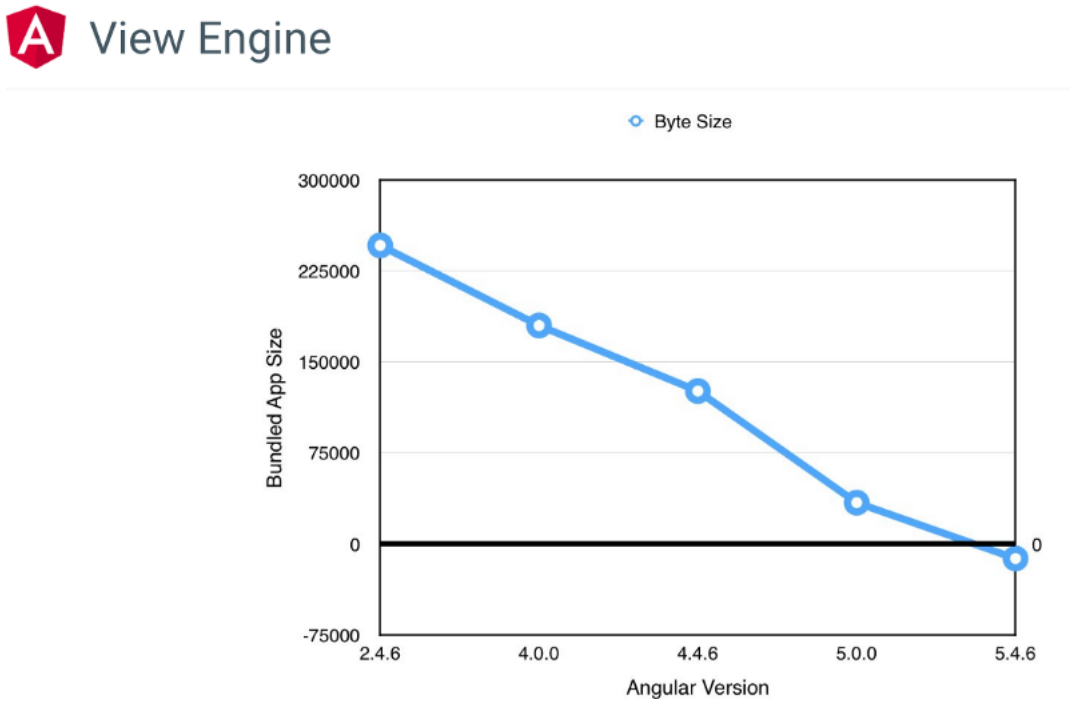
```
var Button = (function () {
    function Button() {
    }
    Button.prototype.getId = function () {
        return this.id;
    };
    return Button;
})();
```

Fonte: Schmitz (2015).

O código gerado é JavaScript, compreensível em qualquer navegador web (SCHMITZ, 2015). O Angular 4 também trouxe melhorias em suas funcionalidades em relação as suas versões anteriores, pode-se citar por exemplo, melhora na robustez nas rotas; nova "View Engine" Otimizando a geração de código como observado na Figura 11; se você não usa animação, o código desta API não é adicionado no bundle final; TypeScript na versão 2.1 é necessário. Esta versão irá melhorar a performance e verificação de tipos na sua aplicação;

segurança de rotas, especialmente para lazy loading module; desativar componentes que não sejam componentes de entrada, dentre várias outras melhorias.

Figura 11: View Engine.



Fonte: Passos (2017).

Evoluindo não apenas em questão de performance e velocidade dentro das plataformas web, o Angular 4 mostra controle sobre escalabilidade e também a possibilidade de desenvolver através de todas as plataformas, desde web, mobile web, mobile nativo e desktop nativo (ANGULAR, 2017).

3.1.3.2 Apache Cordova

Apache Cordova é um framework para desenvolvimento de aplicações móveis que permite os desenvolvedores de software a criar aplicações para dispositivos móveis utilizando CSS3, HTML5 e JavaScript. Quando se desenvolve com Cordova, cria-se um aplicativo híbrido, onde seu código pode ser compilado para diversas plataformas como Android, iOS, Windows Phone, FirefoxOS, BlackBerry, Tizen e Ubuntu; e todas as APIs para acesso de funções do dispositivo são instaladas na própria aplicação (LINO, 2015). Através do comando "build" do Cordova, o código não nativo (HTML, CSS e JavaScript) da aplicação é empacotado dentro de uma aplicação nativa.

A diferença entre o Cordova e o Ionic encontra-se que, enquanto o Ionic cuida do visual da aplicação, o Cordova é responsável por fazer o código JavaScript acessar os recursos nativos do dispositivo, como câmera, GPS, acelerômetro, etc (STACKOVERFLOW, 2017).

Sendo assim, o Cordova encapsula o código feito com IONIC, fazendo com que o app torne-se híbrido (CARLOS, 2016).

3.1.4 Métodos e Etapas

O desenvolvimento foi dividido em etapas. A primeira etapa se concentrou no estudo das tecnologias e conceitos necessários para o desenvolvimento da aplicação, a segunda etapa teve foco no levantamento de requisitos, a terceira etapa no desenvolvimento da aplicação e, por fim, a última etapa se concentrou nos testes em ambiente real e na finalização do relatório e dos resultados obtidos.

- **Estudo das tecnologias e conceitos**

  Essa fase do projeto se concentrou no levantamento dos tipos de serviços de nuvem como por exemplo, Apache CloudStack (CLOUDSTACK, 2017), Digital Ocean (OCEAN, 2017), Amazon Web Services (AWS, 2017a), Windows Azure (AZURE, 2017), Firebase (FIREBASE, 2017a) entre outros, formas de desenvolvimento de aplicações móveis que sejam eficientes em ambiente de nuvem e verificação do estado da arte. Além disso, foram levantados os tipos de tecnologias utilizadas no desenvolvimento de aplicações móveis.

- **Levantamento de requisitos**

  Nessa fase foi realizada a análise do ambiente de trabalho e quais são suas necessidades em relação a gastos de tempo e disponibilidade de acesso a informações sobre os pacientes em uma clínica de fisioterapia localizada em Bauru. Através desses dados, foram definidos os requisitos da aplicação, suas funcionalidades e por fim, o planejamento de seu fluxo de uso e telas. Também criou-se o planejamento da arquitetura do sistema levando em consideração os conceitos de nuvem estudados no item anterior.

- **Desenvolvimento da Aplicação**

  Nessa etapa foi desenvolvido uma versão produto mínimo viável (MVP), apenas com as funcionalidades necessárias para que ele cumpra a função para a qual foi planejado, com base no levantamento de requisitos e nas tecnologias escolhidas a partir do estudo feito nas fases anteriores.

- **Testes**

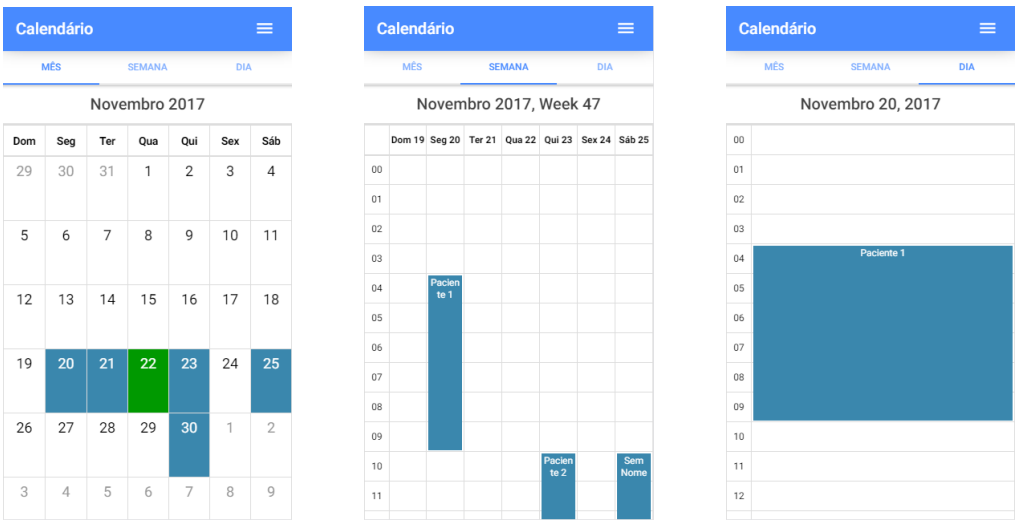
  Por fim, foram feitos testes controlados para verificar a usabilidade da aplicação. A análise da eficiência da aplicação teve base através de um questionário entregue às pessoas que utilizaram a aplicação avaliando pontos como satisfação, facilidade de uso, velocidade de acesso e busca de informações.

4 Arquitetura do Projeto

A arquitetura implementada para o aplicativo de agendamento segue o modelo MVC (Model-View-Controller) que tem como benefício isolar as regras de negócios da lógica de apresentação, possibilitando a existência de várias interfaces com o usuário que podem ser modificadas sem que haja a necessidade de alteração das regras de negócio, tornando a aplicação mais flexível e com oportunidade de reuso das classes. Para isso, o projeto foi construído em três etapas: Construção dos layouts das telas, implementação dos controladores e conexão com o banco de dados em nuvem.

Inicialmente foram desenvolvidos todas as telas da aplicação e seus respectivos fluxos para o produto MVP. Para isso, criou-se uma tela principal onde seria apresentado o calendário como mostrado nas Figura 12, onde o usuário poderia escolher a forma de visualização entre dias do mês, dias da semana ou apenas no dia selecionado.

Figura 12: Calendário.

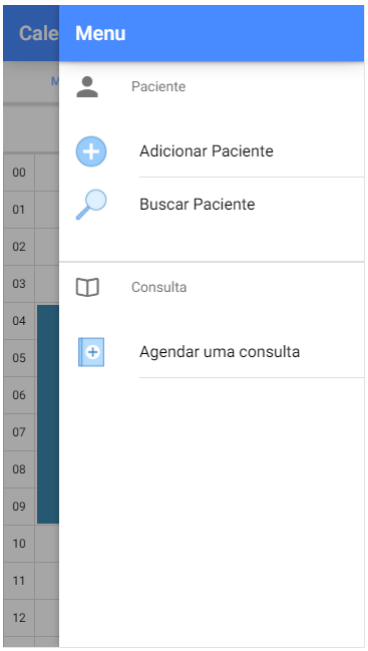


Fonte:Elaborado pelo autor.

Foi desenvolvido também um menu lateral possuindo opções de adicionar um paciente, buscar um paciente e criar uma consulta, como visualizado na Figura 13.

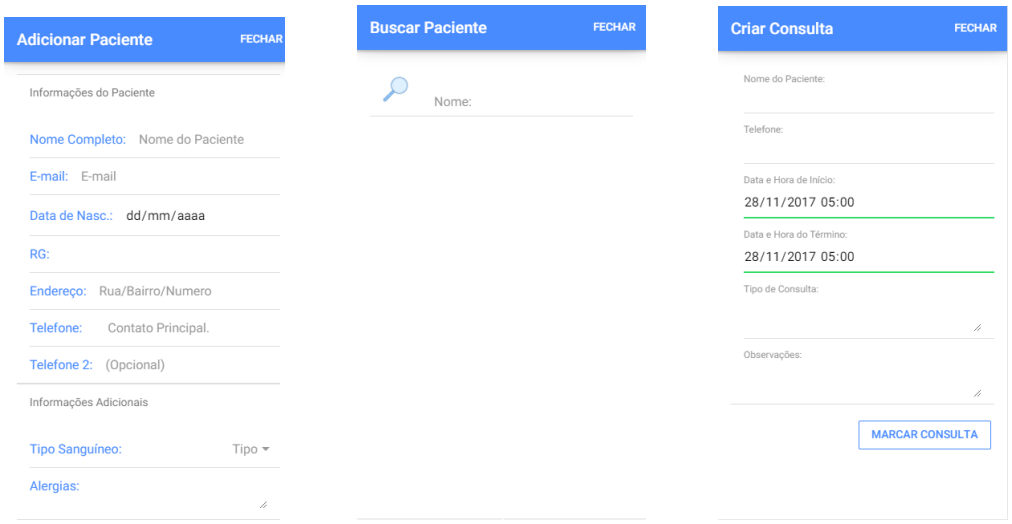
As demais telas da aplicação foram feitas a partir de modais, ou seja, janelas que sobrepõem a tela atual e podem ser fechadas voltando para a visualização em que o modal foi aberto. Dessa forma sempre que o usuário fechar um modal ele voltaria para a página principal, pois é nela que se concentram as informações mais relevantes ao usuário, a do calendário. Os modais desenvolvidos então foram os de adicionar paciente, buscar paciente, agendar uma consulta Figura 14, visualizar uma consulta, perfil do paciente e por fim o histórico de consultas do paciente Figura 15.

Figura 13: Menu Lateral.



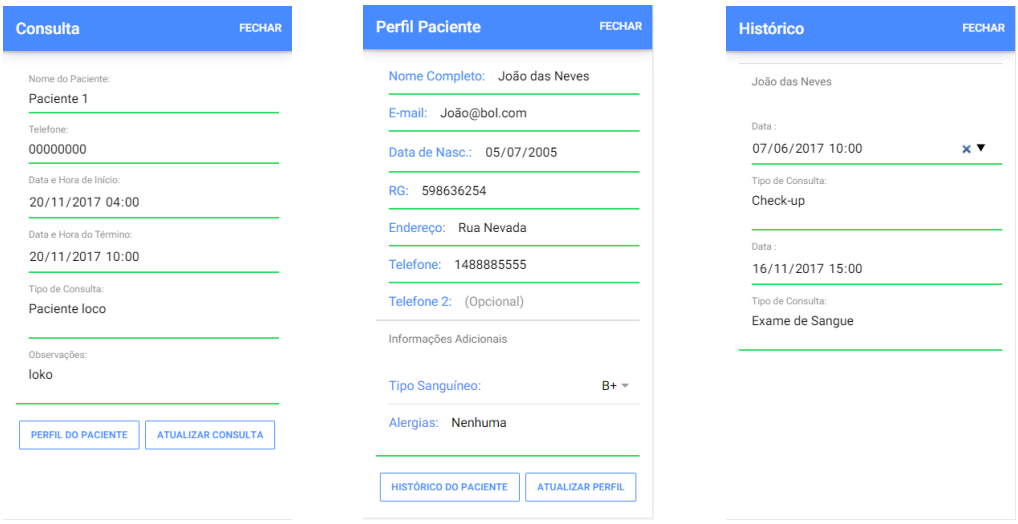
Fonte: Elaborado pelo autor.

Figura 14: Modais: Adicionar Paciente, Buscar Paciente e Agendar Consulta.



Fonte:Elaborado pelo autor.

Figura 15: Modais: Visualizar Consulta, Perfil do Paciente, Histórico.



Fonte:Elaborado pelo autor.

Para poder receber e gerenciar os dados adicionados pelo usuário, como por exemplo no modal de Adicionar Paciente, fora desenvolvidos controladores para poder pegar as entradas de dados, processá-las para então enviar para o banco de dados na nuvem (Figura 16). Os controladores criados também controlam o processo inverso, quando deseja-se visualizar uma consulta, são recebidos dados da nuvem, processados para então disponibilizá-los na visão (View) (Figura 17). Por fim, a configuração do banco de dados em nuvem utilizando o Firebase, para armazenar e tratar as informações enviadas pelo aplicativo. Para o armazenamento, foi utilizado o tipo NoSQL baseado em documentos como visto na Figura 18.

Ao finalizar a aplicação, foram feitos testes controlados para verificar a usabilidade da aplicação. Para isso, foi desenvolvido um questionário utilizando a plataforma TypeForm (TYPEFORM, 2017) que perguntava aos usuários sua satisfação em relação a facilidade de uso, velocidade de acesso, busca de informações e satisfação como mostrado no Anexo A e as respostas no Anexo B.

Figura 16: Controladores.

```
export class AddPacientModalPage {

  user = {} as User;

  //public tipoSangue;
  userRef$: Observable<any[]>;

  constructor(public navCtrl: NavController,
              public navParams: NavParams,
              private view: ViewController,
              private database: AngularFireDatabase) {
    this.userRef$ = this.database.list('pacientes-list').valueChanges();
  }

  ionViewDidLoad() {
    console.log('ionViewDidLoad AddPacientModalPage');
  }


  closeAddPage(){
    this.view.dismiss();
  }

  addPaciente(){
    this.checkIfNull();
    this.database.list('pacientes-list').push({
      name: this.user.name,
      date: this.user.date,
      rg: this.user.rg,
      address: this.user.address,
      phone1: this.user.phone1,
      phone2: this.user.phone2,
      email: this.user.email,
      bloodType: Number(this.user.bloodType),
      alergia: this.user.alergia
    });

    this.user = {} as User;
    this.view.dismiss();
```

Fonte: Elaborado pelo autor.

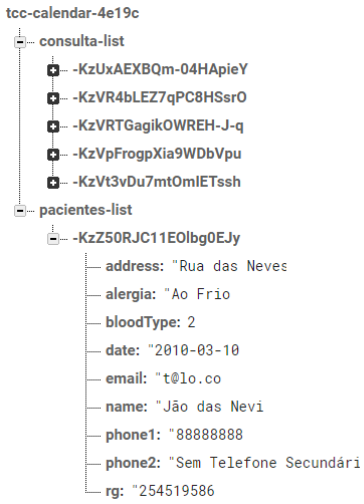
Figura 17: Recebendo dados da nuvem.

```
constructor(public navCtrl: NavController,
  public navParams: NavParams,
  private modal: ModalController,
  private database: AngularFireDatabase) {

  this.database.list<Consulta>('consulta-list').valueChanges().subscribe(_data =>{
    this.consultaList = _data;
  });
}
```

Fonte: Elaborado pelo autor.

Figura 18: Banco de dados NoSQL baseado em documentos



Fonte: Elaborado pelo autor.

5 Resultados

Com os testes controlados, foi possível verificar os seguintes pontos em relação a aplicação: Facilidade de uso, velocidade de acesso, busca de informações e satisfação.

Em relação a facilidade de uso, os usuários acharam o aplicativo simples e com um fluxo agradável, pois com poucas ações era possível obter as informações desejadas ou cadastrar uma consulta e paciente. O fato de ter poucas ações resultou numa velocidade de acesso relativamentes satisfatória pelos usuários, porém houveram alguns feedbacks em relação a velocidade para encontrar um paciente, para encontrar uma forma alternativa além do cadastro de consulta ou busca entre todos os pacientes.

O processo de busca de informações agradou bastante os usuários pois, em relação às datas de consulta, era prático de encontrar no calendário uma data e um horário que possuía uma consulta, e ao encontrar bastava apenas clicar para visualizar as informações. Uma das coisas que agradou os usuários foi do usuário ao abrir a aplicação, já visualizar o que acontece no dia atual, uma vez que isso já retirava o trabalho de buscar uma agenda física, procurar o dia para ver as pendências do dia, pois bastava abrir a aplicação para poder obter as informações desejadas.

Por fim, os usuários encontraram-se satisfeitos com o aplicativo, uma vez que era simples de se utilizar, havia as principais características necessárias para salvar uma consulta, não armazena localmente as informações podendo assim ter apenas um dispositivo conectado a internet simples para visualizar as informações, e pela facilidade de buscar e visualizar em qualquer lugar as informações desejadas apenas utilizando um dispositivo móvel.

6 Conclusão

O presente trabalho apresenta uma aplicação móvel para agendamento de consultas médicas utilizando Computação em Nuvem como forma de armazenamento de dados. O desempenho e satisfação de uso da aplicação foi analisado através de um teste controlado coletando feedbacks de usuários.

Os testes mostraram uma grande satisfação em relação a utilização da Computação em Nuvem, pois isso retirava o trabalho do usuário possuir um computador local, fichas para cadastro e histórico do respectivo paciente. Apenas utilizando um dispositivo móvel conectado a internet era possível visualizar o que desejava. Também houve uma grande satisfação em relação à facilidade de acesso e a diminuição de gasto de tempo ao se buscar uma determinada informação.

Portanto, a aplicação da Computação em Nuvem em uma aplicação móvel possibilitou que os serviços ficassem acessíveis ao usuário de maneira transparente, de modo que ele não precisou se preocupar com a infraestrutura do sistema e sim, apenas com a utilização da aplicação.

6.1 Trabalhos futuros

A partir dos feedbacks, pode-se considerar analisar e melhorar o fluxo do aplicativo, busca de pacientes, uma forma mais rápida de se encontrar uma informação de um determinado paciente e a possibilidade de incluir arquivos como exames de sangue ou raio X.

Referências

ABOULNAGA, A. et al. Deploying database appliances in the cloud. *IEEE Data Eng. Bull.*, 2009. Citado na página 17.

ANDRADE, A. et al. AdoÇÃo de sistemas de armazenamento de dados na nuvem: Um estudos com usuÁrios finais. *Revista de Administração e Inovação, São Paulo*, 2015. Citado 2 vezes nas páginas 12 e 16.

ANGULAR. 2017. Disponível em: <https://angular.io/>. Acesso em: 20 Agosto 2017. Citado na página 28.

ARUTYNOV, V. V. Cloud computing: its history of development, modern state, and future considerations. *Scientific and Technical Information Processing*, 2012. Citado na página 12.

AWS. 2017a. Disponível em: <https://aws.amazon.com/pt/?nc2=h_lg>. Acesso em: 24 Abril 2017. Citado na página 29.

AWS. 2017b. Disponível em: <https://aws.amazon.com/pt/nosql/>. Acesso em: 15 Novembro 2017. Citado 3 vezes nas páginas 9, 19 e 23.

AZURE. 2017. Disponível em: <https://azure.microsoft.com/pt-br/overview/what-is-azure-/>. Acesso em: 24 Abril 2017. Citado 2 vezes nas páginas 18 e 29.

BANDYOPADHYAY, S. *Cloud computing: the business perspective*. 2009. Disponível em: <http://papers.ssrn.com/sol3/papers.cfm?abstract_id=1413545>. Acesso em: 09 Novembro 2017. Citado na página 12.

BRANTNER, M. et al. Building a database on s3. . *In Proceedings of the 2008 ACM SIGMOD international conference on Management of data - SIGMOD '08*, 2008. Citado na página 15.

BUYYA, R. et al. Cloud computing and emerging it platforms: Vision, hype, and reality for delivering computing as the 5th utility. *Future Gener. Comput. Syst.*, 2009. Citado na página 15.

CANTU, A. *The History and Future of Cloud Computing*. 2011. Disponível em: <https://www.forbes.com/sites/dell/2011/12/20/the-history-and-future-of-cloud-computing/5c245d80227b>. Acesso em: 20 Abril 2017. Citado na página 12.

CARLOS, W. *O que é IONIC?* 2016. Disponível em: <http://comocriaraplicativos.com-.br/o-que-e-o-ionic/>. Acesso em: 20 Agosto 2017. Citado 2 vezes nas páginas 22 e 29.

CATTELL, R. *Scalable SQL and NoSQL DataStores.* 2011. Disponível em: <http://citeseerx-.ist.psu.edu/viewdoc/download?doi=10.1.1.692.2621rep=rep1type=pdf>. Acesso em: 16 Novembro 2017. Citado 2 vezes nas páginas 19 e 23.

CHANG, F. et al. Bigtable: A distributed storage system for structured data. *In Proceedings of the 7th USENIX Symposium on Operating Systems Design and Implementation (OSDI'06), volume 26*, 2006. Citado na página 19.

CLOUDSTACK. 2017. Disponível em: <https://cloudstack.apache.org/>. Acesso em: 24 Abril 2017. Citado 2 vezes nas páginas 17 e 29.

COUTINHO, E. et al. Elasticidade em computação em nuvem: Uma abordagem sistemática. *31º Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos – SBRC*, 2013. Citado na página 16.

DEVMEDIA. 2017. Disponível em: <https://www.devmedia.com.br/guia/ionic/38372>. Acesso em: 20 Agosto 2017. Citado na página 26.

DILLON, T.; WU, C.; CHANG, E. Cloud computing: Issues and challenges. *2010 24th IEEE International Conference on Advanced Information Networking and Applications*, 2010. Citado na página 13.

FIREBASE. 2017a. Disponível em: <https://firebase.google.com/products/?authuser=0>. Acesso em: 11 Novembro 2017. Citado 3 vezes nas páginas 18, 24 e 29.

FIREBASE. 2017b. Disponível em: <https://firebase.google.com/products/database-/?authuser=0>. Acesso em: 11 Novembro 2017. Citado na página 25.

FIREBASE. 2017c. Disponível em: <https://firebase.google.com/products/auth-/?authuser=0>. Acesso em: 11 Novembro 2017. Citado na página 25.

FIREBASE. 2017d. Disponível em: <https://firebase.google.com/products/storage-/?authuser=0>. Acesso em: 11 Novembro 2017. Citado na página 25.

GOOGLEDOCS. 2017. Disponível em: <https://www.google.com/docs/about/>. Acesso em: 11 Novembro 2017. Citado na página 18.

HONEY, J. *Armazenamento - no local ou na nuvem?* 2015. Disponível em: <https:/-/canaltech.com.br/infra/armazenamento-no-local-ou-na-nuvem-49707/>. Acesso em: 15 Outubro 2017. Citado na página 13.

INTEL. *Utility Computing*. 2017. Disponível em: <http://meekho.weebly.com/uploads/5/0-/0/3/5003820/utility_computing.pdf>. Acesso em: 15 Novembro 2017. Citado na página 15.

IONIC. *Ionic 3.0 has Arrived!* 2017. Disponível em: <http://blog.ionicframework.com/ionic-3-0-has-arrived/>. Acesso em: 20 Agosto 2017. Citado na página 26.

JACOBS, D.; AUBACH, S. Ruminations on multi-tenant databases. *Aachen Germany: Technische Universität München*, 2007. Citado na página 17.

JOSÉ, E. *Introdução ao TypeScript*. 2017. Disponível em: <https://www.devmedia.com-.br/introducao-ao-typescript/36729>. Acesso em: 20 Agosto 2017. Citado na página 27.

LINO, M. *O que é Apache Cordova?* 2015. Disponível em: <http://blog.marianalino-.com.br/o-que-e-apache-cordova/>. Acesso em: 20 Agosto 2017. Citado na página 28.

NODE. 2017. Disponível em: <https://nodejs.org/en/about/>. Acesso em: 11 Novembro 2017. Citado na página 25.

NODEBR. 2017. Disponível em: <http://nodebr.com/o-que-e-a-npm-do-nodejs/>. Acesso em: 11 Novembro 2017. Citado na página 25.

OCEAN, D. 2017. Disponível em: <https://www.digitalocean.com/products/storage/>. Acesso em: 24 Abril 2017. Citado 2 vezes nas páginas 17 e 29.

OLIVEIRA, S. Banco de dados não-relacionais: Um novo paradigma para armazenamento de dados em sistemas de ensino colaborativo. *Revista Eletrônica da Escola de Administração Pública do Amapá.*, 2014. Citado na página 21.

OPUS. 2014. Disponível em: <https://www.opus-software.com.br/o-que-e-cloud-computing-/>. Acesso em: 15 Novembro 2017. Citado na página 15.

PAYPAL. 2017. Disponível em: <https://www.paypal.com/br/home>. Acesso em: 22 Abril 2017. Citado na página 13.

RUSCHEL, H.; ZANOTTO, M.; MOTA, W. Especialização em redes e segurança de sistemas. *Pontifícia Universidade Católica do Paraná. Curitiba*, 2010. Citado na página 15.

SCHMITZ, D. *Diga olá ao TypeScript e adeus ao JavaScript*. 2015. Disponível em: <https://tableless.com.br/diga-ola-ao-typescript-e-adeus-ao-javascript/>. Acesso em: 20 de Agosto 2017. Citado na página 27.

SHAMIM, S.; SARKER, A.; BAHAR, N. International journal of computer applications. volume 113 – no. 16. *A Review on Mobile Cloud Computing.*, 2015. Citado na página 18.

SILVA, D.; VICENTE, G. *Apresentando o AngularJS*. 2014. Disponível em: <http://dextra-.com.br/pt/blog/apresentando-o-angular-js-4/>. Acesso em: 20 de Agosto 2017. Citado na página 27.

SOROR, A. A. et al. *Automatic virtual machine configuration for database workloads.* [S.l.]: ACM Trans., 2010. Citado na página 15.

SOUSA, F.; MOREIRA, L.; MACHADO, J. Computação em nuvem: Conceitos, tecnologias, aplicações e desafios. *Universidade Federal do Ceará*, 2015. Citado na página 16.

STACKOVERFLOW. 2017. Disponível em: <https://pt.stackoverflow.com/questions-/236145/qual-%C3%A9-a-diferen%C3%A7a-entre-o-apache-cordova-e-o-ionic>. Acesso em: 20 Agosto 2017. Citado na página 28.

TANENBAUM, A.; STEEN, M. *Sistemas Distribuídos: Princípios e paradigmas*. [S.l.]: Ed.Pearson, 2007. Citado na página 12.

TAURION, C. *Cloud Computing: computação em nuvem: transformando o mundo da tecnologia da informação*. [S.l.]: Editora Brasport, 2009. Citado na página 12.

TYPEFORM. 2017. Disponível em: <https://www.typeform.com/>. Acesso em: 20 Novembro 2017. Citado na página 32.

VANDRESEN, R.; MAGALHÃES, W. Conceitos e aplicaÇÕes da computaÇÃo em nuvem. *Paranavaí: Universidade Paranaense*, 2013. Citado na página 13.

VASCONCELOS, L. *Apps Híbridas com Cordova e Ionic*. 2017. Disponível em: <https://medium.com/@lfv89/nessa-s%C3%A9rie-divida-em-3-partes-vou-falar-um-pouco-mais-a-fundo-sobre-desenvolvimento-h%C3%ADbrido-914f22453c83>. Acesso em: Acesso em 16 Novembro 2017. Citado na página 22.

WEN, Y. et al. Cloud mobile media: Reflections and outlook. *IEEE TRANSACTIONS ON MULTIMEDIA, VOL. 16, NO. 4*, 2014. Citado na página 13.

7 Anexo A - Formulário de Satisfação

Link do TypeForm utilizado para o formulário de satisfação:

https://hysumi.typeform.com/to/kyHZkh

O formulário colocado no TypeForm segue o modelo:

Nome Completo:

Cidade / Estado:

Facilidade de Uso.

Facilidade em criar uma consulta.

Facilidade em buscar um paciente..

Velocidade de acesso as informações de consulta.

Velocidade de acesso as informações do paciente.

Satisfação em poder acessar as informações de qualquer dispositivo móvel.

Satisfação em saber que o aplicativo não precisa gravar informações no celular.

Opinião e feedbacks:

8 Anexo B - Resultado da Pesquisa de Satisfação

Nome Completo	**Cidade / Estado**	**Facilidade de uso.**
Thais Freire Wu	Campinas / SP	6
Kim Watanabe Hayakawa	Campinas - SP	9
Daniela Luizi Garcia	Campinas SP	10
Rhuan Benedito Esborini	Bauru / SP	8
Wagner Ferreira dos Santos	Bauru / SP	10
Vinicius Figueiredo Rodrigues	Bauru/SP	7
Thais Maria Alves Pereira	Bauru/SP	8
Gabriel Pereira de Paula	Bauru/SP	9
Thomas Henrique Honda	Bauru - SP	9
Karoline Setoue	Bauru/SP	9
Raissa Rodrigues de Souza	Piracicaba-SP	10
Luis Fernando	São Paulo	9
Jairo Santos	São Paulo	10
Wesley Franco Ferreira	Campo mourão/Paraná	8

Facilidade em criar uma consulta.	**Facilidade em buscar um paciente.**	**Velocidade de acesso as informações de consulta.**
8	4	6
9	7	10
10	7	10
8	9	9
10	10	10
8	5	6
7	5	8
10	8	9
10	7	9
9	10	10
10	9	10
10	10	10
10	9	9
8	9	10

Velocidade de acesso as informações do paciente.	**Satisfação em poder acessar as informações de qualquer dispositivo móvel.**	**Satisfação em saber que o aplicativo não precisa gravar informações no celular.**
4	9	8
4	10	10
6	10	10
9	10	10
10	10	10
4	9	10
7	10	10
9	9	10
10	10	8
9	9	10
5	10	10
8	10	10
9	10	10
10	10	10

Opinião e feedbacks.	**Submit Date (UTC)**
Achei o protótipo muito interessante, acredito que possa ser implementado na realidade dos profissionais com mais refinamento.	2017-11-18 12:01:05
Muito prático agendar uma consulta.	2017-11-18 13:00:53
A melhor parte é não ter que me preocupar em salvar nada! Adorei	2017-11-19 14:04:19
	2017-11-20 15:08:24
muito útil	2017-11-20 15:16:04
Um bom app, porém precisa melhorar o fluxo	2017-11-21 12:09:19
Algumas informações, pela disposição, deixou confuso seu manuseio, contudo é eficiente e satisfaz a necessidade. Precisa de uma disposição e organização das informações facilitando seu uso.	2017-11-21 13:05:46
Excelente aplicativo, simples e bem estruturado. Não poder ver um histórico com todas as consultas do paciente é algo que poderia ser implementado, mas a experiência de uso do aplicativo e utilizar de tecnologia nuvem para o banco de dados é algo que eu inclusive gostaria de ver feito em outros aplicativos com mais frequência.	2017-11-21 13:20:40
Acho interessante adicionar uma sessão para dar upload em exames de sangue, raio-x, ou qualquer exame médico que seja interessante adicionar ao histórico do paciente. Caso o mesmo não tenha o domínio técnico para fazer tal processo, os laboratórios poderiam ter a opção de enviar direto para o perfil do paciente.	2017-11-21 13:51:06
Acho que o aplicativo facilita muito na hora de marcar consultas, proporcionando uma facilidade muito grande de poder fazer tudo diretamente do celular.	2017-11-21 14:16:20
	2017-11-21 18:46:07
uso prático e fácil	2017-11-22 11:38:02
	2017-11-22 11:49:32
Acho que deveria adicionar mais opções de restrições médicas e até um Histórico clinico.	2017-11-22 12:04:32