

Universidade Federal do Triângulo Mineiro

Bruno Ribeiro e Lima

**Desenvolvimento de uma aplicação serverless para estudo
utilizando mobile learning e gamification**

Uberaba

2020

Bruno Ribeiro e Lima

**Desenvolvimento de uma aplicação serverless para estudo
utilizando mobile learning e gamification**

Dissertação apresentada ao Programa de Mestrado Profissional em Inovação Tecnológica da Universidade Federal do Triângulo Mineiro, como requisito para obtenção do título de Mestre em Inovação Tecnológica.

Orientadora: Profa. Dra. Ariana de Campos

Uberaba

2020

**Catálogo na fonte: Biblioteca da Universidade Federal do
Triângulo Mineiro**

L696d Lima, Bruno Ribeiro e
Desenvolvimento de uma aplicação *serverless* para estudo
utilizando *mobile learning* e *gamification* / Bruno Ribeiro e Lima. --
2020.
90 f. : il., graf., tab.

Dissertação (Mestrado Profissional em Inovação Tecnológica) -
- Universidade Federal do Triângulo Mineiro, Uberaba, MG, 2020
Orientadora: Profa. Dra. Ariana de Campos

1. Gamificação. 2. Sistemas de comunicação móvel. 3. Compu-
tação em nuvem. 4. *Microservices*. I. Campos, Ariana de.
II. Universidade Federal do Triângulo Mineiro. III. Título.

CDU 004:37.091.64

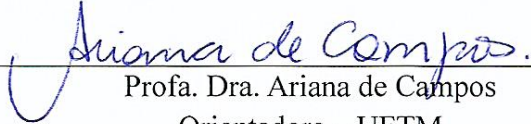
BRUNO RIBEIRO E LIMA

DESENVOLVIMENTO DE UMA APLICAÇÃO SERVERLESS PARA ESTUDO
UTILIZANDO MOBILE LEARNING E GAMIFICATION

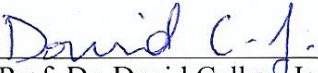
Trabalho de conclusão apresentado ao
Programa de Mestrado Profissional em
Inovação Tecnológica da Universidade Federal
do Triângulo Mineiro, como requisito para
obtenção do título de mestre.

Uberaba, 20 de fevereiro de 2020

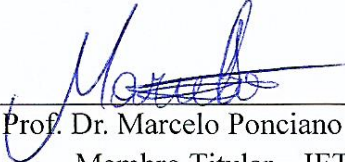
Banca Examinadora:



Profa. Dra. Ariana de Campos
Orientadora – UFTM



Prof. Dr. David Calhau Jorge
Membro titular – UFTM



Prof. Dr. Marcelo Ponciano da Silva
Membro Titular – IFTM

Dedico este trabalho a minha noiva Tamara pela compreensão e apoio em todos momentos e a minha família que sempre contribuiu para o meu crescimento pessoal e profissional.

AGRADECIMENTOS

A Universidade Federal do Triângulo Mineiro pela oportunidade de realização do mestrado, uma grande chance de aprendizado.

Aos professores do PMPIT, por oferecer um ensino de qualidade e suporte sempre que precisei.

A professora Ariana de Campos pelas orientações com dicas precisas e valiosas que ajudaram a tornar este trabalho ainda melhor.

A Capes, por disponibilizar um portal de periódicos rico em conteúdo para pesquisas.

Aos meus pais e familiares que sempre me apoiaram nas minhas escolhas.

A minha noiva Tamara pela paciência e amor incondicional em todos momentos.

Aos companheiros de trabalho pela paciência e compreensão.

Agradeço a todos que, direta ou indiretamente, contribuíram para realização deste trabalho.

“A inovação sempre significa um risco. Qualquer atividade econômica é de alto risco e não inovar é muito mais arriscado do que construir o futuro. ”

Peter Drucker

RESUMO

Com a chegada dos dispositivos móveis, tem surgido novas possibilidades que facilitam as tarefas realizadas no cotidiano, inclusive no processo de ensino e aprendizagem. O *mobile learning* e a *gamification* têm se mostrado como recursos eficazes neste processo. Porém, algumas instituições de ensino ainda não usufruem desses benefícios. Neste contexto, com o intuito de colaborar no processo de ensino e aprendizado, o presente trabalho tem como objetivo desenvolver um ambiente que permita o aluno estudar através de jogos educativos em dispositivos móveis. O conteúdo abordado nesta aplicação foi inteligência artificial, por se tratar de um tema que tem se destacado na tecnologia. Para alcançar esse objetivo foi desenvolvido um aplicativo Android, uma aplicação *web* para usuários que preferem usufruir desses recursos sem a necessidade de instalação em seu dispositivo e um *web service* que irá responder as requisições dessas aplicações. Foi adotado a utilização de infraestrutura na nuvem para adequar os requisitos do governo para novas aplicações em instituições públicas e assim, ser possível usufruir de recursos presentes neste ambiente. O desenvolvimento da aplicação foi feito utilizando tecnologias em evidências no mercado, e ao término do trabalho, foi analisado a performance e qualidade da aplicação, conseguindo obter 99 de 100 pontos possíveis no site *PageSpeed Insight*, do Google.

Palavras-Chaves: Gamification. Mobile Learning. Microservices. Serverless.

ABSTRACT

With the arrival of mobile devices, new possibilities have emerged that facilitate the tasks performed in everyday life, including the process of teaching and learning. Mobile learning and gamification have proven to be effective resources in this process. However, some educational institutions still do not enjoy these benefits. In this context, in order to collaborate in the teaching and learning process, this work aims to develop an environment that allows the student to study through educational games on mobile devices. The content addressed in this application was artificial intelligence, as it is a topic that has stood out in technology. To achieve this goal an Android application was developed, a web application for users who prefer to enjoy these resources without the need for installation on your device and a web service that will answer the requests of these applications. It was adopted the use of cloud computing to adapt the government requirements for new applications in public institutions and thus be able to take advantage of resources present in this environment. The development of the application was carried out using technologies in evidence in the market and, at the end of the work, the application's performance and quality were analyzed, obtaining 99 out of 100 possible points on the Google PageSpeed Insight website.

Keywords: Gamification. Mobile Learning. Microservices. Serverless.

LISTA DE ILUSTRAÇÕES

Figura 1 - Uso dos sistemas operacionais móveis no mundo	29
Figura 2 - REST vs. SOAP	33
Figura 3 - Arquitetura tradicional de uma aplicação monolítica	37
Figura 4 - Exemplo de uma arquitetura de <i>microservices</i>	38
Figura 5 - Serviços da AWS utilizados	45
Figura 6 - Gráfico de Custo/Demanda de Servidores.....	47
Figura 7 - VSCode.....	49
Figura 8 - Fluxograma da aplicação do cliente.....	54
Figura 9 - Publicação no Google Play.....	55
Figura 10 - Aplicação em diferentes aparelhos	56
Figura 11 - Tela inicial sem autenticar.....	58
Figura 12 - Tela de cadastro	59
Figura 13 - Tela de autenticação.....	60
Figura 14 - Tela inicial restrita	61
Figura 15 - Opções de estudo.....	62
Figura 16 - Tela do modo FlashCard.....	63
Figura 17 - Tela exibindo a resposta do FlashCard.....	64
Figura 18 - Tela no modo Perguntas e Respostas	65
Figura 19 - Tela no modo Verdadeiro ou Falso	66
Figura 20 - Tela de resultado	67
Figura 21 - Exibindo perguntas e respostas do sumário	68
Figura 22 - Tela do Ranking.....	69

Figura 23 - Gráfico com a média do tempo de execução (ms)	72
Figura 24 - Desempenho no PageSpeed Insights	73

LISTA DE QUADROS

Quadro 1 – Distribuição das versões do Android	31
Quadro 2 – Cartão de história: Cadastro de novo usuário	85
Quadro 3 – Cartão de história: Login de usuário já cadastrado	85
Quadro 4 – Cartão de história: Selecionar o método de estudo	86
Quadro 5 – Cartão de história: Estudar com FlashCards	86
Quadro 6 – Cartão de história: Estudar com Perguntas e Respostas	87
Quadro 7 – Cartão de história: Estudar com Verdadeiro ou Falso	87
Quadro 8 – Cartão de história: Exibir Resultado	88
Quadro 9 – Cartão de história: Exibir Ranking	88

LISTA DE TABELAS

Tabela 1 - Operações do protocolo HTTP para manipulação de dados	34
Tabela 2 - Tempo de resposta ao executar lógica (ms)	71

LISTA DE SIGLAS

AWS – Amazon Web Services

API – Application Programming Interface

CSS – Cascading Style Sheets

ERP – Enterprise resource planning

GPS – Global Positioning System

IA – Inteligência Artificial

IBGE – Instituto Brasileiro de Geografia e Estatística

IFTM – Instituto Federal do Triângulo Mineiro

iOS – iPhone Operational System

JS – JavaScript

JSON – JavaScript Object Notation

HTTP – Hypertext Transfer Protocol

HTML – Hypertext Markup Language

MOODLE – Modular Object-Oriented Dynamic Learning Environment

NOSQL – Not Only SQL

PSI – PageSpeed Insights

PWA – Progressive Web Apps

RAM – Random Access Memory

REST – Representational State Transfer

SDK – Software Development Kit

SOAP – Simple Object Access Protocol

SQL – Structured Query Language

SVN – Subversion

TI – Tecnologia da Informação

UFTM – Universidade Federal do Triângulo Mineiro

URI – Uniform Resource Identifier

URL – Uniform Resource Locator

VSCODE – Visual Studio Code

XML – eXtensible Markup Language

WSDL – Web Service Definition Language

SUMÁRIO

1.INTRODUÇÃO	17
2. OBJETIVOS	19
2.1 OBJETIVOS ESPECÍFICOS	19
3. REVISÃO BIBLIOGRÁFICA.....	20
3.1 GERAÇÕES E SEUS COMPORTAMENTOS	20
3.2 MOBILE LEARNING.....	22
3.3 GAMIFICATION.....	23
3.4 PLATAFORMAS ADAPTATIVAS.....	26
3.5 SISTEMAS OPERACIONAIS PARA DISPOSITIVOS MÓVEIS.....	28
3.6 WEB SERVICES	32
3.6.1 REST.....	33
3.6.2 GraphQL.....	35
3.7 PADRÕES DE ARQUITETURAS DE SOFTWARE	36
3.7.1 Arquitetura Monolítica.....	36
3.7.2 Arquitetura de <i>Microservices</i>	37
3.8 HOSPEDAGEM NA NUVEM USANDO <i>SERVERLESS</i>	39
3.8.1 Serverless	40
4. MATERIAIS E MÉTODOS.....	43
4.1 TECNOLOGIAS.....	43
4.1.1 React.....	43
4.1.2 Node.js	44
4.1.3 Python	44
4.2 IMPLANTAÇÃO NA AMAZON WEB SERVICES (AWS).....	45
4.3 PAGESPEED INSIGHT	48
4.4 AMBIENTE E RECURSOS.....	48
4.5 ESPECIFICAÇÕES PARA O DESENVOLVIMENTO.....	51
4.6 ENGENHARIA DE REQUISITOS	51
4.6.1 Requisitos Funcionais	52
4.6.2 Requisitos Não Funcionais	52
4.7 FLUXOGRAMA	53
5. RESULTADOS E DISCUSSÃO.....	55

5.1 DISPONIBILIZAÇÃO DAS APLICAÇÕES	55
5.1.1 Publicação no Google Play	55
5.1.2 Aplicação para internet	56
5.2 APLICAÇÃO DESENVOLVIDA.....	57
5.2.1 Tela inicial	57
5.2.2 Tela de cadastro	58
5.2.3 Tela de autenticação	59
5.2.4 Tela inicial restrita	60
5.2.5 Opções de estudo	61
5.2.6 Tela do modo FlashCard	62
5.2.7 Tela no modo Perguntas e Respostas	65
5.2.8 Tela no modo Verdadeiro ou Falso	66
5.2.9 Tela de resultado	67
5.2.10 Tela exibindo o ranking	68
5.3 SERVIDOR DAS APLICAÇÕES.....	70
5.4 DESEMPENHO	72
6. CONCLUSÃO	75
REFERÊNCIAS.....	77
GLOSSÁRIO	83
APÊNDICE A – CARTÕES DE HISTÓRIA	85
APENDICE B – CONFIGURAÇÃO DO MANIFEST.XML	90

1.INTRODUÇÃO

Com a chegada de novas ferramentas tecnológicas, os hábitos das pessoas têm mudado. Um dos aparelhos que surgiram e faz parte do cotidiano de mais da metade da população mundial, é o celular (LECHETA, 2013). Esse aparelho evoluiu, transformando-o em *smartphone*, adquirindo cada vez mais capacidades computacionais e ainda tem a vantagem de estar na maior parte do tempo próximo do proprietário.

Por conta dos recursos e da flexibilidade dos aparelhos móveis, pesquisadores foram atraídos para explorar essa tecnologia e investigar os seus impactos em estudantes e educadores. Diversas instituições têm adotado o *mobile learning* (aprendizado móvel) como apoio ao ensino (AL-EMRAN; ELSHERIF; SHAALAN, 2015). Junto ao *mobile learning*, outro recurso que tem se destacado no aprendizado é o *gamification*. Pesquisas mostram que a utilização de jogos educativos como quiz (CREMONTTI, 2016; VARGAS, 2017) e o uso de *flashcards* (HART-MATYAS *et al.*, 2019) têm conseguido resultados positivos e boa aceitação entre professores e alunos. Porém, esses artifícios de ensino, utilizando jogos educativos, ainda não são práticas comuns em algumas instituições de ensino, como no Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro (IFTM).

O IFTM é uma instituição de ensino pública, que contém 9163 matrículas em cursos em diferentes níveis, que vão desde o ensino médio até a pós-graduação. Possui uma reitoria e nove câmpus distribuídos por Minas Gerais (principalmente no Triângulo Mineiro), contando com 1121 servidores, incluindo docentes e técnicos administrativos (IFTM, 2020). Esse instituto possui várias soluções administrativas em seu ERP (*Enterprise Resource Planning*), mas necessita de um ambiente de estudo com essas tecnologias. Além do ERP, a instituição possui um aplicativo para os alunos de consultas às notas e frequências, mas para o estudo se restringe a um disco virtual com documentos inseridos pelos professores.

Com o intuito de contornar a limitação de instituições que não possuem esses artefatos como apoio no ensino e aprendizagem, o estudo propõe desenvolver um aplicativo para Android e uma aplicação *web* utilizando os benefícios do *mobile learning* junto ao *gamification*, usando tecnologias em evidência no desenvolvimento

de *software*. A proposta, é ter mais um apoio para o aprendizado em instituições de ensino, pensando em especial no IFTM, entidade que o autor deste trabalho foi estudante de curso técnico e superior e sentia falta de sistemas que apoiavam diretamente o aprendizado. Atualmente, o autor trabalha nesta instituição e é responsável pela criação e manutenção de diversos sistemas, porém, nenhum desses auxiliam o discente a estudar usando conceitos de *gamification*, no qual poderia contribuir para manter o aluno mais envolvido durante o aprendizado.

Para o desenvolvimento de uma aplicação que pode ser utilizada em instituições públicas, é preciso seguir legislações como a instrução normativa número 1 de 4 de abril de 2019 (BRASIL, 2019) que requisita dar prioridade a contratação de infraestrutura que usa computação na nuvem. Devido a esse fato, e a importância de ser criterioso na escolha das opções tecnológicas utilizadas, foi realizado uma consulta na literatura e em sites especializados em tecnologia para conhecer o que está em evidência em sistemas que utilizam computação em nuvem, e, o que é promissor no desenvolvimento de *software*.

O público alvo das aplicações desenvolvidas durante esse trabalho são pessoas com interesse em usar dispositivos móveis durante o aprendizado e instituições que pretendem desenvolver aplicações para o ensino. Com essas informações, espera-se atingir principalmente alunos que concluíram o ensino médio e têm interesse em tecnologia. O conteúdo tomado como exemplo de estudo para as aplicações criadas, foi inteligência artificial por se tratar de um assunto em alta nos últimos anos e devido ao fato de o conceito envolvido nesse assunto ser considerado muito teórico (LEMOS; FERNANDES, 2015). Apesar do assunto da aplicação desenvolvida ter sido explorado apenas inteligência artificial, através de alteração no cadastro de questões é possível expandir para ensinar diferentes tipos de conteúdo, conseguindo assim servir de apoio no ensino de diversas matérias distintas, além do que foi utilizado nesta dissertação.

Com as tecnologias presentes neste trabalho, é esperado obter uma boa performance durante o uso das aplicações devido a arquitetura e as tecnologias utilizadas durante o desenvolvimento. Após a criação, as aplicações foram testadas no PageSpeed Insights para avaliar a qualidade do sistema desenvolvido e verificar se o desempenho da aplicação foi satisfatório.

2. OBJETIVOS

O projeto visa a criação de um ambiente de estudos que apoie o ensino e aprendizado por meio de dispositivos móveis, utilizando recursos presentes na *gamification*, fazendo o uso de tecnologias em evidência no desenvolvimento de *software* e com uma arquitetura de *software* robusta e implantada em uma infraestrutura de serviços na nuvem.

2.1 OBJETIVOS ESPECÍFICOS

- Criar uma aplicação *web* que permita os alunos estudarem através da internet;
- Criar um aplicativo Android e disponibilizar no Google Play;
- Desenvolver *microservices* para responder as requisições das aplicações;
- Desenvolver ambiente para estudo usando *FlashCards*;
- Desenvolver o modo de estudo Pergunta e Respostas;
- Desenvolver o modo de estudo Verdadeiro ou Falso e
- Criar um *ranking* de pontos dos estudantes.

3. REVISÃO BIBLIOGRÁFICA

Nesta seção serão abordadas informações sobre tecnologias que surgiram, ou que auxiliam na construção de novos artefatos tecnológicos que podem servir como apoio no processo de aprendizagem, assim como é possível obter os benefícios do uso desses equipamentos em conjunto a sistemas educativos que apresentam características encontradas em jogos.

3.1 GERAÇÕES E SEUS COMPORTAMENTOS

Conhecer as características, assim como as tendências, costumes e influências de cada período de tempo, ajuda na relação social entre as pessoas. O comportamento de cada indivíduo pode estar ligado a fatores culturais e regionais, entretanto muitas pesquisas são realizadas separando grupos sociais em gerações, levando em consideração o ano de nascimento do sujeito envolvido. Cada geração é formada por crianças nascidas em um intervalo de datas, que anteriormente era de aproximadamente 20 anos, porém com a rapidez das mudanças que tem acontecido atualmente, o intervalo entre as gerações mais novas estão sendo representados por períodos de dez anos (CORDONI, 2016).

Os indivíduos da chamada geração Z são constituídos por pessoas que nasceram após os anos 2000 e antes de 2010, período que apareceu a Web 2.0, que proporcionou um ambiente virtual mais dinâmico e colaborativo, foi nesta época também que surgiu e popularizou novos aparelhos e ferramentas digitais (INDALÉCIO; RIBEIRO, 2017) tais como *tablets*, iPods e redes sociais (CORDONI, 2016). A geração Z já surgiu em um universo altamente tecnológico, marcado pelo aparecimento de tecnologias disruptivas, como os carros autônomos, nanotecnologia e a computação em nuvem. Os indivíduos dessa geração têm uma dependência da internet maior que a passada, possuem menos intimidade física, permanecem constantemente online, apreciam manifestar suas desigualdades e ainda detêm um grande senso de humanitarismo e responsabilidade social (SOLIS, 2018).

Após o período da geração Z, iniciou a chamada geração Alpha, formada pelas crianças nascidas após o ano 2010. A geração Alpha teve seu nome usado pela

primeira vez em março de 2010 pelo sociólogo australiano Mark McCrindle, fazendo referência a primeira letra do alfabeto grego (α) (FRANQUIA, 2015). A principal diferença entre essa geração em relação com a anterior, é que desde seus primeiros anos de vida essas crianças já se interagem com a tecnologia (LINS, 2018). Também conhecida como geração M (Mobile), os indivíduos desse período estão em um ambiente onde as novas tecnologias de comunicação estão bem desenvolvidas e difundidas na sociedade (SOLIS, 2018). As pessoas que nasceram no período da geração mais nova do século 21, podem ter o nível de aprendizado mais elevado que as gerações anteriores, já que essas crianças vão iniciar os estudos mais novos e irão ser a primeira geração a vivenciar um novo sistema escolar, adaptado ao estudante, híbrido (on-line e off-line) e que tem como característica a independência do aluno e no seu conhecimento, ainda possuindo a seu favor os avanços da tecnologia da sua época (FRANQUIA, 2015).

Smartphones, tablets, videogames interativos, computadores cada vez mais rápidos, iPods, televisores gigantes em alta definição, redes sociais, mensagem de texto e de voz e selfie fazem parte da rotina dos jovens Z e das crianças Alpha, enquanto para as outras gerações a tecnologia era algo novo a ser aprendido e incorporado ao dia a dia (CORDONI, 2016).

A tecnologia está mudando a maneira que as pessoas vivem, a ampliação das possibilidades de comunicação e informação, através do celular, televisão e computador, altera a forma de viver e aprender na atualidade (KENSKI, 2008). Existe cerca de 7,5 bilhões de *smartphones* e *tablets* sendo utilizados no mundo, isso significa que a quantidade desses aparelhos é maior que o número de pessoas no planeta (WILDEN, 2017). Outro dado interessante é que o número de aparelhos móveis em uso cresce cinco vezes mais que a população global. De acordo com pesquisa realizada pelo Instituto Brasileiro de Geografia e Estatística (IBGE, 2018) a presença do celular no Brasil subiu de 92,6% em 2016, para 93,2% em 2017 e o percentual de domicílios brasileiros que utilizavam a internet aumentou de 69,3% para 74,9%. A pesquisa ainda apontou que em 98,7% dos domicílios pesquisados que continham internet, o celular foi o principal equipamento para acessar a rede. Segundo Streit (2015), diante do crescente número de acesso à internet no Brasil, o ambiente é propício a aumentar a utilização das tecnologias digitais entre os jovens e que a tendência é afetar diretamente o ambiente escolar. Através de tecnologias

educacionais, o acesso à informação é facilitado e auxilia no aprendizado de novas competências.

3.2 MOBILE LEARNING

Uma das ferramentas que está se destacando no aprendizado das novas gerações é o *mobile learning*, ou ainda *m-learning*. Conhecido também como “aprendizagem móvel”, se for traduzido para o português, esse conceito é compreendido como a utilização das tecnologias móveis no contexto educativo (MONTEIRO, 2015). Por conta da sua versatilidade e por permitir realizar as tarefas diárias de forma mais cômoda, o *mobile learning* tem se popularizado por educadores e estudantes. Diante essas circunstâncias, diversas universidades do planeta estão usando o *mobile learning* para transmitir o conhecimento, de maneiras diferentes e em qualquer lugar (AL-EMRAN; ELSHERIF; SHAALAN, 2015). É possível utilizar uma variedade de abordagens pedagógicas apropriadas para o uso em aplicações para dispositivos móveis com o objetivo de ampliar o conhecimento dos alunos. Isso acontece porque o aprendizado com dispositivos móveis, permite o estudante ser mais produtivo em diferentes contextos enquanto consome, partilha ou interage com a informação e com os outros (MONTEIRO, 2015).

No contexto educacional o m-learning se fixa no estudo do uso dos dispositivos móveis na direção de como adaptar conteúdos e aplicações para as plataformas móveis existentes, assim como sobre quais estratégias de cooperação e colaboração utilizar, com o objetivo de torná-las eficientes no processo de ensino e aprendizagem (JÚNIOR; SILVEIRA, 2016).

Importante destacar que o *mobile learning* não se restringe apenas aos aplicativos de celulares e que o aprendizado móvel não pretende trocar o processo de ensino aprendizagem, entretanto permite ser um meio de interação que pode ajudar o discente em suas atividades. Através dos *smartphones* é possível acessar vídeos, enviar materiais para os amigos de turma e entrar em redes sociais para discutir os temas abordados em sala de aula. Estudar por meio dessas redes também é uma nova realidade, através dela é possível trocar informações e tirar dúvidas. O aprendizado é coletivo, por intermédio de blogs ou fóruns espalhados pela internet (MENDONÇA, 2016).

A educação a distância e de modelo híbrido (que combina o ensino presencial e a distância) tem ficado cada vez mais popular nos últimos anos, em consequência disso a tecnologia digital tem sido importante tanto para a disponibilização dos conteúdos online, como no desenvolvimento de alternativas de como pode ser consumido essa informação (RIBEIRO, 2017).

Em pesquisa sobre o uso do *mobile learning* na educação superior realizada por Crompton e Burke (2018), ao analisar 321 estudos e filtrando os que mediam o impacto do aprendizado por meio de dispositivos móveis no desempenho dos alunos, em 70% desses casos foram considerados positivo o uso do *mobile learning* e em apenas 4% relataram impacto negativo no aprendizado. Em 4% desses casos ainda consideram o impacto como positivo e neutro e em 22% consideraram o resultado neutro.

Estudo publicado por West (2012), retrata que a presença dos dispositivos móveis está posicionando para influenciar o ensino e a aprendizagem de um modo que os computadores pessoais nunca conseguiram. Devido a onipresença e a portabilidade dos dispositivos móveis, esses aparelhos possuem vantagens em relação a computadores pessoais, podendo conquistar mais adeptos ao utilizar os recursos presentes nesses novos aparelhos.

Junto com o *mobile learning*, têm surgido alguns modelos de como tornar os estudos ainda mais interativos e atrativos. Uma dessas formas que tem se destacado é o modelo que utiliza além dos recursos presentes em dispositivos móveis, características presentes em jogos na busca do conhecimento. Ao abordar a introdução de jogos durante a aprendizagem, é possível destacar que o uso da *gamification* foi a maneira encontrada para implantar a mecânica e técnicas de jogos no aprendizado (BRAGA; OBREGON, 2015).

3.3 GAMIFICATION

Gamification pode ser compreendido como a utilização de recursos presentes em jogos em outros contextos. O uso em um ambiente escolar visando promover mais engajamento durante o aprendizado pelos alunos é um dos cenários que a

gamification pode ser empregada (LORENZONI, 2016). Como pontos fortes em *games*, pode-se destacar que os conceitos básicos existentes são bem entendidos e conhecidos por grande parte das pessoas que já participaram de algum tipo de jogo (ALVES, 2015). Tonéis (2017) diz que “na gamificação, o ‘jogador’ deve poder se utilizar de estímulos intrínsecos (competição e cooperação) e extrínsecos (pontos, níveis, *ranking*) para realizar as tarefas propostas”. Segundo Alvez (2015), no momento que os jogos estão no ambiente de aprendizagem, será utilizado características presentes nos *games*, porém se for analisado minuciosamente, não será apenas um jogo, pois ele foi introduzido em um contexto de ensino e não consistirá em uma atividade voluntária.

Para transformar o conteúdo usado na aprendizagem em um material no formato de jogo para viabilizar no E-Learning (aprendizado digital) ou no M-Learning, o processo passa pelas etapas de planejamento, design, produção, avaliação, distribuição e manutenção (BRAGA e OBREGON, 2015; KHAN, 2005). É importante envolver durante a criação desse item de estudo, uma equipe para o desenvolvimento de produção multidisciplinar, com a participação desde professores, designer instrucional, designer gráfico, game designer, desenvolvedores de *software*, ilustradores, psicólogos e redatores. Neste processo da criação, a *gamification* vira um subprojeto que será coordenado pelo designer instrucional, no qual este intermediará a produção do jogo com toda equipe envolvida no desenvolvimento (BRAGA; OBREGON, 2015).

Existem características dos jogos que aumentam a motivação e o interesse dos alunos, proporcionando um ambiente de estudos que contém particularidades encontradas em jogos e que podem ser levados ao ambiente de estudos através da *gamification* (TOLOMEI, 2017). Entre as peculiaridades encontradas em *games* Tolomei (2017) destaca:

- **Pontuação:** A utilização de sistema de pontos para recompensar os alunos após cumprir as atividades propostas;
- **Níveis:** Tem como finalidade, mostrar como está o desempenho do usuário na realização das tarefas, geralmente é definido o nível de acordo com a pontuação do jogador;

- **Ranking:** Através do *ranking* é possível o usuário saber como está seu desempenho em relação aos outros utilizadores do *game* e visa também estimular a competitividade entre eles;
- **Medalhas/Conquistas:** São elementos gráficos que são concedidos aos usuários após a conclusão das atividades propostas;
- **Desafio e missões:** Este item corresponde às tarefas específicas que os jogadores precisam realizar, para conseguirem conquistar as recompensas (pontos e medalhas), permitindo a sensação de ter sido desafiado.

Ao finalizar e disponibilizar o jogo para realizar as atividades com *gamification*, o aprendizado a partir dele têm como benefício um ambiente favorável ao engajamento dos estudantes nas atividades escolares que podem ter sido considerado por eles enfadonhas, mas com a utilização dos jogos é possível que o aluno tenha mais empatia ao processo de aprendizagem, já que foi levado para sua própria realidade o conteúdo de um modo que ele pode achar mais aceitável, pelo ambiente que ele costuma ter mais afinidade. Dessa maneira o estudante pode despertar o interesse no aprendizado em primeiro lugar pelo fato do ambiente estimular a conclusão das atividades propostas para avançar no curso e conquistar as recompensas, secundamente por facilitar o acesso, dado que pode ser acessado através de *smartphones*, *tablets* e computadores, permitindo se adaptar ao aparelho que o aluno achar mais interessante e que tem a disposição no momento da realização da atividade (TOLOMEI, 2017).

Cremonetti (2016) realizou uma pesquisa com alunos e professores do ensino médio do Colégio de Aplicação da Universidade Federal de Roraima para avaliar a utilização de um aplicativo que utilizava práticas pedagógicas com técnicas de *gamification* como apoio no ensino de matrizes e determinantes. Nesse estudo mais de 95% dos alunos que responderam ao questionário aprovaram o uso do aplicativo para o estudo. E na avaliação dos docentes, foi considerado como muito boa a utilização de jogos matemáticos em sala de aula e que tinham interesse em usar os dispositivos móveis para passar tarefas relacionadas à matéria estudada.

Em estudo realizado por Vargas (2017) foi analisado o processo de aprendizagem e avaliação por meio do uso de Quiz. No trabalho foi elaborada uma pesquisa a partir do uso das aplicações Kahoot, Socrative e do Plickers. Foi avaliado

a utilização do Quiz em conjunto com a metodologia ativa por pares ou *peer instruction*. O artigo teve como núcleo a avaliação sobre a percepção do professor e no final do estudo, chegou-se a conclusão que o uso dos aplicativos aproximou o professor do aluno e viabilizou um progresso no processo de construção de saberes, aumentando a interação e motivação dos alunos que participaram da atividade.

Segundo Rodrigues (2014) a usabilidade de soluções como o Quiz do Moodle (*software* livre de apoio à aprendizagem) não motiva o suficiente os professores a utilizarem o recurso dessa plataforma, mesmo os docentes possuindo interesse no uso de um sistema de perguntas e respostas com correção automática e considerando ser uma ferramenta útil no ensino. Para contornar essa situação o autor desenvolveu uma alternativa ao módulo chamando-o de 'iQuiz'. Como resultado da pesquisa a média de professores que concordaram que gostaria de usar o sistema Quiz frequentemente, passou de 84,61% para 92,30% com a proposta do trabalho. Em relação a facilidade de usar esse recurso, passou de 58,46% para 90,76% e no item que avaliava se a maioria das pessoas aprenderia usar o sistema rapidamente, passou de 55,38% para 90,76% com a nova proposta.

Ao utilizar os aparelhos móveis e seus aplicativos como apoio ao aprendizado, tem a possibilidade de usar plataformas que podem comportar diferente para cada tipo de aluno, adaptando o material estudado de acordo com as dificuldades encontradas pelo estudante.

3.4 PLATAFORMAS ADAPTATIVAS

Plataformas adaptativas são sistemas inteligentes que recomendam atividades diferentes para cada aluno, propondo um conteúdo personalizado sob medida, a partir do perfil do aluno que foi traçado como base das respostas que ele deu anteriormente, mensurando o desempenho e permitindo adaptar qual tipo de conteúdo o estudante precisa absorver para evitar cometer os mesmos erros cometidos no passado (LEAL, 2018).

Através da aprendizagem adaptativa e a coleta de informações dos exercícios resolvidos pelos alunos neste tipo de plataforma, os professores são capazes de

conhecer melhor algumas características dos alunos como suas dificuldades e ainda permite o docente diante esses dados reavaliar os rumos, alterar metas, interferir no planejamento, entre outros tipos de estratégias educacionais, possibilitando assim benefícios para alunos e professores (LOPES, 2016).

Reis e Barrere (2015) realizaram uma pesquisa para avaliar a utilização de um aplicativo que faz recomendações de conteúdo e possibilita o professor escolher um material de estudo para indicar para seus alunos de acordo com o nível de conhecimento e contexto no qual estes estudantes estão incluídos. Para conhecer o perfil do estudante foram coletados dados usando o monitoramento de logs do aplicativo e foi aplicado um questionário para os alunos responderem ao final dos testes. Ao finalizar o trabalho, os autores concluíram que o uso de dispositivos móveis e o sistema de recomendações fizeram aumentar a demanda para consumir informações sobre os conteúdos estudados pelo aluno, deixando-os mais motivados durante o aprendizado.

Lopes (2016) apresentou cinco vantagens que a utilização de aprendizagem adaptativa traz para o professor:

- Permitir o encontro de um roteiro de aprendizagem adaptado com as necessidades de cada aluno, possibilitando resultados mais eficazes;
- Entregar informações úteis sobre os alunos, além dos resultados das atividades, também dados como o tempo que ele levou para realizar a tarefa, de forma automática;
- Evitar o gasto de muito tempo corrigindo as atividades, possibilitando utilizar esse tempo dando maior atenção pessoal para cada aluno;
- Auxiliar na organização e programação das aulas, permitindo personalizar o conteúdo de acordo com a necessidade dos alunos e
- Possibilitar uma maior autonomia para o professor, podendo promover outros tipos de atividades ou levando a prática.

Além dessas vantagens para o professor, Lopes (2016) enumerou também os benefícios que o aprendizado adaptativo possibilita para os estudantes:

- Personalização de acordo com as necessidades individuais, permitindo que os alunos tenham um planejamento de estudo sob medida, adaptado a suas necessidades;
- O aprendizado é mais efetivo e mais ágil, mostrando seus pontos fortes e fracos, permitindo analisar seus erros e possibilitando saber onde é necessário dar mais atenção;
- É estimulante, já que funciona como um desafio para o aluno, que ele é capaz de enfrentar, tendo um retorno rápido, podendo deixá-lo mais confiante;
- Melhorar a capacidade do estudante aprender de forma autônoma e
- Permitir o acesso da plataforma em qualquer lugar, possibilitando estudar em casa e por meio de dispositivos móveis.

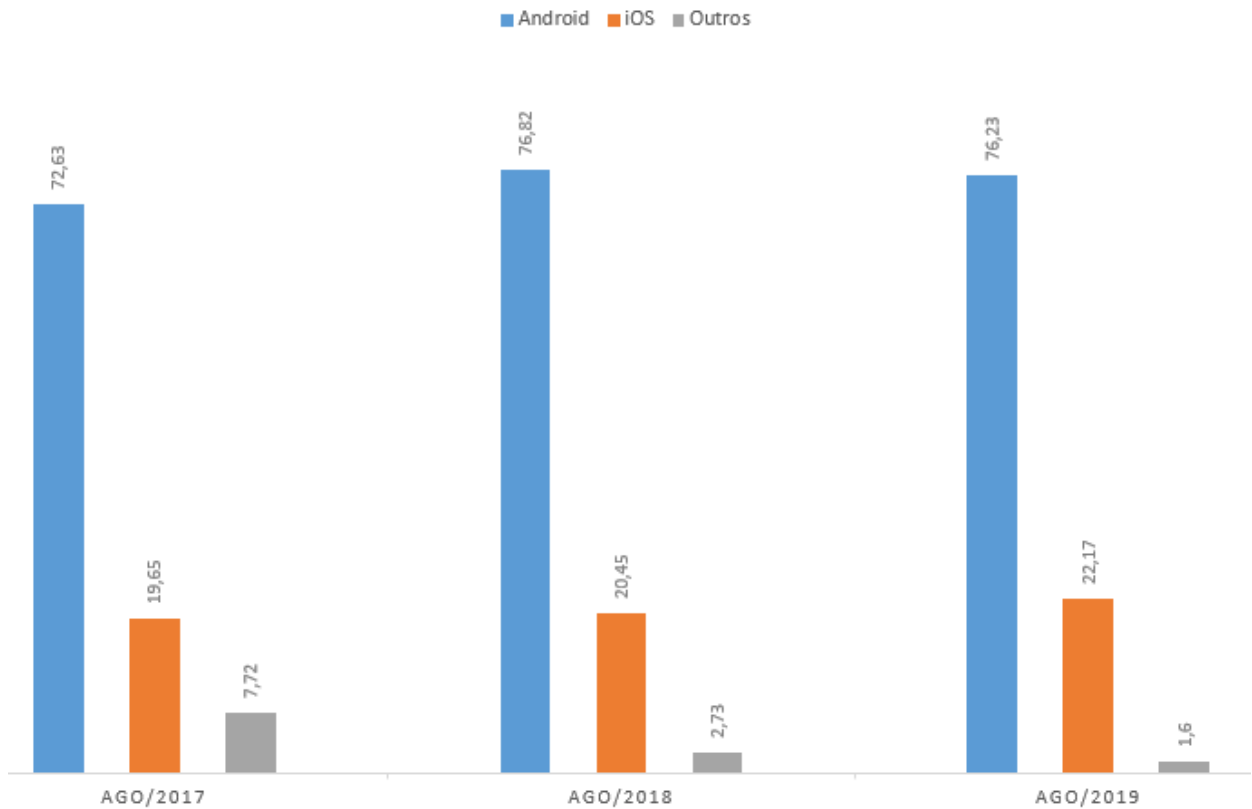
3.5 SISTEMAS OPERACIONAIS PARA DISPOSITIVOS MÓVEIS

Para o desenvolvimento de uma plataforma de ensino que pode ser utilizada em *smartphones* e *tablets*, é necessário conhecer as características dos sistemas operacionais existentes para dispositivos móveis, uma vez que cada sistema possui suas particularidades.

De acordo com Pressman e Maxim (2016), as plataformas móveis assim como os dispositivos de computação, são diferenciados pelo *software* que possuem tanto como o sistema operacional (Android, iOS entre outros) quanto pelo seus aplicativos que são instalados no sistema. Muitos desenvolvedores de *software* acreditam que os aplicativos criados para dispositivos móveis são um dos *softwares* mais desafiadores para sua construção, pois as plataformas utilizadas pelos dispositivos móveis são muito complexas, ao ponto de os sistemas operacionais Android e o iOS chegar a alcançar mais de 12 milhões de linhas de código. Cada dispositivo móvel possui um sistema operacional e este possui um ambiente de desenvolvimento com características próprias para ele, além disso os aparelhos podem possuir o tamanho de tela de diversas medidas diferentes, influenciando assim na construção do layout para cada parte visível do sistema. Os aplicativos móveis são escritos por no mínimo três linguagens de programação diferentes (Java, Objective C e C#), para pelo menos

cinco *frameworks* de desenvolvimento (Android, iOS, BlackBerry, Windows e Symbian) (PRESSMAN; MAXIM, 2016).

Figura 1 - Uso dos sistemas operacionais móveis no mundo



Fonte: Statcounter (2019), adaptado pelo autor

Segundo pesquisa realizada pelo Statcounter (2019) como demonstrado na Figura 1, o sistema Android continua no topo entre os sistemas operacionais móveis mais utilizados do mundo com 76,23% de participação no mercado em agosto de 2019, e o iOS ficou em segundo lugar em 22,17% no mesmo período. É possível observar que apesar de existirem várias opções de sistemas operacionais móveis, os mais populares são o Android e o iOS. Com o sistema Android sendo mais popular que o sistema da Apple no mundo e que o iOS tem uma participação muito mais significativa que os outros sistemas operacionais, com exceção ao sistema do Google.

A origem do sistema Android, começou através de uma empresa fundada em 2003, chamada Android Inc. e que era voltada para o desenvolvimento de aplicativos móveis em pleno Vale do Silício nos Estados Unidos, logo o Google atento às inovações, percebeu que a empresa tinha um bom propósito e começou a financiar

as pesquisas realizadas por ela. Pouco tempo depois a empresa criou um sistema operacional com seu próprio nome, Android e em 2005 o Google adquiriu a empresa Android Inc. (CARLOS, 2013). A principal linguagem para desenvolvimento dos aplicativos para a plataforma é o Java, porém desde 2017 é possível desenvolver aplicativos utilizando também Kotlin, como linguagem oficial (MATEUS; MARTINEZ, 2018).

A primeira versão do Android, foi lançada em 2008, e desde seu lançamento foram disponibilizadas várias versões visando melhorias. Com essas versões são disponibilizados novos recursos, classes e bibliotecas que podem auxiliar no desenvolvimento de aplicativos além disso a partir do número ou nome da versão, é possível saber quais APIs estão presentes nela, para durante o desenvolvimento ser possível utilizar os recursos presentes nos aparelhos do público alvo, uma vez que se for usado uma classe, ou ainda apenas um método que não está presente nesta versão, ocasionará um erro no momento que esse código for executado pela aplicação (GLAUBER, 2019).

Até a versão Gingerbread, o Android rodava exclusivamente em smartphones, mas quando os primeiros tablets Android foram lançados, eles vinham com a versão Honeycomb, que era exclusiva para esse tipo de aparelho. Com o lançamento do Ice Cream Sandwich, houve a reunificação da plataforma, e novamente a mesma versão do Android passou a rodar em smartphones e tablets (GLAUBER, 2019).

De acordo com dados disponibilizados pelo proprietário do sistema operacional Android e disponibilizados no quadro a baixo, o Google (2019), informa que a versão mais recente do sistema operacional, que possui o codinome 'Pie' estava presente em 10,4% dos aparelhos que possuem o sistema operacional da empresa em março de 2019.

Quadro 1: Distribuição das versões do Android

Versão	Nome	API	Distribuição
2.3.3 - 2.3.7	Gingerbread	10	0.3%
4.0.3 - 4.0.4	Ice Cream Sandwich	15	0.3%
4.1.x	Jelly Bean	16	1.2%
4.2.x		17	1.5%
4.3		18	0.5%
4.4	KitKat	19	6.9%
5.0	Lollipop	21	3.0%
5.1		22	11.5%
6.0	Marshmallow	23	16.9%
7.0	Nougat	24	11.4%
7.1		25	7.8%
8.0	Oreo	26	12.9%
8.1		27	15.4%
9	Pie	28	10.4%

Fonte: Google (2019), adaptado pelo autor

Durante o desenvolvimento de uma aplicação Android, no arquivo 'AndroidManifest.xml' que é informado boa parte das configurações do aplicativo. É ele que define em quais versões do sistema Android que a aplicação irá ter suporte para um bom funcionamento de acordo com os recursos presentes nesta versão. Neste arquivo pode-se definir também outras configurações, como nome do aplicativo e as permissões que ele necessitará do dispositivo para executar as funcionalidades incluídas (GLAUBER, 2019).

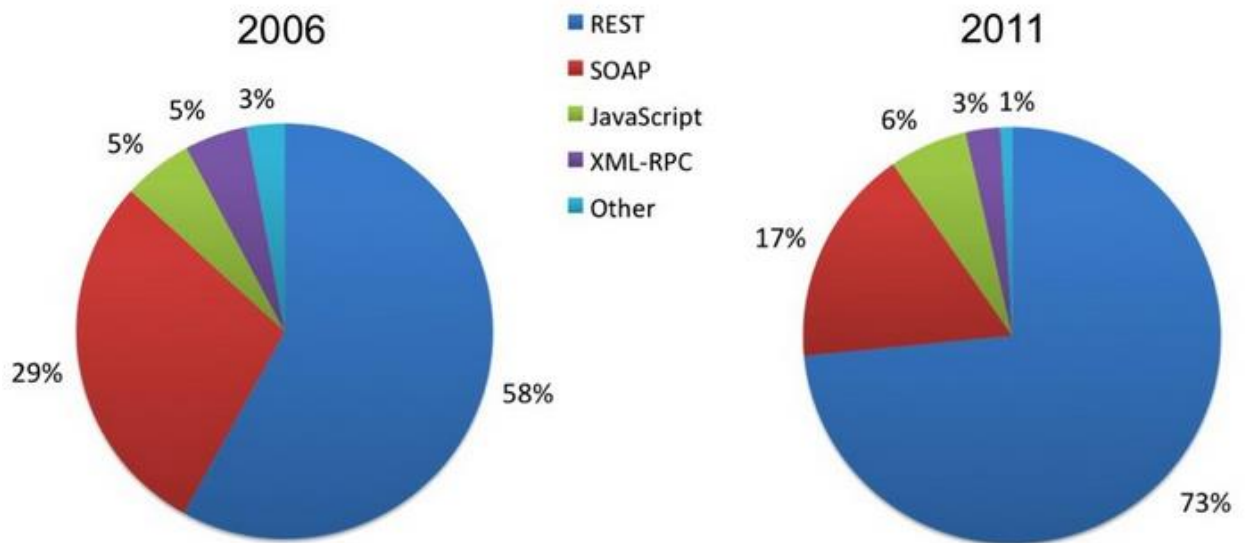
Aplicativos desenvolvidos para Android ou outros sistemas operacionais, utilizam frequentemente dados armazenados em servidores externos e que são

acessados por meio da internet. Na comunicação entre a aplicação e esses servidores é necessário ser usado um padrão de comunicação que os dois aparelhos consigam falar um 'idioma' em comum. Para solucionar esse problema, um método muito utilizado para realizar essa troca de informações entre os sistemas é por meio de *Web Services*.

3.6 WEB SERVICES

O *web service* é o principal meio de integração entre sistemas, possibilitando a comunicação entre eles independentemente da plataforma de *hardware* e *software* utilizado, ganhando um papel ainda mais importante devido ao uso da mobilidade. É através de *web services* que é realizado a comunicação entre os aplicativos desenvolvidos para dispositivos móveis. A chegada do *web service* ocorreu como uma evolução dos sistemas da segunda metade dos anos 90 que utilizavam modelos de computação distribuídas, porém essas tecnologias obtiveram mais êxito na integração de *softwares* que eram utilizados em ambientes de redes locais e homogêneos. Com o crescimento do uso da internet no mercado corporativo, criou-se a necessidade de fazer integração dos seus sistemas para além das redes locais, como resultado foi necessário realizar a comunicação em ambientes heterogêneos que utilizavam diferentes tipos de tecnologias para trocar informações. Com essa nova realidade que surgiu essa tecnologia conhecida como *web services*, que foi criado a partir de um consórcio formado por empresas que pertenciam ao W3C e outras grandes corporações como Microsoft, IBM e BEA. Baseado nas tecnologias que existiam nesse período para a criação de aplicações para a internet, foi desenvolvido um padrão chamado SOAP (*Simple Object Access Protocol*) que permitia a comunicação entre sistemas que poderiam ter utilizado diferentes linguagens de programação em seu desenvolvimento, não importando o *hardware* nem mesmo o sistema operacional que cada aplicação estava sendo executado (GOMES, 2014).

Figura 2 - REST vs. SOAP



Fonte: Avram (2011), adaptado pelo autor

Existem dois padrões de *web services* muito utilizados, o SOAP e o padrão REST ou RESTfull (GOMES, 2014). Como observado na Figura 2, nota-se que o uso do REST tem crescido em relação aos anos de 2006 para 2011, esse resultado pode ter ocorrido devido a simplicidade de adotar a arquitetura REST em relação ao SOAP.

3.6.1 REST

A Transferência de Estado Representacional (tradução do termo *Representational State Transfer* - REST) representa um tipo de arquitetura de *software* com o uso crescente em todo planeta que auxilia na integração de sistemas e na elaboração de serviços disponibilizados na internet. Para produzir esses serviços o REST faz o uso do protocolo HTTP retornando dados geralmente no formato XML ou JSON com proposito de fornecer como opção em relação ao SOAP e o WSDL. Ao ser criado um sistema ou serviço com base nos conceitos do REST, é possível dizer que ele é *web service* RESTful (LECHETA, 2015). Como características dos serviços RESTful pode-se destacar o uso das operações utilizadas na tabela a seguir:

Tabela 1 – Operações do protocolo HTTP para manipulação de dados

Protocolo HTTP equivalente	Ação com os dados
GET	Utilizado para realizar consultas
POST	Utilizado para inserir
PUT	Utilizado para atualizar
DELETE	Utilizado para remover

Fonte: Lecheta (2015), adaptado pelo autor

De acordo com Lecheta (2015), além das operações presentes no protocolo HTTP para as requisições web, os serviços REST possuem mais algumas particularidades como a possibilidade de retornar informações para o serviço web através de conteúdo (mime-type), deixando o cliente escolher o tipo de conteúdo que deseja receber do servidor como JSON ou XML. Cada recurso do serviço REST precisa ter uma sintaxe universal para identificar recursos (*Uniform Resource Identifier* - URI), para que exista a possibilidade de pesquisar informações sobre este recurso. E outra peculiaridade do REST é que ele possui um conjunto de operações padronizadas para realizar as requisições do tipo (*GET, POST, PUT e DELETE*).

As requisições respondidas pelo servidor através do protocolo HTTP, fornecem além do seu conteúdo, informações no seu cabeçalho que pode ser útil para a aplicação que irá receber e tratar esta resposta. Uma dessas informações contidas no cabeçalho da resposta é o status da solicitação, que é uma forma simplificada de dar pistas do que aconteceu com a requisição que foi feita para o servidor. É possível receber diversos tipos de respostas utilizando classes de status diferentes que mudam de acordo com o que ocorreu com a requisição (MOLINARI, 2016).

Apesar da popularidade e dos benefícios que vieram com a arquitetura REST, existem algumas limitações nesse modelo como a necessidade de realizar várias requisições para conseguir todas as informações necessárias ou ainda, do problema ao ser enviado mais dados que o necessário pelas requisições. Uma alternativa que surgiu para resolver essas adversidades é chamada de GraphQL.

3.6.2 GraphQL

Com o surgimento dos bancos de dados relacionais houve a necessidade da criação também de uma maneira de manipular essas informações, e para essa tarefa foi criada a Linguagem de Consulta Estruturada (*Structured Query Language* - SQL), que pode ser entendido como uma linguagem utilizada para a administração de manipulação dos dados de algum banco de dados. Com os comandos que o SQL possui (*SELECT*, *INSERT*, *UPDATE* e *DELETE*) tornou fácil a manipulação dos dados dos bancos de dados. E ainda, através do SQL é possível escrever um comando que consiga retornar dados disponíveis em várias tabelas do banco de dados. A ideia que os dados podem ser apenas lidos, criados, atualizados ou apagados chegou até o REST, e ele exige que seja utilizado os métodos HTTP para realizar as quatro operações básicas (*GET*, *POST*, *PUT*, *DELETE*). Porém para acessar os tipos de dados que precisamos a partir do REST, temos a necessidade de utilizar as URLs de endpoints e não uma linguagem de consulta. Em razão desta limitação, foi criado o GraphQL para utilizar as ideias criadas para fazer pesquisas nos bancos de dados e aplicar esse conceito para a internet. Através de uma única consulta a partir o GraphQL é possível devolver dados interligados, permitindo também fazer manipulação e escolher os dados que a requisição necessita (PORCELLO; BANKS, 2018).

Ainda que GraphQL e SQL sejam linguagens de consulta, elas são utilizadas para propósitos distintos. Enquanto as consultas em SQL são enviadas a um banco de dados, no caso do GraphQL elas são enviadas para uma API. Em relação ao armazenamento de dados, no SQL são feitos em tabelas de dados, e quando é usado o GraphQL eles podem ser armazenados em qualquer lugar, podendo ser um ou mais banco de dados, sistemas de arquivos, APIs REST ou até mesmo em outras APIs GraphQL. Resumindo o SQL é uma linguagem para realizar consultas em banco de dados e o GraphQL é uma linguagem para fazer consultas em uma rede (PORCELLO; BANKS, 2018).

O GraphQL é uma tecnologia relativamente nova, foi criada pelo Facebook para utiliza-lo em seus aplicativos em 2012, sendo anunciado ao público em 2015 e somente disponibilizou seu código aberto em 2016. No momento o GraphQL é um projeto da GraphQL Foundation e foi hospedado pela Linux Foundation e é mantido

por várias grandes empresas como Airbnb, Coursera, Facebook, GitHub, Shopify, Twitter entre outras. Ao usar o GraphQL é necessário fazer a declaração de como deve ser a estrutura de dados, dessa forma o cliente consegue saber a qual estrutura que ele pode receber, permitindo dessa forma também a geração automática de documentação e validação dos tipos. Outra vantagem encontrada nessa tecnologia é que é possível receber dados de vários objetos em apenas uma requisição, exatamente o que precisar, sem dados a mais ou a menos (HANASHIRO, 2019).

Em pesquisa realizada por Vázquez-Ingelmo, Cruz-Benito e García-Peñalvo (2017) foi comparado o desempenho da utilização do REST e do GraphQL. Em análise da média do tempo de resposta pela API REST, ele gastou 6,15 segundos e na API GraphQL 4,16 segundos, conseguindo assim reduzir em 32,36% a espera nas requisições com o GraphQL. Quando comparado o tamanho das requisições feitas pelas duas APIs mostrou que com o REST gastou 26,66 KB enquanto o GraphQL apenas 13,90 KB, desta maneira a API que usou o GraphQL foi 47,86% mais eficiente na entrega das requisições.

3.7 PADRÕES DE ARQUITETURAS DE SOFTWARE

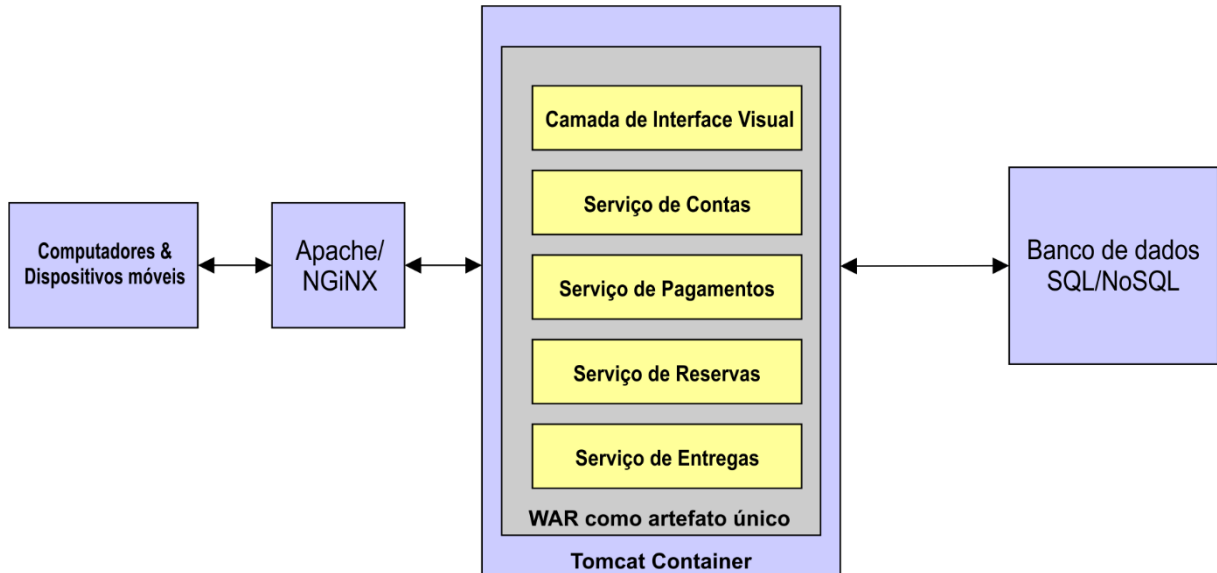
Arquitetos de *software* precisam ter bastante atenção na estrutura durante o processo de construção de *softwares*, pois a solução desenvolvida pode precisar de atender milhares de requisições por segundo e para suportar essa demanda, a aplicação necessita ser escalável tanto horizontalmente quanto verticalmente (RAJPUT, 2018).

3.7.1 Arquitetura Monolítica

Uma arquitetura de aplicação do tipo monolítica tem como característica um *software* que possui apenas um artefato e nele inclui todas as camadas da aplicação, como é exibido na Figura 3, que abrange a interface visual do cliente que contém as páginas HTML e no lado do servidor a lógica que vai receber e atualizar o banco de

dados englobando as classes DAO (*Data Access Object*) que manipularam esses dados (RAJPUT, 2018).

Figura 3 - Arquitetura tradicional de uma aplicação monolítica



Fonte: Rajput (2018), adaptado pelo autor

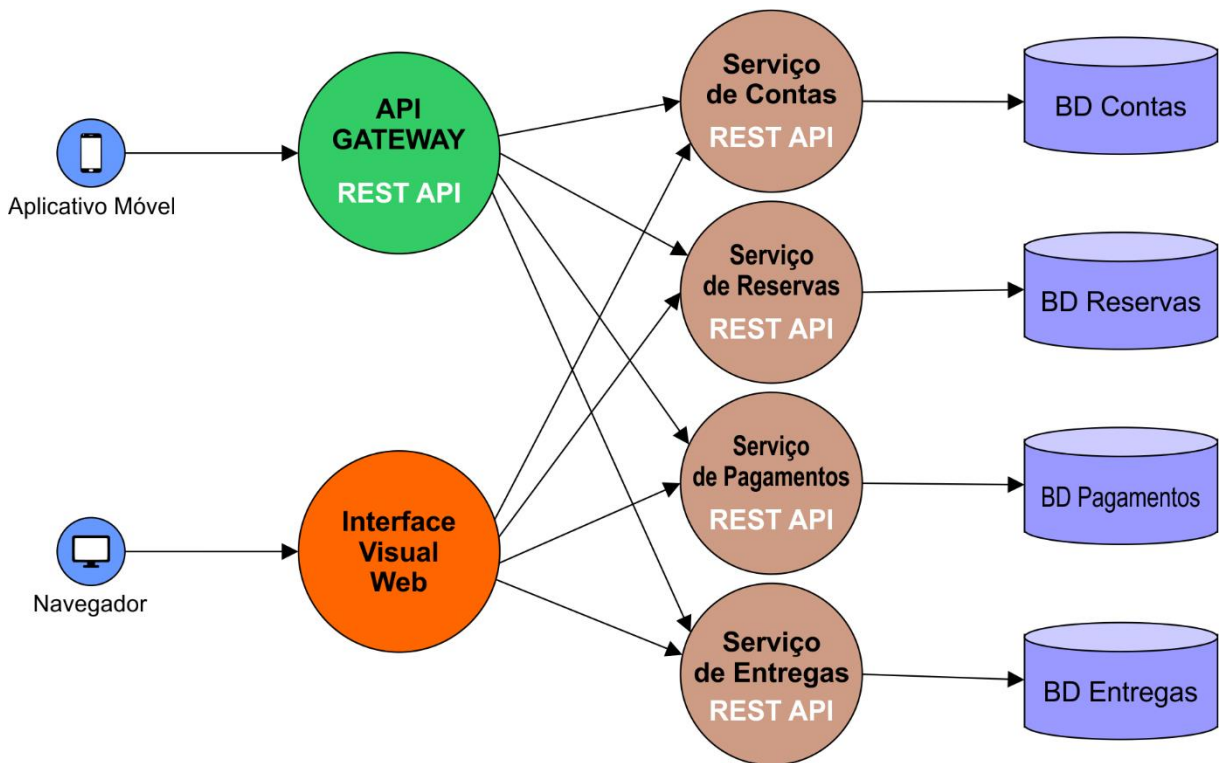
Programas que usam este tipo de arquitetura frequentemente possuem uma grande base de códigos e quando é necessário fazer uma alteração torna preciso atualizar a versão da aplicação no lado do servidor para não perder o controle das modificações realizadas (RAJPUT, 2018).

3.7.2 Arquitetura de *Microservices*

Microservices são o oposto do desenvolvimento da arquitetura monolítica. Um *software* monolito é um sistema grande e permite fazer a implantação do sistema no serviço apenas enviando uma única parte, sendo que ele teria que passar por todas etapas da entrega contínua, que consiste nas fases de desenvolvimento, testes e entrega. Devido ao desenvolvimento monolítico ser uma peça grande, percorrer essas fases do processo se torna algo trabalhoso, diminuindo a flexibilidade e aumentando o custo do processo. Esse monolito pode até ter sido desenvolvido utilizando uma estrutura modular, porém para seu funcionamento seria necessário que todos módulos entrassem em produção ao mesmo tempo. Já no caso de sistemas que foram

desenvolvidos utilizando a arquitetura de *microservices*, por ser um *software* dividido em módulos separados, torna as alterações que necessitam ser realizadas um processo mais simples de ser feito tanto no sentido de desenvolvimento quanto no de publicação no servidor de produção (WOLFF, 2016).

Figura 4 - Exemplo de uma arquitetura de *microservices*



Fonte: Rajput (2018), adaptado pelo autor

Na Figura 4 pode-se observar que diferentemente da arquitetura monolítica, as camadas correspondentes a conta de usuário, lista de livros para vendas, lógica do pagamento e de entrega que ficavam todos juntos num mesmo lugar, na arquitetura de *microservices* cada uma desses módulos são separados em pequenas aplicações e serviços distintos. Desta forma cada serviço pode ser desenvolvido e atualizado de forma independente no servidor, não ficando amarrado aos outros processos como é feito na arquitetura monolítica. Pelo fato de estarem separados, cada parte do serviço pode ser desenvolvido utilizando uma tecnologia diferente, permitindo usar o melhor de cada uma de acordo com o contexto que ela possui vantagem em relação a outras tecnologias. Além desses benefícios que é possível usufruir com a arquitetura de *microservices*, Rajput (2018) lista essas outras vantagens:

- **Facilidade na manutenção:** O entendimento se torna muito mais fácil para o desenvolvedor, dessa forma o desenvolvimento se torna mais rápido e mais produtivo;
- **Facilidade em escalar:** Permite escalar os recursos de acordo com a demanda de cada serviço;
- **Diversificar a tecnologia:** A utilização de *microservices* permite utilizar frameworks, banco de dados e linguagens diferentes;
- **Isolamento de falhas:** Caso um componente falhe ele não derruba toda aplicação;
- **Melhor suporte:** A arquitetura é adequada para times pequenos e permite o trabalho em paralelo;
- **Implantação independente:** É possível implantar um *microservice* de forma independente sem afetar o funcionamento de outro *microservice*.

Pelo fato do desenvolvimento utilizando a arquitetura de *microservices* proporcionar esses benefícios, o uso do processo de desenvolvimento ágil funciona muito bem, uma vez que nesse processo é dividido em pequenas equipes para trabalharem em partes da aplicação, podendo separar as equipes para cada uma ficar responsável por um módulo que foi separado nesse tipo de arquitetura (RAJPUT, 2018).

3.8 HOSPEDAGEM NA NUVEM USANDO *SERVERLESS*

Para hospedar e disponibilizar seus aplicativos, os desenvolvedores tradicionalmente faziam o uso de servidores dedicados e que ficavam geralmente em *datacenters*. Os custos para adquirir essas máquinas inicialmente eram elevados e toda vez que tinham um pico de acesso, para suportar a demanda era necessário comprar novos equipamentos, que após passar o período de muito acesso, essas novas máquinas poderiam ficar ociosas (ADZIC; CHATLEY, 2017). Depois que surgiu a computação nas nuvens, os desenvolvedores não precisam mais de adquirir máquinas físicas, passando a utilizar máquinas virtuais, permitindo responder ao

crescimento de acessos de forma mais rápida, permitindo também o aumento dos recursos disponibilizados pelo servidor apenas em períodos específicos.

As arquiteturas tradicionais de cliente/servidor envolvem um processo de servidor que permanece aguardando o acesso de clientes para conectar e enviar requisições, além desse papel, esses servidores geralmente possuem mais responsabilidades, servindo como um porteiro das requisições que chegam e executando outras tarefas, além das regras do sistema. Com o surgimento e utilização do modelo *serverless*, os desenvolvedores são responsáveis pela regra do processamento de um evento enviando as tarefas e agendamentos, porém o servidor que se compromete em receber e responder os clientes (ADZIC; CHATLEY, 2017).

3.8.1 Serverless

Com a apresentação da plataforma 'Lambda' pela *Amazon Web Services* (AWS) no fim de 2014 e sua forte aceitação no segundo semestre de 2016, os principais provedores de infraestrutura de computação na nuvem lançaram serviços que possuíam um estilo semelhante de implantação e operação. Diferente dos serviços existentes anteriores a essa tecnologia, que implantavam serviços monolíticos e máquinas virtuais dedicadas, através desse novo recurso tornava possível os usuários implantar funções individuais, pagando apenas pelo tempo que ela era executada. Os fornecedores dessas tecnologias acreditam que ela tem capacidade de alterar consideravelmente como as aplicações cliente/servidor são estruturadas, desenvolvidas e operadas. O termo criado para o entendimento da reunião dessas tecnologias é conhecido como '*serverless*' e se refere a nova geração das plataformas de serviços na nuvem (ADZIC; CHATLEY, 2017).

O modelo *serverless* refere a aplicações que o gerenciamento do servidor e as tarefas operacionais são ocultas dos usuários e desenvolvedores. Permitindo nesse modelo os desenvolvedores se dedicarem particularmente no código da lógica de negócio e na aplicação. Não precisando se preocupar em onde está o servidor ou sua performance. Gupta (2018) relata que se bem implementada, a arquitetura *serverless* pode proporcionar algumas vantagens:

- **Alta disponibilidade e tolerância a falhas:** As aplicações *serverless* são construídas com arquiteturas que tem suporte a alta disponibilidade. Na AWS por exemplo, é utilizado um conceito de zonas, podendo deixar disponível a aplicação para uma região próxima ao usuário, mas caso aconteça algum problema, pode continuar a execução em outra região. Dessa forma, o desenvolvedor não precisa implementar esse tipo de capacidade no seu sistema;
- **Escalabilidade:** É desejado que a aplicação seja um sucesso, e caso aumente a demanda de acesso por ela, que mantenha o bom funcionamento. Com o *serverless*, é possível escalar facilmente a aplicação. Esse modelo irá rodar utilizando os limites definidos pelo programador permitindo aumentar de acordo com a necessidade, incrementando a quantidade de memória por exemplo, através de poucos cliques;
- **Produtividade do desenvolvedor:** Toda complexidade de configuração de *hardware*, rede, instalação e gerenciamento de *software* é feita pelo fornecedor do *serverless*. Desta maneira, os programadores podem focar apenas na implementação da regra de negócio, conseqüentemente a produtividade do desenvolvedor é elevada, uma vez que ele não precisará se preocupar de como a engenharia do sistema foi feita no servidor;
- **Sem capacidade ociosa:** No modelo *serverless*, não é necessário ter uma capacidade de armazenamento e computação avançada. Em outras palavras, isso significa que caso um site de *e-commerce* estiver durante datas comemorativas ou promoções, pode ser que precise de uma capacidade elevada por poucos dias, dessa maneira é possível aumentar os recursos do site nesse dia e após esse período de pico de acesso, voltar a usar apenas os recursos necessários que utilizava anteriormente;
- **Implantação em minutos:** O modelo *serverless* facilita a implantação, fazendo o trabalho duro e eliminando o entendimento da infraestrutura;
- **Sem gerenciamento de servidor:** Tarefas como gerenciamento de servidor são complicadas, preparar e testar o servidor pode demorar de dias até meses. E isto pode ser um problema para disponibilizar um *software* no mercado. Com o modelo *serverless*, tem o benefício de mascarar todo trabalho de engenharia no servidor do time de desenvolvimento do projeto;

- **Rapidez para o mercado:** Ao desenvolver uma aplicação usando o modelo *serverless*, em questão de minutos é possível ter uma infraestrutura pronta para ser utilizada. Podendo aumentar ou diminuir o *hardware* usado com apenas poucos cliques. Desta forma, é economizado o tempo gasto com o trabalho de engenharia e é lançado aplicações rapidamente. Essa é uma das vantagens que está fazendo companhias passarem a utilizar o modelo *serverless* em seus produtos.

A tecnologia da computação *serverless*, que corresponde a nova geração de plataforma como serviço (*platform-as-a-service*), está disponível além do Lambda da *Amazon Web Services* (AWS), pelos principais provedores de serviço nas nuvens, podendo usar esses recursos também no *Google Cloud Functions*, *Azure Functions* e *IBM OpenWhisk*. Um dos benefícios da utilização do modelo *serverless* é a possibilidade de disponibilizar mais de uma versão do sistema ao mesmo tempo e não ser necessário manter diferentes servidores sempre ligados para ofertar cada versão da aplicação, precisando ainda de aumentar o gasto com essas máquinas mesmo sem sua utilização (ADZIC; CHATLEY, 2017).

Existem estudos, como o do Adzic e Chatley (2017) que relatam a migração de arquitetura feita por empresas que passaram a utilizar o modelo *serverless* e seus resultados. Como o caso da empresa MindMup que moveu sua arquitetura do Heroku para usar os recursos do *serverless* ofertados pela *Amazon* e o caso da empresa londrina Yubl que já utilizava os servidores da AWS, porém com arquitetura monolítica, quando migrou seu sistema para a arquitetura *serverless* da *Amazon* tiveram economia nos gastos de servidor em cerca de 95%. Além da economia, as duas empresas notaram o aumento de produtividade dos desenvolvedores, uma vez que com o novo modelo utilizado foi possível disponibilizar mais funcionalidades nos seus sistemas em menos tempo. Dessa maneira estas duas empresas ao passar a usar o modelo *serverless* conseguiram além de economizar no pagamento dos servidores, aumentar a produtividade das equipes de desenvolvimento (ADZIC; CHATLEY, 2017).

4. MATERIAIS E MÉTODOS

4.1 TECNOLOGIAS

Para tornar a interface da aplicação desenvolvida com qualidade e deixar o processo de desenvolvimento produtivo será utilizada a biblioteca React. Foi escolhido essa tecnologia entre outras opções pelo fato dela ter sido usada na criação de aplicativos consolidados no mercado e pela sua aceitação, uma vez que ela ficou na liderança em pesquisa realizada pelo Stateofjs (2018) como *framework* mais usado e desejado pelos profissionais da área. Em conjunto com o React será utilizada a plataforma Node.js neste trabalho para auxiliar na instalação de bibliotecas e suas dependências. Na aplicação do lado do servidor, será utilizado além do Node.js o Python, por ter um bom desempenho ao executar instruções no Lambda da Amazon e ser uma linguagem que tem se destacado nos últimos anos no mercado de desenvolvimento.

4.1.1 React

O React é uma biblioteca JavaScript desenvolvida pelo Facebook, através dele é possível a criação e o reuso de componentes para montar a interface da tela, separando cada pedaço do layout em um componente diferente e permitindo a construção da tela através de blocos (RAJPUT, 2019). Além de ter uma boa aceitação da comunidade de desenvolvedores no mundo, possui algumas vantagens (RAJPUT, 2019):

- Um dos principais recursos do React JS é o JSX, que significa JavaScript e XML. Não é um código HTML como parece ser, mas o JSX é usado para criar componentes React e essas peças constroem os blocos da interface através do React UI;
- Outra vantagem do uso do React JS é a performance da aplicação, já que ele utiliza um virtual DOM para manipular os objetos e desta forma consegue montar o DOM da tela mais rápido que se estivesse usando o DOM comum;

- Pelo fato do React utilizar componentes para montar a interface, utilizando em pequenas partes visuais, é facilitado o reuso de cada componente do mesmo tipo em outras partes da aplicação;
- Com o layout dividido em pequenos pedaços para montar o visual da página, o código escrito utilizando esses trechos de componentes torna a leitura do código de fácil entendimento além de usar padrões para tornar mais simples a manutenção de aplicações grandes.

4.1.2 Node.js

Apresentado por Ryan Dahl em 2009 na JSConf Europa, o Node.js é uma plataforma robusta para o desenvolvimento de aplicações de forma simplificada, através do motor V8 que é uma *engine* JavaScript *open source* de alta performance do navegador Google Chrome. O Node.js "utiliza uma arquitetura orientada a eventos e um modelo I/O não bloqueante que faz com que seja leve e eficiente" (MORAES, 2018). Essas particularidades são ideais para resolver problemas de intenso tráfego de rede e aplicação em tempo real, que em vários momentos é uma das principais dificuldades encontradas em aplicações web no momento. O Node.js é muitas vezes usado no desenvolvimento de sistemas para internet, porém os seus recursos não se limitam a apenas isto, com ele é possível escrever linhas de comando e ainda permitir a criação de aplicações nativas para o sistema operacional, tornando possível compilar um mesmo código-fonte para os sistemas Windows, Linux e OS X (MORAES, 2018).

4.1.3 Python

Linguagem de programação de alto nível lançada por Guido van Rossum em 1991, de propósito geral, suporta o paradigma orientado a objetos, procedural, funcional e imperativo. Com o Python é possível desenvolver vários tipos de aplicações, sendo muito utilizado para *machine learning*, processamento de texto e soluções para internet. Foi considerada a linguagem de programação mais desejada

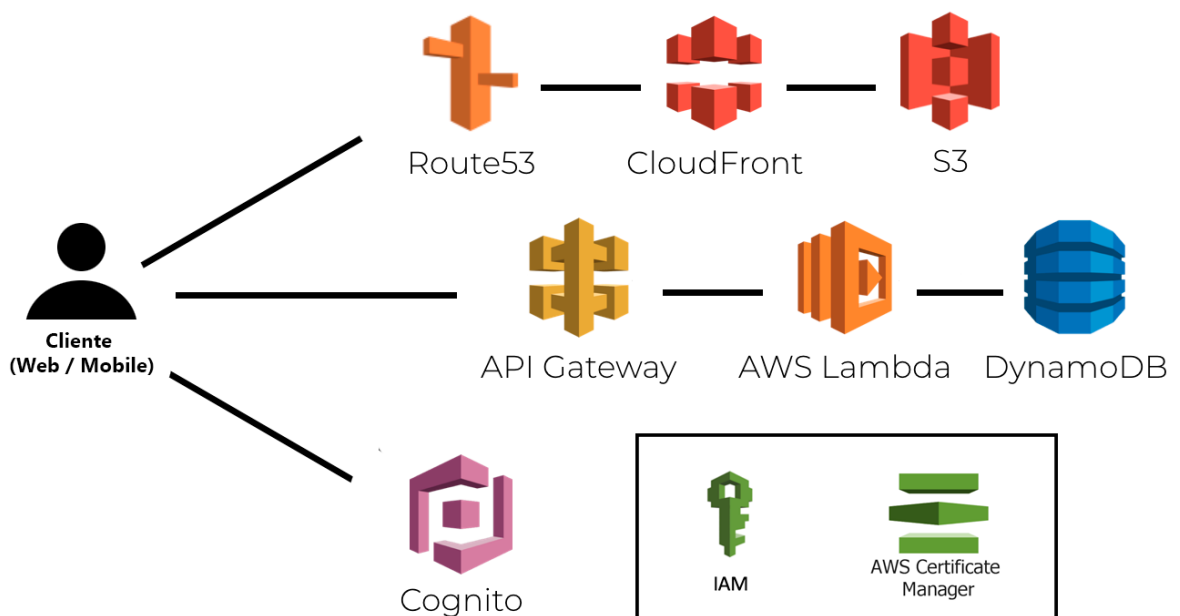
pelos desenvolvedores, além de ser a linguagem com maior crescimento em relação a adesão no último ano em pesquisa realizada pelo Stackoverflow (2019).

4.2 IMPLANTAÇÃO NA AMAZON WEB SERVICES (AWS)

Para tornar o sistema disponível para ser acessado por meio da internet, será utilizado os recursos presentes no ambiente da AWS. Nele ficará hospedado os arquivos da aplicação *web* e será responsável por responder as requisições realizadas pelos usuários, entregando os recursos e dados solicitados.

A Amazon Web Services (AWS) é o provedor de nuvem pública número um do mercado (MENGA, 2018). Diversas empresas, desde startups a grandes corporações estão passando a utiliza-la ao migrar seus sistemas para a nuvem. A Amazon oferece também soluções como monitoramento de serviços, balanceamento de carga (*load-balancing*) e *auto scaling* (WILKINS, 2019). No ambiente da Amazon Web Services existem dezenas de serviços com finalidades diferentes, cada com uma proposta de resolver um obstáculo com a maneira mais apropriada.

Figura 5 - Serviços da AWS utilizados



Fonte: Adaptado pelo autor (2019)

Na Figura 5 é representado pelos principais recursos que serão utilizados pela aplicação desenvolvida neste trabalho. A seguir será descrito de forma resumida qual o papel de cada um desses serviços que serão usados:

Route53: Permite realizar o registro de domínios e o seu gerenciamento, sendo usado para apontar o site para o caminho onde encontra os seus arquivos correspondentes;

S3 (*Simple Storage Service*): Responsável por armazenar os arquivos que serão acessados pelos usuários das aplicações, local onde é enviado e hospedado os arquivos de site e sistemas;

CloudFront: Distribui os arquivos em várias regiões demográficas e pode-se configurar o cache dos recursos para tornar a performance da aplicação com bom desempenho em diferentes localidades;

API Gateway: Neste serviço é configurado as rotas da aplicação que ficarão esperando as requisições para realizar o tratamento de cada solicitação. Usado para fazer o direcionamento das requisições recebidas pelo *web service* e encaminhamento para o serviço que vai processar a informação;

Cognito: Serviço de autenticação da AWS, responsável por gerenciar os usuários das aplicações, permitindo configurar as informações cadastradas para realizar a autenticação, a força da senha e ainda ativar/desativar os usuários da aplicação;

IAM (*Identity and Access Management*): Permite criar políticas de segurança para o uso dos serviços da AWS, restringindo o acesso a conteúdo de maneira apropriada, evitando brechas de segurança nos serviços utilizados;

AWS *Certificate Manager*: Gerencia os certificados dos sites usados no ambiente da Amazon Web Services, utilizado para comprovar que o usuário do domínio configurado fora da AWS tem permissão de usá-lo;

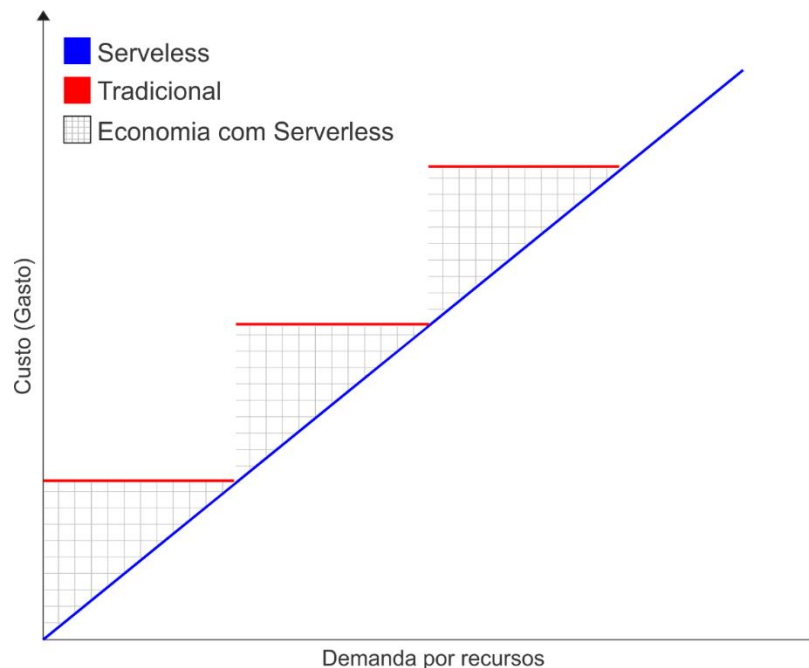
AWS *Lambda*: Ambiente onde fica as lógicas que irão executar as ações solicitadas pelos *microservices*, usando arquitetura *serverless*, o container desse serviço apenas é ativado quando requisitado. Suporta o desenvolvimento em diferentes linguagens e tecnologias como Python, Java, C#, Node.js e Go;

DynamoDB: Solução para banco de dados NoSQL desenvolvida pela Amazon. As tabelas globais criadas no DynamoDB são replicadas em várias regiões da AWS, permitindo conseguir o acesso rápido aos aplicativos, possibilitando atingir o processamento de mais de 10 trilhões de solicitações por dia e suportar mais de 20 milhões de solicitações por segundo. É um banco de dados de valor-chave e documento, utilizado por grandes corporações como Toyota, Airbnb e Samsung.

O Amazon DynamoDB é *serverless*, não sendo necessário *softwares* para fazer a instalação, nem aplicar patches pelo usuário, sendo capaz de aumentar ou diminuir as tabelas automaticamente realizando ajustes conforme a capacidade e o desempenho. O DynamoDB, assim como outros bancos não relacionais (NoSQL), conseguem trabalhar bem com volume grande de dados (*big data*) sendo uma boa opção para aplicações que precisam de alta performance mesmo trabalhando com um número muito alto de dados.

Além dos serviços da AWS descritos, existem ainda dezenas de outros disponíveis na plataforma, mas que não foram usados por terem propostas diferentes do contexto da aplicação desenvolvida para esse trabalho.

Figura 6 - Gráfico de Custo/Demanda de Servidores



Fonte: Elaborado pelo Autor (2019)

Para responder as requisições das aplicações criadas, foi desenvolvida uma aplicação usando a arquitetura de *microservices* e o modelo *serverless* para tornar o custo dos recursos usados pelo servidor da AWS mais eficientes, uma vez que no modelo tradicional é cobrado um valor mesmo sem o uso de todos recursos disponíveis, como é ilustrado na Figura 6.

4.3 PAGESPEED INSIGHT

Para avaliar o desempenho da aplicação web desenvolvida, foi utilizado o PageSpeed Insights (PSI). O PageSpeed Insights é uma ferramenta disponibilizada pelo Google através de uma página na internet, que mede o desempenho de um site informado pelo usuário, analisando os seus recursos utilizados, como ele foi desenvolvido, qual o tempo de resposta que ele necessita para enviar os arquivos do site e o tempo que leva para exibir e permitir a interação do usuário com ele. Ele faz ainda, um diagnóstico do site informado, apontando algumas informações que poderiam ser alteradas para conseguir um melhor desempenho. A pontuação máxima que pode ser conquistada é de 100 pontos, considerando acima de 90 como um site com bom desempenho, de 50 a 90 como moderado e menor que 50, lento.

Além disso o PSI, faz uma auditoria se foi utilizado boas práticas no desenvolvimento web. Ele avalia se o contraste usado está adequado, se está configurado o uso de cache em arquivos estáticos, se as imagens foram comprimidas e se os arquivos de estilo (CSS) e os JavaScript estão sem espaços vazios nos textos desnecessários, visando reduzir o tamanho do arquivo enviado pelo servidor e diminuir o tempo necessário para responder as requisições dos clientes.

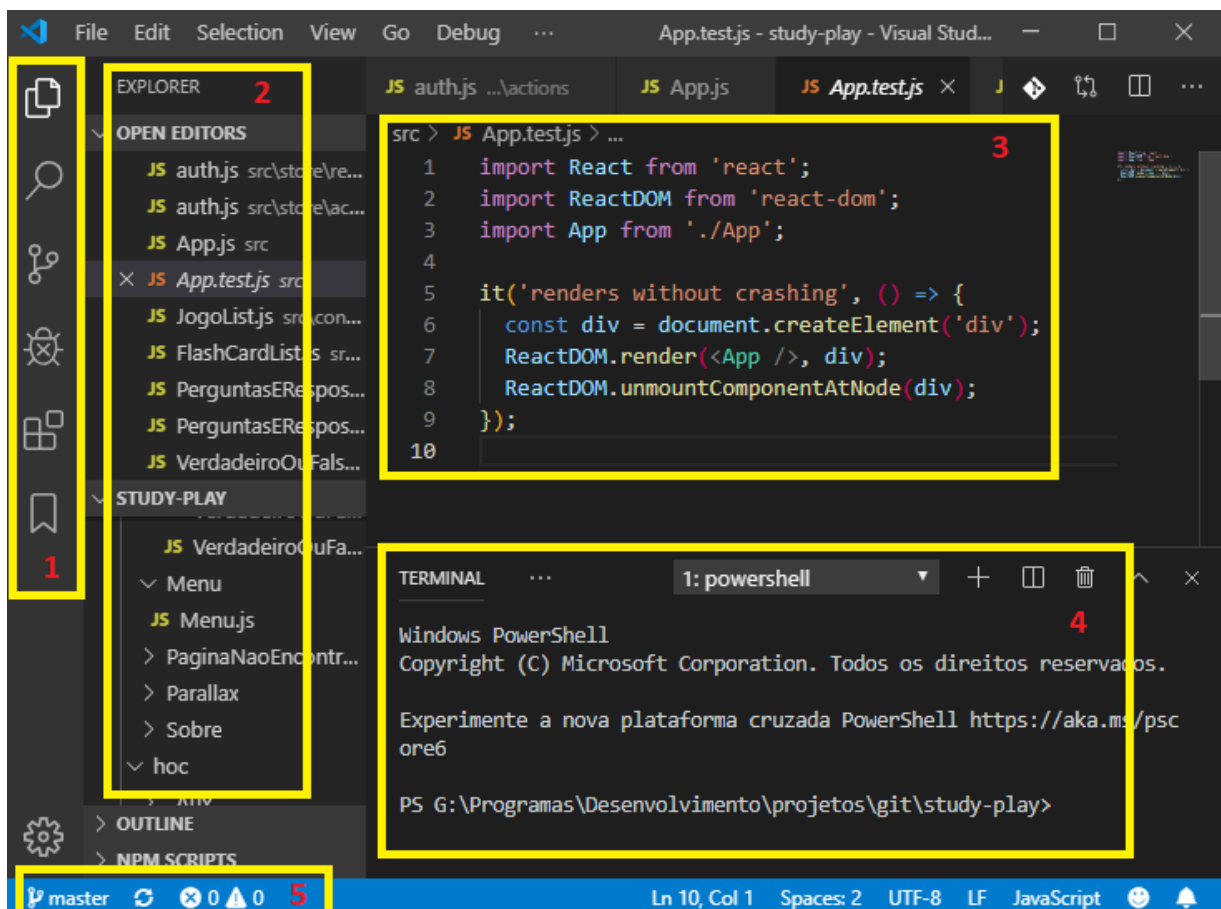
4.4 AMBIENTE E RECURSOS

Ao desenvolver um *software* é possível usar ferramentas que auxiliam na construção, permitindo obter ganho de produtividade, entregar qualidade ao produto e facilitar a manutenção do sistema, uma vez que os sistemas são formados em grande parte por milhares de linhas de código fonte. Para a realização desse trabalho,

o ambiente escolhido para realizar a codificação das aplicações desenvolvidas foi o Visual Studio Code, por se tratar de uma plataforma robusta e a mais usada pelos desenvolvedores no mundo (STACKOVERFLOW, 2019).

O Visual Studio Code, conhecido também como VSCode, é um ambiente de desenvolvimento leve, criado pela Microsoft e pode ser utilizado em diferentes sistemas operacionais como no Windows, Linux e macOS. Entre as linguagens que ele suporta, pode-se destacar o JavaScript, Java, TypeScript, C++, C# entre outras disponíveis. Dentre vários recursos que a ferramenta possui, é possível encontrar funcionalidades como completção de códigos, destaque da sintaxe, além do suporte para um grande número de linguagens de programação e marcação. Esta ferramenta foi desenvolvida usando TypeScript e Electron e seu uso é excelente em aplicações desenvolvidas para a internet ou que utilizam Node.js (ROZENTALS, 2019).

Figura 7 - VSCode



Fonte: Elaborado pelo autor (2019)

A Figura 7 representa uma tela do programa VSCode com um projeto aberto, sendo dividido a imagem em várias partes para facilitar o desenvolvimento de *software*. No pedaço representado pelo retângulo destacado com o número 1, permite o usuário clicar nos ícones para alterar o ambiente de acordo com a ação que irá ser realizada, podendo ser desde exibir o conteúdo dos arquivos para desenvolver o código fonte, até mesmo para carregar os recursos para realizar um *debug* do sistema que está sendo desenvolvido. No item 2, está sendo exibido os arquivos que existem e os que estão abertos no projeto. No item 3 é o código fonte do arquivo selecionado. No bloco do item 4 permite usar um terminal para executar comandos e no item 5 exibe um status informando se existem arquivos para serem atualizados ou enviados no repositório de códigos que a aplicação está usando (como Git e SVN).

Para a criação da aplicação que poderá ser acessada por navegadores para internet, será utilizado recursos do *Progressive Web Apps* (PWAs). Essa tecnologia está em evidencia no desenvolvimento web nos últimos anos, pois o desenvolvimento usando os recursos do PWA, torna as aplicações acessíveis em várias plataformas e ainda possuindo a opção de instalar na tela inicial do dispositivo, sendo uma boa alternativa aos aplicativos nativos para *smartphones* (KINSBRUNER, 2018). Para a criação de uma aplicação PWA pode-se utilizar artifícios populares no desenvolvimento de sites, como HTML, JS e CSS. Através do PWA é possível adicionar recursos em sites que utilizam mecanismos presentes em dispositivos móveis, como a possibilidade de ativar a câmera do aparelho para tirar uma foto e enviar no sistema, ativar o GPS para pegar a localização atual ou ainda possibilitar receber notificações *push*, para ser informado de atualizações do sistema sem a necessidade de ficar fazendo requisições para o servidor em um período contínuo de tempo.

A fim de contornar a limitação do PWA que não permite disponibilizar a aplicação na loja de aplicativos do Google, será utilizado o *Trusted Web Activities* (TWA). Através dessa tecnologia, lançada pelo Google em 2019, o aparelho móvel consegue executar a aplicação com segurança, passando ainda a sensação de estar usando um aplicativo desenvolvido especificamente para aparelhos Android (FIRTMAN, 2019).

4.5 ESPECIFICAÇÕES PARA O DESENVOLVIMENTO

Para a criação de *softwares* é necessário elaborar uma documentação que descreva suas funcionalidades. Esse recurso torna ainda mais importante ao desenvolver grandes aplicações e/ou sistemas complexos. Um dos principais modelos utilizados em empresas de tecnologia para definir e organizar os requisitos necessários para o desenvolvimento de *softwares* é através do sistema de *user stories* (COHN, 2004).

O *user stories* é um artefato usado por profissionais de tecnologia da informação que utilizam a metodologia ágil de desenvolvimento de *software*. Ele especifica uma funcionalidade que deverá estar disponível para um usuário, utilizador do *software* ou sistema. Na criação do *user story* deve escrever a descrição como se tratasse de uma lembrança, refrescando os detalhes como se estivesse em uma conversação e posteriormente documentar como precisará realizar os testes, visando permitir o conhecimento de quando a história foi completada. Importante destacar também que essas histórias de usuários não devem desaparecer no meio do projeto, mesmo em equipes que usam um processo de desenvolvimento ágil como o *Scrum* e o *Extreme Programming* (COHN, 2004). Entre as empresas que utilizam esse recurso durante o desenvolvimento, pode-se destacar a Microsoft, Google, BBC, Nokia, Philips, Xerox, HP, Uol, Abril, Locaweb e a Globo (FERREIRA, 2013).

Os *user stories* criados para o desenvolvimento do sistema e aplicativo podem ser visualizados no apêndice deste trabalho.

4.6 ENGENHARIA DE REQUISITOS

Na engenharia de *software*, considera-se requisito toda a solicitação, necessidade, funcionalidade ou ainda características do sistema. Durante o processo de levantamento de requisitos, é realizado a identificação do que precisa ser implementado na aplicação (ALFF, 2018). Existem dois tipos de requisitos, os “Requisitos funcionais” e os “Requisitos não funcionais”.

4.6.1 Requisitos Funcionais

Os requisitos funcionais são as características, funcionalidades ou necessidades previstas para serem implementadas pelo sistema. Em outras palavras, é a ação esperada para ser realizada por meio do sistema (ALFF, 2018). Os recursos funcionais desenvolvidos para a aplicação deste trabalho são:

- Permitir cadastrar dados para *login*;
- Permitir entrar na área interna com seus dados de acesso;
- Visualizar *ranking* de pontos de usuários;
- Estudar através de *Quiz* com recursos de *gamification*;
- Estudar através de baralho com *flashcards*;
- Estudar através de jogo verdadeiro ou falso e
- Criar um servidor para disponibilizar e salvar os dados.

4.6.2 Requisitos Não Funcionais

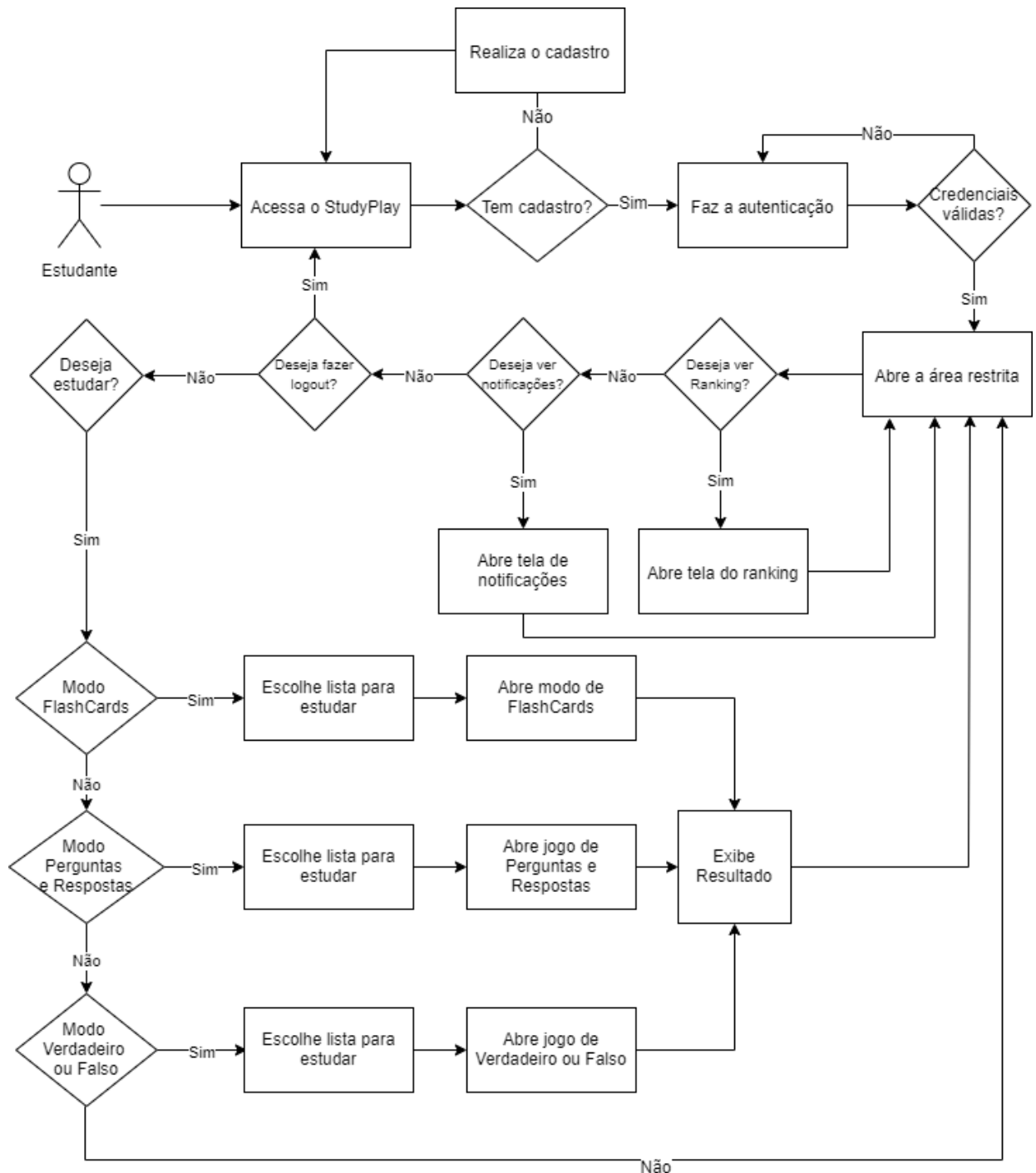
Um requisito não funcional pode ser entendido como a maneira que o sistema deverá implementar suas funcionalidades ou características. Devendo ser mensurável com possibilidade de confirmar se ele está sendo ou não, atendido pelo sistema (ALFF, 2018). Os requisitos não funcionais criados para a aplicação deste trabalho são:

- O *login* será feito através de suas credenciais cadastradas no próprio sistema;
- O sistema utilizará a internet para obter as perguntas, se comunicando com *microservices* implantados nos servidores da AWS;
- A aplicação vai receber as informações através de *web services*, no padrão REST e GraphQL, comunicando através de dados no formato JSON;
- A interface da aplicação terá como referência o Material Design criado pelo Google para ser utilizado em suas aplicações Android e
- O aplicativo será compatível com *smartphones* que possuírem a versão 5.0 do *Android* ou superior.

4.7 FLUXOGRAMA

Um *software* frequentemente utiliza milhares de linhas de código fonte na sua construção, e o entendimento do seu funcionamento a partir dessas instruções é um processo trabalhoso, demorado e ainda não deixa claro todo o funcionamento. Uma das principais maneiras utilizadas para facilitar o entendimento do fluxo dos processos que um sistema é capaz de percorrer, é através do Fluxograma, que é um tipo de diagrama usado para a representação por meio de esquemas e que exibe de forma simplificada o funcionamento de um processo ou algoritmo. Os fluxogramas, usualmente são feitos utilizando formas gráficas que ilustram de maneira compreensível as etapas que os processos podem percorrer, mostrando as possibilidades existentes em cada etapa do fluxo de forma descomplicada para o entendimento dos usuários, que podem ser desde desenvolvedores, até usuários comuns que não conhecem a parte técnica envolvida na construção da aplicação.

Figura 8 - Fluxograma da aplicação do cliente



Fonte: Elaborado pelo autor (2019)

A Figura 8 ilustra o fluxograma criado para a aplicação desenvolvida referente a este trabalho. Será explicado o processo de como funciona as funcionalidades no próximo capítulo.

5. RESULTADOS E DISCUSSÃO

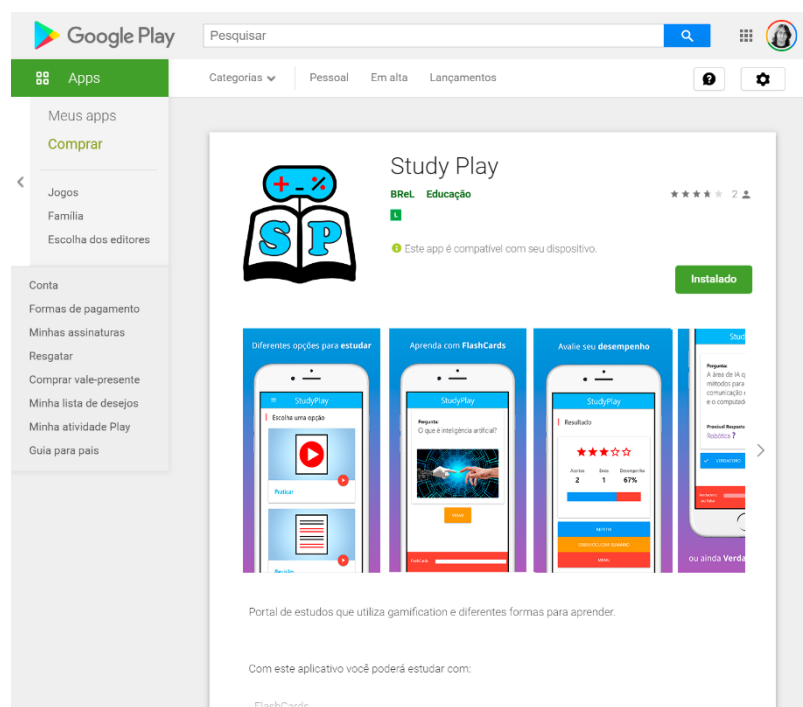
5.1 DISPONIBILIZAÇÃO DAS APLICAÇÕES

Cada aplicação desenvolvida precisou ser implantada em um ambiente mais apropriado com seu propósito. O aplicativo para Android foi publicado nos servidores do Google, através do Google Play, a aplicação web foi publicada no serviço S3 da AWS e os *microservices* que responderão as requisições feitas por essas duas aplicações foi disponibilizado por meio dos recursos presentes na *Amazon Web Services* para a utilização de *serverless*.

5.1.1 Publicação no Google Play

Para realizar a implantação do aplicativo desenvolvido para os usuários que utilizam aparelhos Android no Google Play, foi necessário a criação de artes gráficas com imagens da aplicação e texto informativo com a descrição das funcionalidades presentes no aplicativo.

Figura 9 - Publicação no Google Play



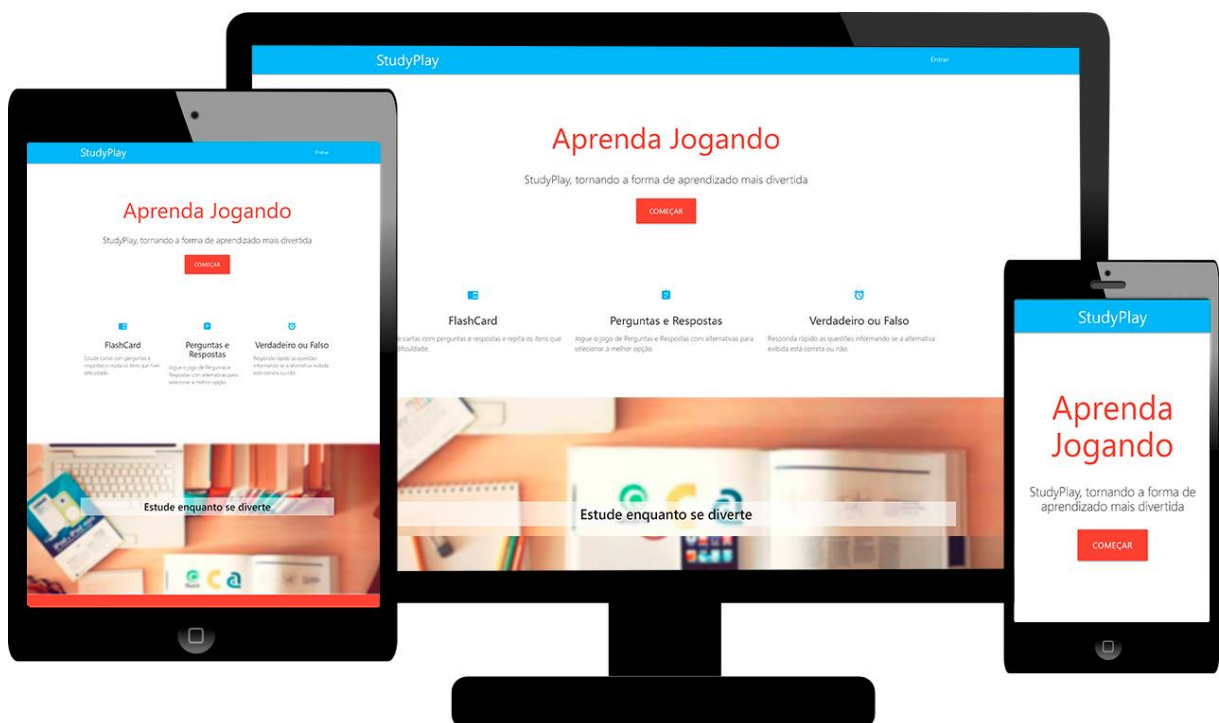
Fonte: Elaborado pelo autor (2019)

Na Figura 9 é apresentado como ficou a página do aplicativo no Google Play para as pessoas que desejam instalar em seu dispositivo e com informações que poderão ajudar novos usuários que pretendem conhecer quais funcionalidades estão presentes na aplicação.

5.1.2 Aplicação para internet

Com o intuito de disponibilizar o conteúdo da aplicação para usuários que não usam o sistema Android ou que preferem não instalar em seus aparelhos (como iPhones, *tablets* e computadores de mesa), foi desenvolvida uma aplicação para a internet, que é compatível com diferentes dispositivos e adequa o conteúdo exibido de acordo com o tamanho da tela. Essa aplicação pode ser utilizada através de computadores, *tablets* e *smartphones* com acesso à internet, independente do sistema operacional do dispositivo.

Figura 10 - Aplicação em diferentes aparelhos



Fonte: Elaborado pelo autor (2019)

Para verificar se o *layout* site ficou adequado para ser utilizado em diferentes aparelhos e está responsivo (consegue se adaptar de acordo com o tamanho da tela), como exibido na Figura 10, foi testado o uso da aplicação em um *smartphone*, *tablet* e em um computador *desktop*, sendo possível o uso dos recursos presentes na aplicação sem perda de informações e nem necessidade de instalação no aparelho, conseguindo dessa maneira, alcançar o objetivo esperado.

Nos testes foram utilizados um *smartphone* Samsung S7 Edge com processador octa-core (quad-core 2.3 GHz M1 Mongoose + quad-core 1.6 GHz Cortex-A53), 4 GB de RAM, tela de 5.5 polegadas e Android 8, um tablet Samsung Galaxy Note 10.1 N8000, com processador quad-core 1.4 GHz Cortex-A9, 2 GB de RAM, tela de 10.1 polegadas e Android 4.0.3 e um computador com processador i5 quad-core de 3.3 GHz com 16 GB de RAM e Windows 10.

5.2 APLICAÇÃO DESENVOLVIDA

O ambiente desenvolvido possui duas partes distintas para os usuários, a área de acesso público que qualquer pessoa tem acesso e uma parte restrita para usuários autenticados. Na área pública que é representada pela página inicial, cadastro de usuários e *login*, o utilizador da aplicação pode conhecer os recursos do StudyPlay permitindo realizar o cadastro ou entrar com seus dados para acessar a área restrita. Dentro da área restrita, que é acessada após a autenticação dos usuários, os usuários podem navegar pelos modos de estudo que o ambiente possui, além de visualizar o *ranking* e a página com as novidades em relação ao cadastro de questões da aplicação.

5.2.1 Tela inicial

Na tela inicial do StudyPlay é exibido informações básicas do que ele apresenta, para usuários conhecerem os recursos que ele possui e decidir se quer criar um cadastro. Ou caso já tenha cadastrado, entrar com suas credenciais para acessar a área restrita para usuários autenticados.

Figura 11 - Tela inicial sem autenticar



Fonte: Elaborado pelo autor (2019)

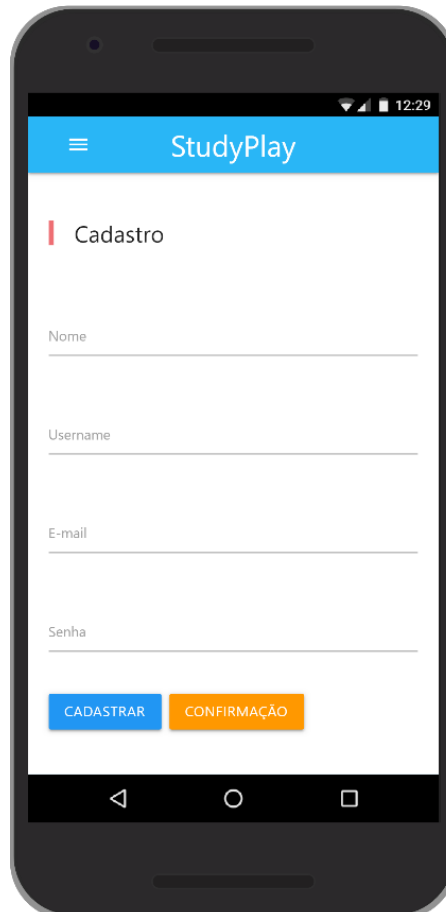
A Figura 11 representa a tela inicial da aplicação quando acessado sem realizar nenhuma autenticação. Foi utilizado um visual minimalista e inserido o efeito Parallax na imagem de fundo.

5.2.2 Tela de cadastro

Na tela de cadastro é solicitado apenas informações básicas como nome, *username*, e-mail e senha para o usuário poder utilizar o sistema. Para realizar o cadastro é necessário que seja preenchido todos campos. Ao clicar no botão para salvar será enviado uma requisição para o *web service* do StudyPlay para verificar se os dados de entrada são válidos, caso positivo ele informará o usuário que o cadastro foi realizado com sucesso através de uma resposta no formato JSON, onde o sistema

irá interpretar e exibir uma mensagem na tela para o cliente da aplicação. Caso tenha algum problema na solicitação do cadastro, será exibida uma mensagem na tela, informado em qual campo precisará modificar para concluir o cadastro.

Figura 12 - Tela de cadastro

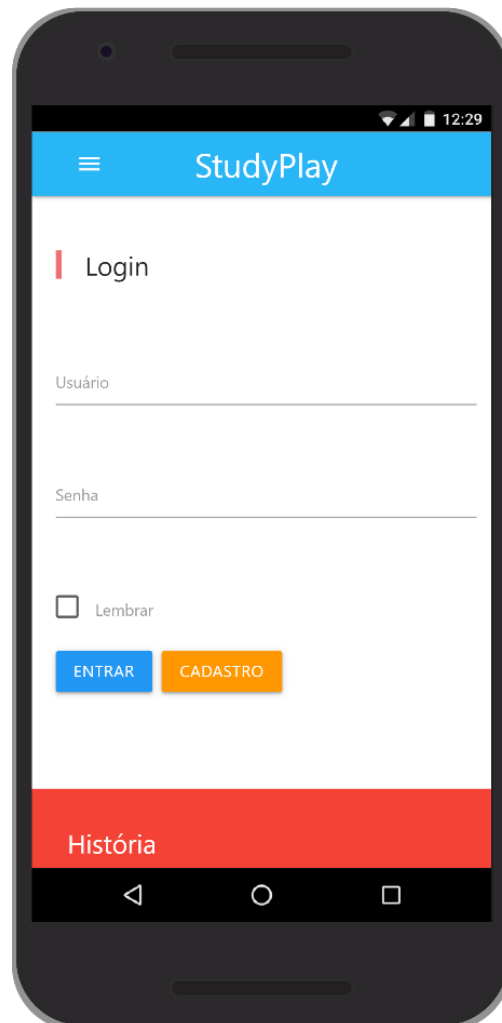


Fonte: Elaborado pelo autor (2019)

5.2.3 Tela de autenticação

Após o cadastro do usuário, ele poderá acessar a tela de autenticação (*login*) para entrar com suas credenciais e caso as informações estejam corretas, acessar a área restrita destinada para usuários autenticados. Para manter o usuário conectado durante o uso do StudyPlay é gerado um *token* de acesso ao realizar a autenticação e armazenado pela aplicação do usuário, este artefato expira a cada 30 minutos e é renovado automaticamente pela aplicação, além disso ele é enviado junto com todas trocas de mensagens para mostrar para o servidor que já foi efetuado o *login* no sistema e assim, permitir o acesso das áreas restritas.

Figura 13 - Tela de autenticação



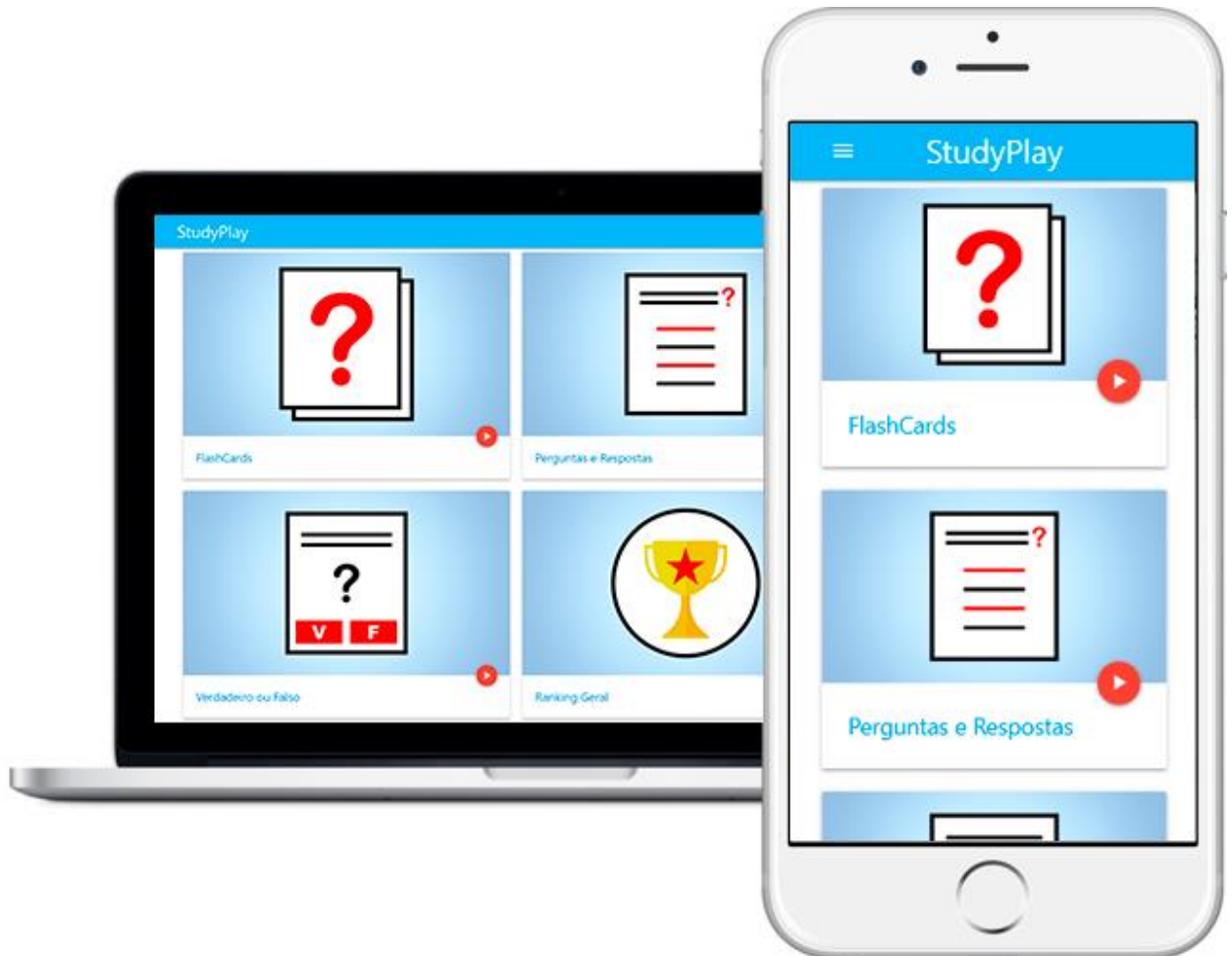
Fonte: Elaborado pelo autor (2019)

Na Figura 13 é apresentado como ficou a tela de autenticação (Login) da aplicação ao ser acessada por um *smartphone*. Nessa tela é feito autenticação e o armazenamento do *token* de acesso para uso das telas restritas.

5.2.4 Tela inicial restrita

Após a autenticação o usuário é encaminhado para a área restrita, onde tem a possibilidade de acessar alguns tipos de estudo, além de opções como *ranking* e lista de atualizações, vide Figura 14.

Figura 14 - Tela inicial restrita

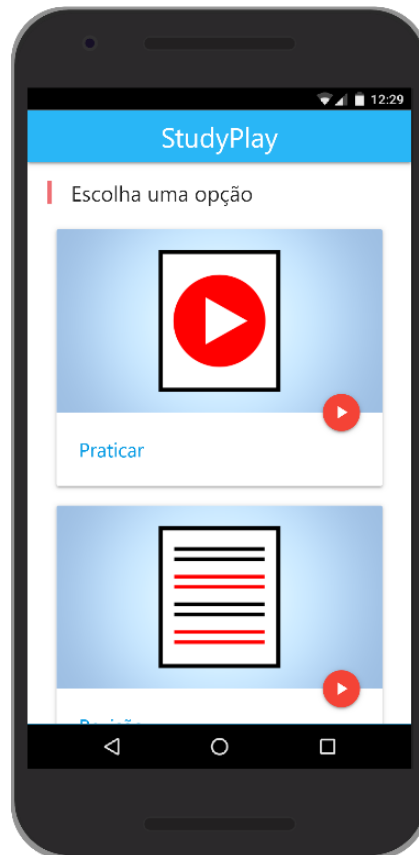


Fonte: Elaborado pelo autor (2019)

5.2.5 Opções de estudo

Ao selecionar um tipo de jogo, entre “*FlashCards*”, “Perguntas e Respostas” e “Verdadeiro ou Falso”, a aplicação carrega uma tela para o usuário selecionar o modo de jogo, como apresentado na Figura 15.

Figura 15 - Opções de estudo



Fonte: Elaborado pelo autor (2019)

Nessa tela é possível escolher se pretende estudar no modo “Praticar”, que apresentará uma lista de perguntas em modo aleatório ou o modo “Revisão” que tem a finalidade do usuário estudar um material mais enxuto, apresentando apenas as perguntas que ele teve mais dificuldade em responder anteriormente, baseado nos erros que ele obteve ao responder o modo “Praticar”, adaptando assim, o aprendizado para cada aluno de forma personalizada.

5.2.6 Tela do modo FlashCard

Quando utilizar o modo *FlashCards* para estudo, é apresentado um cartão com um conceito, como mostrado na Figura 16.

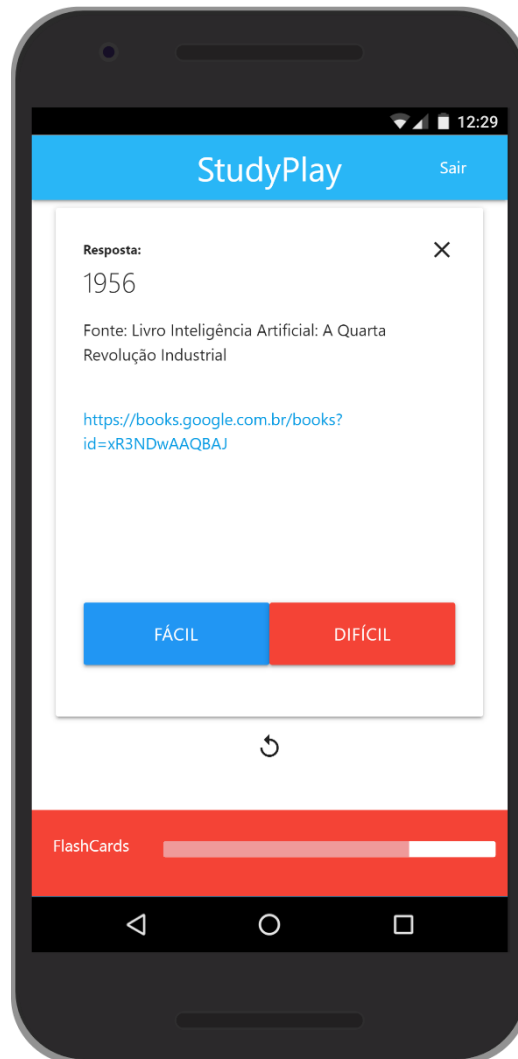
Figura 16 - Tela do modo FlashCard



Fonte: Elaborado pelo autor (2019)

Ao se clicar neste quadro, a carta é virada e a resposta é apresentada, Figura 17. Nessa mesma tela, o usuário poderá selecionar a opção que representa a dificuldade encontrada para se lembrar deste conceito. Caso essa pergunta seja considerada fácil, o usuário deve clicar no botão “FÁCIL” e caso contrário selecionar o botão “DIFÍCIL”.

Figura 17 - Tela exibindo a resposta do FlashCard



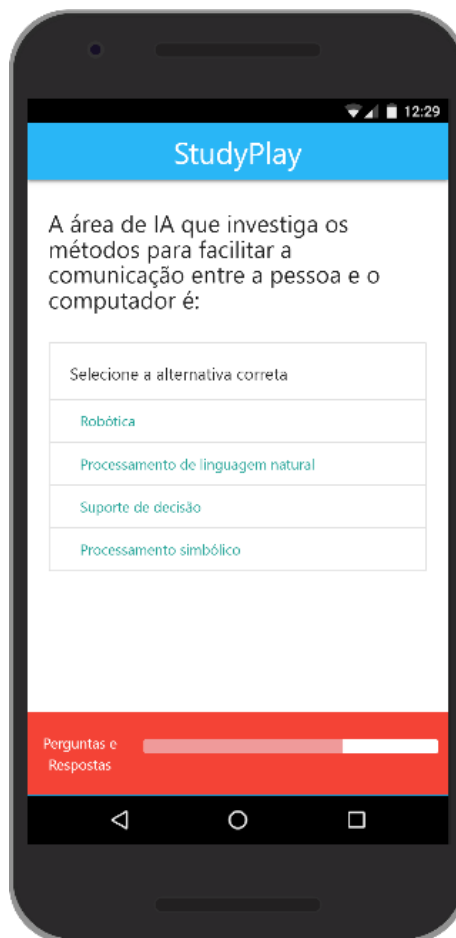
Fonte: Elaborado pelo autor (2019)

Ao escolher entre as opções “FÁCIL” ou “DIFÍCIL” o sistema torna capaz de montar um grupo de questões para estudo direcionado para o usuário, exibindo apenas conceitos considerados por ele como complicado e tornando o estudo posterior adaptado com ênfase nas dificuldades encontradas pelo usuário. Ao utilizar o modo “Revisão”, exibirá as perguntas que foram respondidas mais vezes como “DIFÍCIL” pelo usuário autenticado no sistema.

5.2.7 Tela no modo Perguntas e Respostas

Ao escolher estudar com esse tipo de estudo, será exibida uma pergunta por vez junto com as alternativas para ser indicado a resposta correta. Quando selecionado a opção certa ganha-se um ponto, e em seguida é exibida a próxima questão.

Figura 18 - Tela no modo Perguntas e Respostas



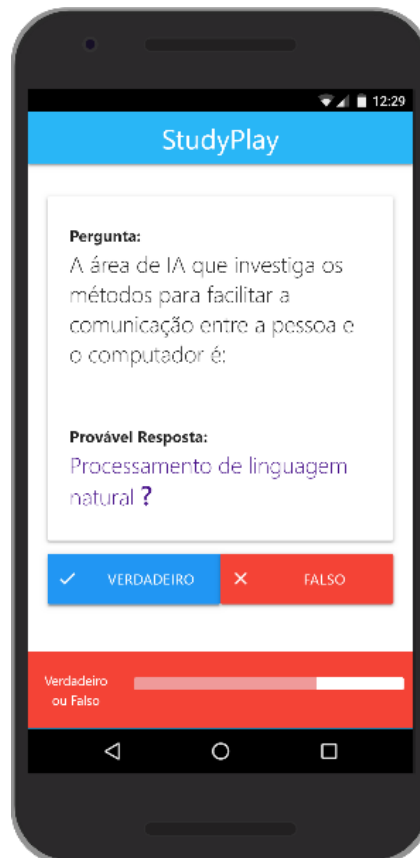
Fonte: Elaborado pelo autor (2019)

Na Figura 18 é apresentado como é exibido a tela desse modo de estudos, exibindo além da pergunta e suas alternativas, uma barra de progresso no rodapé que avança ao responder as questões.

5.2.8 Tela no modo Verdadeiro ou Falso

O modo Verdadeiro ou Falso funciona de forma similar ao modo Perguntas e Respostas, porem só existe duas opções para ser selecionado “Verdadeiro” e “Falso”, como apresentado na Figura 19.

Figura 19 - Tela no modo Verdadeiro ou Falso



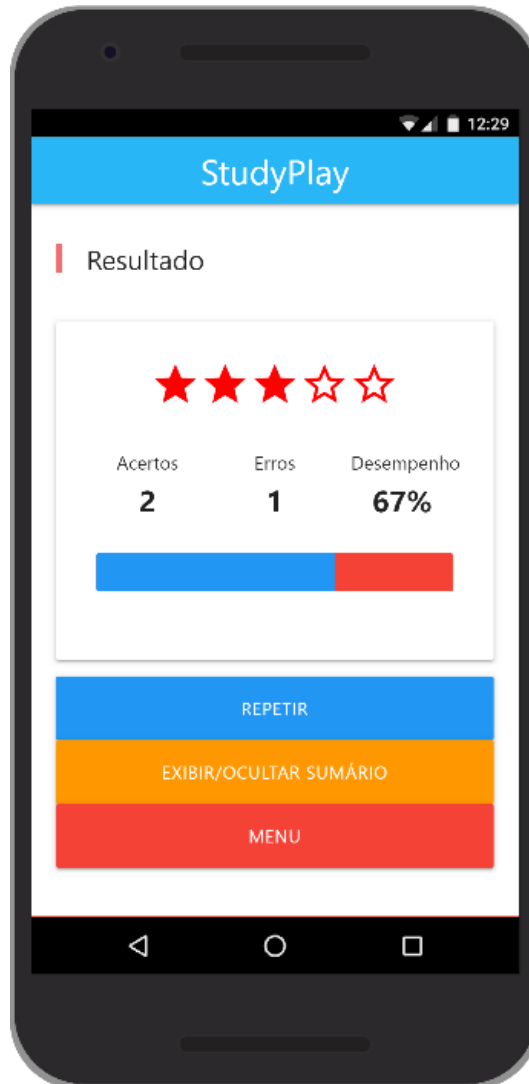
Fonte: Elaborado pelo autor (2019)

Esse tipo de estudos foi desenvolvido utilizando o mesmo banco de questões dos outros modos, com o intuito de tornar o processo de cadastro de questões mais eficiente e possibilitando mais de uma opção de estudo para os alunos que estão utilizando a aplicação.

5.2.9 Tela de resultado

Após o término de cada um dos modos de estudo, é exibida uma tela com todas perguntas e respostas apresentando o desempenho do usuário em relação as questões estudadas, como apresentado na Figura 20.

Figura 20 - Tela de resultado



Fonte: Elaborado pelo autor (2019)

Caso o usuário queira responder as perguntas novamente, pode-se utilizar o botão "Repetir". Para voltar ao menu principal pode clicar em "Menu" e para exibir as perguntas e respostas corretas como apresentado na Figura 21, pode clicar em "Exibir/Ocultar Sumário".

Na tela de resultado foi utilizado estrela como representação gráfica para pontuar o desempenho do usuário. Esse recurso é utilizado em sistemas que usam *gamification* com o intuito de estimular o usuário a conquistar todas estrelas para ter a sensação de dever cumprido.

Figura 21 - Exibindo perguntas e respostas do sumário

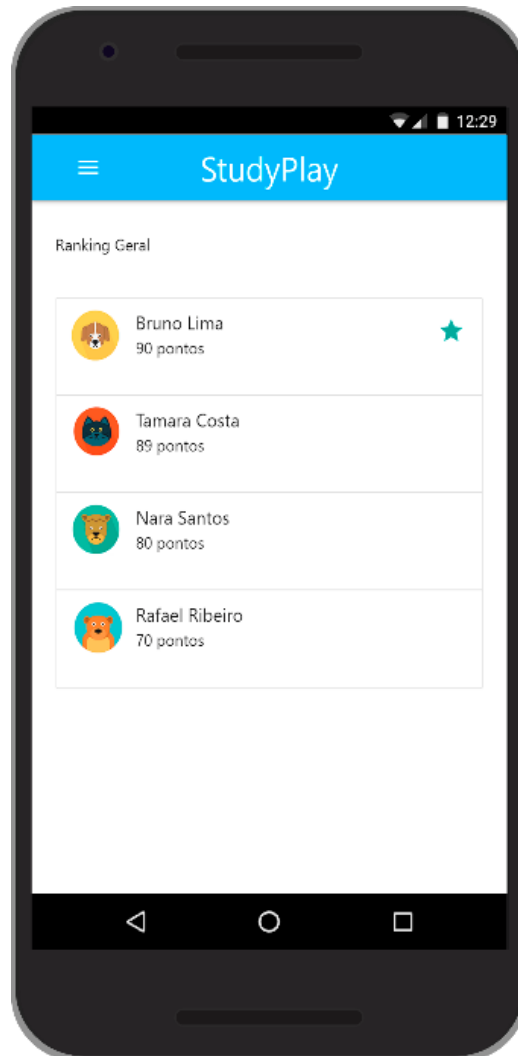


Fonte: Elaborado pelo autor (2019)

5.2.10 Tela exibindo o ranking

Na tela de exibição do *ranking*, apresentada na Figura 22, pode-se ver como está o desempenho do usuário em relação aos outros, para assim estimular a competição entre eles.

Figura 22 - Tela do Ranking



Fonte: Elaborado pelo autor (2019)

Esse é um recurso presente em *games* que além de medir o desempenho em relação aos outros competidores, almeja aumentar o desejo do usuário de melhorar o desempenho, realizando as atividades novamente, buscando notas maiores para subir posições no *ranking*.

A pontuação de cada usuário é calculada baseada na quantidade de acertos e no tempo que ele gastou para responder as perguntas no modo “Praticar”, ficando no topo do *ranking* os usuários que conseguiram ter mais acertos em menos tempo, conseguindo assim, as maiores pontuações.

Para possibilitar utilizar o mesmo cadastro de questões para os três tipos de jogos, foi usado uma estratégia de exibição específica para cada tipo de jogo. No

“FlashCard” utiliza a pergunta e resposta cadastrada, ignorando as alternativas erradas, no “Verdadeiro ou Falso”, utiliza a pergunta e uma alternativa aleatória, podendo ser considerada como “Verdadeiro” caso a alternativa exibida seja a resposta correta, ou como “Falso”, caso a alternativa seja a errada, e por último, no jogo tipo “Pergunta e Resposta” é exibido a pergunta, junto com todas as alternativas, pontuando apenas quando o usuário selecionar a alternativa correta. Dessa maneira, a forma que as aplicações foram projetadas por esse trabalho tornou possível o estudo a partir de um cadastro unificado de questões, permitindo usar uma mesma pergunta em diferentes modos de estudo. Além disso, é possível alterar o tema abordado apenas modificando as perguntas cadastradas no banco de dados, permitindo utilizar as aplicações deste trabalho para servir como apoio no estudo de conceitos além de inteligência artificial.

5.3 SERVIDOR DAS APLICAÇÕES

Para a aplicação que responderá as solicitações da aplicação web ou do aplicativo, foi necessário escolher a quantidade de memória RAM disponível para o container que irá executar o *microservice* para retornar a lista de questões cadastradas no banco de dados. Com o intuito de escolher o ambiente com melhor custo benefício, foram realizados testes escrevendo a mesma lógica usando Node.js versão 10 e Python na versão 3.7 e executado quatro vezes cada configuração para observar o tempo de resposta de cada execução. Foi realizado o teste mais de uma vez, pelo fato do tempo de execução no primeiro momento ter um valor muito discrepante em relação a segunda vez, conseguindo visualizar que o tempo necessário para executar após a primeira tentativa permanece um valor mais estável na maioria das vezes, escolhendo assim executar quatro vezes para posteriormente calcular uma média do tempo de execução para cada quantidade de memória RAM utilizada pelo container.

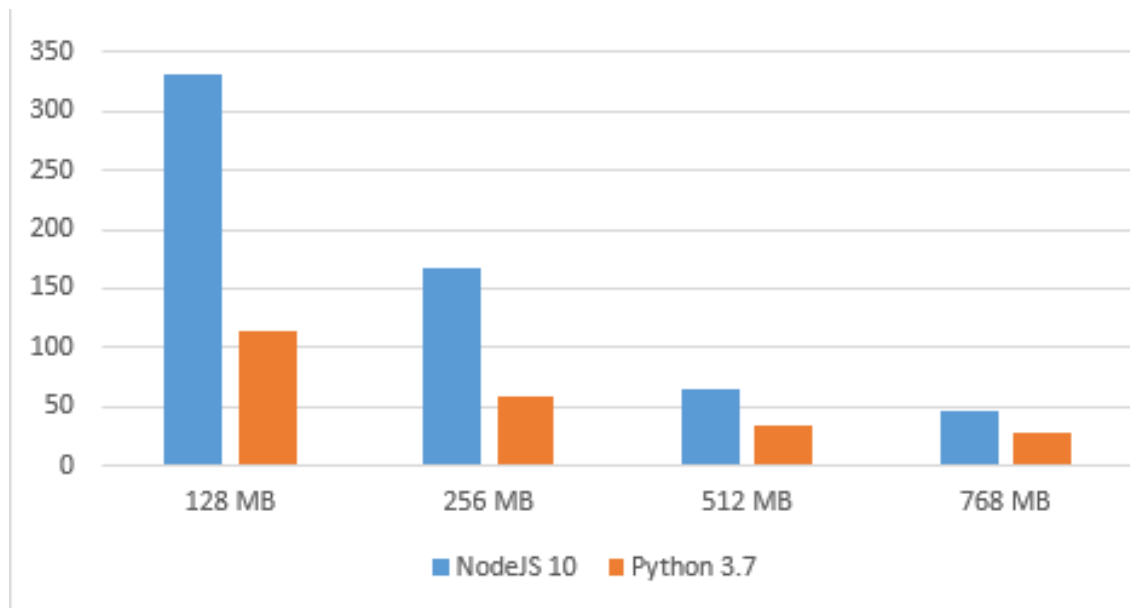
Tabela 2 - Tempo de resposta ao executar lógica (ms)

	128MB	256MB	512MB	768MB
Node.js 10	1073	365	167	109
	111	51	47	30
	94	216	22	26
	49	39	25	23
Python 3.7	242	135	56	49
	64	31	23	8
	77	34	9	9
	71	35	50	42

Fonte: Elaborado pelo autor (2019)

Na Tabela 2 mostra que a primeira execução de cada configuração teve um tempo de resposta maior que nas outras vezes que foram executadas, mesmo usando a mesma configuração. Isso ocorre devido fatores como a utilização de cache nas próximas requisições, que diminui a necessidade de realizar o mesmo processamento em um tempo de execução curto, como a resolução do *Domain Name System* (DNS) do endereço informado.

Figura 23 - Gráfico com a média do tempo de execução (ms)



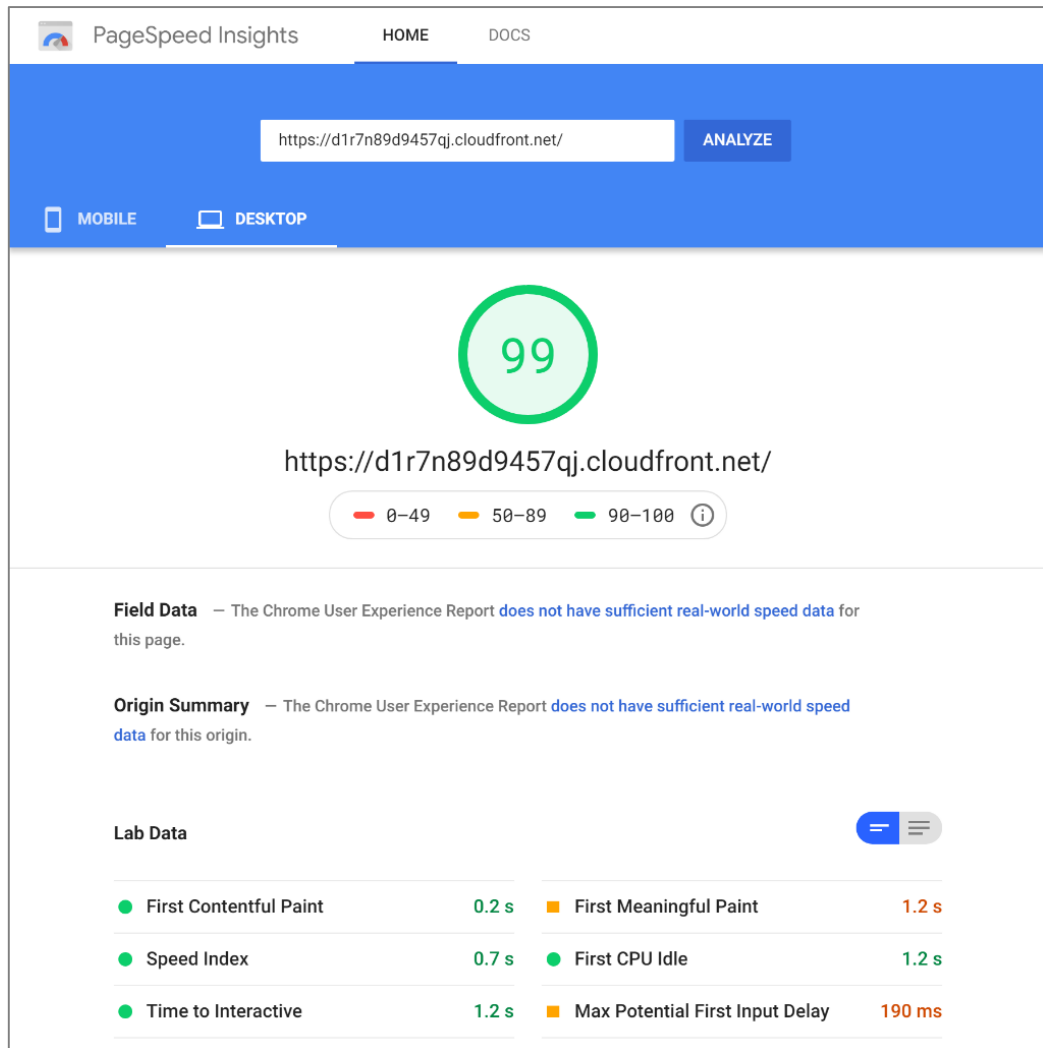
Fonte: Elaborado pelo autor (2019)

No gráfico da Figura 23 mostra que o tempo de execução do container usando Python obteve um melhor desempenho em relação com o Node.js, conseguindo obter um bom tempo de execução, mesmo usando um container com pouca memória RAM. Com essas informações, foi escolhido usar um container de 128 MB utilizando Python na versão 3.7 para ficar responsável para executar o código desse *microservice*, pelo fato dele ter conseguido um bom custo benefício, uma vez que o servidor na nuvem no modelo *serverless*, cobra pela utilização do *microservice* de acordo com a quantidade de recurso utilizado (memória RAM) e do tempo de execução gasto por ele.

5.4 DESEMPENHO

Ao utilizar o site PageSpeed Insights, criado pelo Google para avaliar a aplicação em relação ao tempo de resposta do servidor para realizar a entrega dos conteúdos ao cliente e o uso de recursos que melhoram a usabilidade do usuário, a nota alcançada pela aplicação desenvolvida nesse trabalho foi de 99 pontos, de 100 possíveis.

Figura 24 - Desempenho no PageSpeed Insights



Fonte: Elaborado pelo autor (2020)

O desempenho de 99 pontos no PageSpeed (2020), apresentado na Figura 24, é considerado bom, uma vez que nos testes realizados o site respondeu rápido as requisições e que foi utilizado boas práticas no seu desenvolvimento, apontado assim que o site está otimizado e em conformidade com o esperado em sites considerados de bom desempenho por esta ferramenta.

Como parâmetro de comparação, foi analisado o desempenho do sistema Virtual-IF do IFTM no PageSpeed e ele obteve como resultado apenas 73 pontos, nota considerada moderada pelo site, recomendando realizar uma série de ajustes para melhorar a qualidade da aplicação testada. Dessa forma, a aplicação desenvolvida conseguiu uma pontuação 26% melhor que do sistema atual do IFTM.

Com esses dados obtidos é possível afirmar que a aplicação desenvolvida nesse trabalho apresenta um bom desempenho, foi construída usando boas práticas de programação e que os servidores utilizados para hospedar possuem uma boa performance, atendendo as necessidades de uma aplicação para internet.

6. CONCLUSÃO

Com a utilização de uma infraestrutura de serviços na nuvem, além de atender um requisito do governo para instituições públicas, possibilitou o uso de uma série de ferramentas disponíveis neste ambiente. Ainda, foi possível utilizar os recursos do servidor de forma mais eficiente, possibilitando a configuração dos computadores que irão hospedar a aplicação de forma mais apropriada e sendo capaz de alterar o poder computacional do servidor com mais agilidade.

Ao se optar em usar o modelo *serverless* e a arquitetura baseada em *microservices*, apesar de aumentar a complexidade durante o desenvolvimento, tornou possível configurar os recursos utilizados no servidor na medida que cada funcionalidade precisava separadamente e pagar apenas pelo o que está sendo usado pelos usuários. Não necessitando desembolsar pela disponibilidade de recursos não utilizados, possibilitando assim eliminar gastos desnecessários, que seriam cobrados caso tivesse adotado o modelo tradicional. Além disso, possibilitou aumentar a capacidade computacional do servidor para cada funcionalidade separadamente, tornando possível, inclusive, configurar o servidor para replicar a aplicação em momentos de alta demanda.

Diante disso, as aplicações desenvolvidas conseguiram permitir o estudo por meio de dispositivos móveis usando recursos da *gamification* e ter a flexibilidade de auxiliar no ensino em diferentes temas. Sendo possível, também, a implantação dessas aplicações no servidor e disponibilizar para os estudantes por meio de um modelo que cobra apenas quando é usado efetivamente o recurso. Não desperdiçando gastos com funcionalidades não utilizadas pelos usuários.

Com as aplicações desenvolvidas é esperado contribuir com instituições que necessitam de opções que utilizam *mobile learning* e *gamification* em seu ensino. E ajudar no aprendizado de pessoas que gostariam de estudar por meio de seus dispositivos. Possibilitando assim, aumentar a satisfação no aprendizado ao usar aparelhos que já fazem parte do cotidiano das pessoas e proporcionar mais um meio de estudo que auxilia na fixação do conhecimento.

Como trabalho futuro, tem a opção de testar as aplicações desenvolvidas com alunos e também a possibilidade de comparar mais detalhadamente como seria os gastos ao adotar o modelo *serverless* e o tradicional em aplicações relativas por um período de tempo.

REFERÊNCIAS

ADZIC, G.; CHATLEY, R. **Serverless computing: economic and architectural impact**. In: Proceedings of the 2017 11th Joint Meeting on Foundations of Software Engineering - ESEC/FSE 2017, New York, USA: ACM Press, 2017.

AL-EMRAN, M.; ELSHERIF, H. M.; SHAALAN, K. **Investigating attitudes towards the use of mobile learning in higher education**. Computers in Human Behavior. Elsevier, 2016.

ALFF, F. **O Que São Requisitos Funcionais E Requisitos Não Funcionais?** 2018. Disponível em <<https://analisederequisitos.com.br/requisitos-funcionais-e-requisitos-nao-funcionais-o-que-sao/>>. Acesso em 10 jun. 2019.

ALVES, F. **Gamification: Como criar experiências de aprendizagem engajadoras**. DVS Editora, 2015.

AVRAM, A. **Is REST Successful in the Enterprise?** 2011. Disponível em <<https://www.infoq.com/news/2011/06/Is-REST-Successful/>>. Acesso em 10 jun. 2019.

BRAGA, MCG; OBREGON, RFA. Gamificação: Estratégia para processos de aprendizagem. In: Congresso Nacional de Ambientes Hiperfídia para Aprendizagem. 2015.

BRASIL. Ministério da Economia. Instrução Normativa nº 1, de 4 de abril de 2019. Dispõe sobre o processo de contratação de soluções de Tecnologia da Informação e Comunicação - TIC pelos órgãos e entidades integrantes do Sistema de Administração dos Recursos de Tecnologia da Informação - SISP do Poder Executivo Federal. **Diário Oficial da União**: seção 1, Brasília, DF, n. 66, p. 54, 5 abr. 2019.

CARLOS, L. Q. F. Desenvolvendo seu primeiro aplicativo Android. São Paulo: Novatec, 2013.

COHN, M. **User Stories Applied: For Agile Software Development**. Addison-Wesley, 2004.

CORDONI, R. **Guia Geração da Internet**. ON LINE EDITORA, 2016.

CREMONTTI FILHO, Jorge Luiz. **O uso da aprendizagem móvel e técnicas de gamificação como suporte ao ensino de matrizes**. Boa Vista, 2016. Dissertação (mestrado) - UFRR, Universidade Federal de Roraima, Programa de Pós-graduação Mestrado Profissional em Matemática em Rede Nacional, 2016.

CROMPTON, Helen; BURKE, Diane. **The use of mobile learning in higher education: A systematic review**. Computers & Education, Volume 123, 2018, Pag. 53-64, ISSN 0360-1315, <https://doi.org/10.1016/j.compedu.2018.04.007>.

FERREIRA, C. R. **Conhecendo o Scrum**. 2013. Disponível em <<https://www.devmedia.com.br/conhecendo-o-scrum/29129>>. Acesso em 10 jun. 2019.

FIRTMAN, M. **Google Play Store now open for Progressive Web Apps**. 2019. Disponível em <<https://medium.com/@firt/google-play-store-now-open-for-progressive-web-apps-ec6f3c6ff3cc>>. Acesso em 10 jun. 2019.

FRANQUIA, Franco. **Geração Alpha e o futuro da educação**. 2015. Disponível em <<https://tutores.com.br/blog/geracao-alpha-e-o-futuro-da-educacao/>>. Acesso em 18 fev. 2019.

GLAUBER, N. **Dominando o Android com Kotlin**. NOVATEC, 2019.

GOMES, D. A. **Web Services SOAP em Java - 2ª Edição: Guia prático para o desenvolvimento de web services em Java**. NOVATEC, 2014.

GOOGLE. **Painel de distribuição - Desenvolvedores Android**. 2019. Disponível em <<https://developer.android.com/about/dashboards?hl=pt-br>>. Acesso em 20 fev. 2020.

GUPTA, M. **Serverless Architectures with AWS: Discover how you can migrate from traditional deployments to serverless architectures with AWS**. Packt Publishing, 2018.

HANASHIRO, A. **GraphQL: A revolucionária linguagem de consulta e manipulação de dados para APIs**. Casa do Código, 2019.

HART-MATYAS, M. et al. Twelve tips for medical students to establish a collaborative flashcard project. **Medical Teacher**, v. 41, n. 5, p. 505–509, 4 maio 2019.

IBGE. PNAD Contínua TIC 2017: Internet chega a três em cada quatro domicílios do país. 2018. Disponível em

<<https://agenciadenoticias.ibge.gov.br/agencia-sala-de-imprensa/2013-agencia-de-noticias/releases/23445-pnad-continua-tic-2017-internet-chega-a-tres-em-cada-quatro-domicilios-do-pais>>. Acesso em 10 jun. 2019.

IFTM. Indicadores do IFTM. 2020. Disponível em <<http://indicadores.iftm.edu.br/>>. Acesso em 20 fev. 2020.

INDALÉCIO, Anderson Bençal; RIBEIRO, Maria da Graça Martins. Gerações Ze Alfa: os novos desafios para a educação contemporânea. **Revista UNIFEV: Ciência & Tecnologia**, v. 2, p. 137-148, 2017.

JÚNIOR, C. F. A.; SILVEIRA, I. F. **Tablets no Ensino Fundamental e Médio: princípios e aplicações.** Terracota Editora, 2016.

KENSKI, Vani Moreira. **Tecnologias e Ensino Presencial e a Distância.** Campinas. Papyrus Editora, 2008.

KHAN, B. H. **Managing E-learning: Design, Delivery, Implementation, and Evaluation.** Information Science Pub., 2005.

KINSBRUNER, Eran. **Why Continuous Testing Will Be the Make or Break for Progressive Web Apps.** Database and Network Journal. Academic OneFile, 2018. Disponível em <<http://link.galegroup.com/apps/doc/A554181146/AONE?u=capes&sid=AONE&xid=a0403b9a>>. Acesso em 19 fev. 2019.

LEAL, Bárbara. **Entenda a Aprendizagem Adaptativa.** 2018. Disponível em <<https://www.edools.com/aprendizagem-adaptativa/>>. Acesso em 19 fev. 2019.

LECHETA, R. R. **Web services RESTful: Aprenda a criar web services RESTful em Java na nuvem do Google.** NOVATEC, 2015.

LECHETA, Ricardo R. **Google Android-3ª Edição: Aprenda a criar aplicações para dispositivos móveis com o Android SDK.** Novatec Editora, 2013.

LEMONS, Heverton de; FERNANDES, Anita Maria da Rocha. (2015). **Pesquisa e mapeamento de Shells, Frameworks e jogos para auxiliar no ensino de**

Inteligência Artificial. Revista Eletrônica Argentina-brasil De Tecnologias Da Informação E Da Comunicação, v. 1(n. 2).

LINS, L. **Chamados para Liderar: Um Guia Prático para Líderes de Adolescentes e Jovens.** CPAD, 2018.

LOPES, Aurea. **10 vantagens da aprendizagem adaptativa.** 2016. Disponível em <<http://www.aredo.inf.br/4111-2/>>. Acesso em 18 fev. 2019.

LORENZONI, Marcela. **Gamificação: o que é e como pode transformar a aprendizagem.** 2016. Disponível em <<http://info.geekie.com.br/gamificacao/>>. Acesso em 18 fev. 2019.

MATEUS, Bruno Gois; MARTINEZ, Matias. **An Empirical Study on Quality of Android Applications written in Kotlin language.** arXiv preprint arXiv:1808.00025, 2018.

MENDONÇA, Bruno. **Como funciona o Mobile Learning?** 2016. Disponível em <<http://www.edools.com/mobile-learning/>>. Acesso em 18 fev. 2019.

MENGA, J. **Docker on Amazon Web Services: Build, deploy, and manage your container applications at scale.** Packt Publishing, 2018.

MOLINARI, W. **Desconstruindo a Web: As tecnologias por trás de uma requisição.** Casa do Código, 2016.

MONTEIRO, Angélica; MOREIRA, J. António; LENCAS, José Alberto. **Blended (e)Learning na Sociedade Digital.** Whitebooks. 2015.

MORAES, W. B. **Construindo aplicações com NodeJS.** Novatec Editora, 2018.

PAGESPEED INSIGHTS. **Make your web pages fast on all devices.** 2019. Disponível em <<https://developers.google.com/speed/pagespeed/insights/>>. Acesso em 11 dez. 2019.

PORCELLO, E.; BANKS, A. **Introdução ao GraphQL: Busca de dados com abordagem declarativa para aplicações web modernas.** NOVATEC, 2018.

PRESSMAN, R.; MAXIM, B. **Engenharia de Software - 8ª Edição.** McGraw Hill Brasil, 2016.

RAJPUT, D. **Hands-On Microservices -- Monitoring and Testing: A performance engineer's guide to the continuous testing and monitoring of microservices.** Packt Publishing, 2018.

RAJPUT, D. **Designing Applications with Spring Boot 2.2 and React JS: Step-by-step guide to design and develop intuitive full stack web applications.** BPB Publications, 2019.

REIS, Gustavo Henrique da Rocha; BARRERE, Eduardo. **CA Learning - Recomendação Híbrida de Conteúdos Educacionais.** In: IV Congresso Brasileiro de Informática na Educação, 2015, Maceió. IV Congresso Brasileiro de Informática na Educação - CBIE 2015, 2015.

RIBEIRO, Gonzaga. **A gamificação gamificada: desenvolvimento de um curso para capacitação de docentes.** 2017. Disponível em: <<https://periodicos.fclar.unesp.br/rpge/article/view/10074/7165>>. Acesso em 16 fev. 2019.

RODRIGUES, José Roberto Alves. **iQuiz: ambiente de autoria para avaliação do aprendizado no Moodle.** 2014. Dissertação (Mestrado em Ciência da Computação) - Instituto de Matemática e Estatística, Universidade de São Paulo, São Paulo, 2014. doi:10.11606/D.45.2014.tde-23012015-095520.

ROZENTALS, N. **Mastering TypeScript 3: Build enterprise-ready, industrial-strength web applications using TypeScript 3 and modern frameworks, 3rd Edition.** Packt Publishing, 2019.

SOLIS, C. **Ecos da alma brasileira #3.** 2018. Instituto Civitas Solis. 3ª ed.

STACKOVERFLOW. **Developer Survey Results.** 2019. Disponível em <<https://insights.stackoverflow.com/survey/2019>>. Acesso em 15 jun. 2019.

STATCOUNTER. **Mobile Operating System Market Share Worldwide – 2017 - 2019.** 2019. Disponível em <<https://gs.statcounter.com/os-market-share/mobile/worldwide/#yearly-2017-2019>>. Acesso em 10 jun. 2019.

STATEOFJS. **The State of JavaScript 2018_ Front-end Frameworks - Overview.** 2019. Disponível em <<https://2018.stateofjs.com/front-end-frameworks/overview/>>. Acesso em 10 jun. 2019.

STREIT, E. **Era digital e crise na educação**. Editora Appris, 2015.

TOLOMEI, B. A Gamificação como Estratégia de Engajamento e Motivação na Educação. **EaD em FOCO**, v. 7, 2017.

TONÉIS, C. N. **Os games na sala de aula: Games na educação ou a gamificação da educação**. Bookess, 2017.

VARGAS, Daiana de. **O processo de aprendizagem e avaliação através de QUIZ**. 2017. Artigo (Especialização) – Curso de Docência na Educação Profissional, Universidade do Vale do Taquari - Univates, Lajeado, 22 set. 2017. Disponível em: <http://hdl.handle.net/10737/2038>

VÁZQUEZ-INGELMO, Andrea; CRUZ-BENITO, Juan; GARCÍA-PEÑALVO, Francisco J. **Improving the OEEU's data-driven technological ecosystem's interoperability with GraphQL**. In: Proceedings of the 5th International Conference on Technological Ecosystems for Enhancing Multiculturality. 2017. p. 1-8.

WEST, Mark. **Turning on mobile learning: global themes**. UNESCO, 2012. <https://unesdoc.unesco.org/ark:/48223/pf0000216451>

WILDEN, S. **Mobile Learning**. Oxford University Press, 2017.

WILKINS, M. **Learning Amazon Web Services (AWS): A Hands-On Guide to the Fundamentals of AWS Cloud**. Pearson Education, 2019.

WOLFF, E. **Microservices: Flexible Software Architecture**. Pearson Education, 2016.

GLOSSÁRIO

Android – Sistema operacional para smartphones desenvolvido pelo Google

App – Forma resumida de escrever “aplicativo”

Apache – Servidor que interpreta os dados de uma linguagem de programação para ser exibido a informação na internet

Back-end – Parte do desenvolvimento que fica com os códigos no servidor

FlashCard – São pequenos cartões utilizados para quem quer memorizar melhor alguns assuntos

Framework – Conjunto de códigos visando resolver problemas recorrentes e ajudando no desenvolvimento de um recurso computacional

Front-end – Parte do desenvolvimento que deixa os códigos no cliente

JSX – Sintaxe estendida do JavaScript, utilizada em aplicações React JS

IDE – Ambiente de desenvolvimento de softwares

Ionic – Plataforma de desenvolvimento de aplicativos para desenvolvedores web

iOS – Sistema operacional presente em aparelhos da Apple, como iPhones e iPads

Joomla – Sistema gerenciador de conteúdos para internet

JSON – Formato compacto de arquivo para troca de mensagens entre sistemas que utiliza texto legível para humanos, no formato atributo-valor

Heroku – Plataforma de serviço na nuvem

Kahoot – Plataforma baseado em jogo para estudo

Kotlin – Linguagem de programação

Moodle – Plataforma para criar e gerenciar um ambiente de estudos

Netbeans – Software usado para desenvolver programas

Quiz – Jogo de perguntas e respostas

SDK – Kit de desenvolvimento para desenvolvimento de software e/ou aplicativos

Shell – Interface para usuário acessar serviços do sistema operacional

Storyboard – Representação gráfica que permite contar uma história

VSCode – Programa utilizado para desenvolver programas para diversos tipos de linguagens

Web Service – É uma solução utilizada na integração de sistemas e na comunicação entre aplicações diferentes

APÊNDICE A – CARTÕES DE HISTÓRIA

Segue os cartões de histórias usados pelo aplicativo, utilizados como referência para a criação do sistema e como requisito funcionais.

Quadro 2 – Cartão de história: Cadastro de novo usuário

Nº cartão	1
Título	Cadastro de novo usuário
História	O usuário precisa realizar um cadastro para poder utilizar o sistema
Testes de sucesso	<ul style="list-style-type: none"> • O aluno preenche todos campos obrigatórios e ao enviar para o servidor, recebe uma resposta que foi cadastrado
Testes de erro	<ul style="list-style-type: none"> • Exibir mensagem: É necessário preencher todos dados obrigatórios para realizar o cadastro

Fonte: Elaborado pelo autor, 2019

Quadro 3 – Cartão de história: Login de usuário já cadastrado

Nº cartão	2
Título	Login de usuário já cadastrado
História	O usuário precisa informar seus dados de acesso na tela de login para entrar na aplicação
Testes de sucesso	<ul style="list-style-type: none"> • Os dados do usuário são validados e criado um token para identificar a sessão ativa • O usuário será redirecionado para uma tela dentro do sistema
Testes de erro	<ul style="list-style-type: none"> • Exibir mensagem: Preencha todos campos obrigatórios para continuar.

	<ul style="list-style-type: none"> • Exibir mensagem: Os dados de acesso informado não estão válidos, confira e preencha novamente. • Exibir mensagem: O servidor não está respondendo como o esperado, tente novamente mais tarde
--	--

Fonte: Elaborado pelo autor, 2019

Quadro 4 – Cartão de história: Selecionar o método de estudo

Nº cartão	3
Título	Selecionar o método de estudo
História	O aluno logado seleciona qual método deseja estudar para abrir o modo apropriado
Testes de sucesso	<ul style="list-style-type: none"> • O aluno ao entrar no sistema precisa exibir na tela os métodos de estudos disponíveis
Testes de erro	<ul style="list-style-type: none"> • Exibir mensagem: Ocorreu um erro ao carregar os métodos de estudo

Fonte: Elaborado pelo autor, 2019

Quadro 5 – Cartão de história: Estudar com flashcards

Nº cartão	4
Título	Estudar com flashcards
História	O aluno autenticado precisa informar qual grupo de cartões tem interesse para estudar os flashcards correspondentes
Testes de sucesso	<ul style="list-style-type: none"> • O aluno entra no modo de flashcard para estudo, exibindo a definição e ao clicar, será exibido a palavra chave referente

Testes de erro	<ul style="list-style-type: none"> Exibir mensagem: Selecione um grupo de flashcard para estudar.
----------------	--

Fonte: Elaborado pelo autor, 2019

Quadro 6 – Cartão de história: Estudar com Perguntas e Respostas

Nº cartão	5
Título	Estudar com Perguntas e Respostas
História	O aluno autenticado precisa informar qual grupo de cartões tem interesse para estudar os grupo de perguntas e respostas
Testes de sucesso	<ul style="list-style-type: none"> O aluno entra no modo de Perguntas e Respostas para estudo, exibindo a pergunta e as alternativas para o aluno escolher a resposta mais apropriada
Testes de erro	<ul style="list-style-type: none"> Exibir mensagem: Selecione uma resposta

Fonte: Elaborado pelo autor, 2019

Quadro 7 – Cartão de história: Estudar com Verdadeiro ou Falso

Nº cartão	6
Título	Estudar com Verdadeiro ou Falso
História	O aluno autenticado precisa informar qual grupo de cartões tem interesse para estudar os grupo de verdadeiro ou falso
Testes de sucesso	<ul style="list-style-type: none"> O aluno entra no modo de Verdadeiro ou Falso para estudo, exibindo a pergunta e as alternativas para o aluno escolher a resposta mais apropriada

Testes de erro	<ul style="list-style-type: none"> Exibir mensagem: Selecione uma resposta
----------------	---

Fonte: Elaborado pelo autor, 2019

Quadro 8 – Cartão de história: Exibir Resultado

Nº cartão	7
Título	Exibir Resultado
História	Ao terminar de responder as questões o aluno irá ver os erros para saber como foi o seu desempenho
Testes de sucesso	<ul style="list-style-type: none"> No final do modo flashcards, verdadeiro ou falso e do perguntas e respostas precisa exibir uma tela de resultado mostrando o desempenho do aluno, exibindo quantos acertos e o número de erros ele obteve, junto com a porcentagem do seu desempenho Mostrar um número de estrelas correspondente ao seu desempenho. Quanto melhor desempenho, maior a quantidade de estrelas conquistadas.
Testes de erro	<ul style="list-style-type: none"> Exibir mensagem: Não foi possível carregar o resultado

Fonte: Elaborado pelo autor, 2019

Quadro 9 – Cartão de história: Exibir Ranking

Nº cartão	8
Título	Exibir Ranking
História	O aluno poderá saber como está seu desempenho em relação aos outros alunos por meio do ranking para motivar nos estudos

Testes de sucesso	<ul style="list-style-type: none">• O aluno escolhe a opção e o sistema exibe o ranking com a pontuação que os estudantes conseguiram obter ao responder as mesmas questões
Testes de erro	<ul style="list-style-type: none">• Exibir mensagem: Não existem registros para serem exibidos

Fonte: Elaborado pelo autor, 2019

APENDICE B – CONFIGURAÇÃO DO MANIFEST.XML

```

<manifest xmlns:android="http://schemas.android.com/apk/res/android"
  xmlns:tools="http://schemas.android.com/tools"
  package="br.com.studyplay.droid">

  <application
    android:allowBackup="true"
    android:icon="@mipmap/ic_launcher"
    android:label="StudyPlayDroid"
    android:roundIcon="@mipmap/ic_launcher_round"
    android:supportsRtl="true"
    android:theme="@style/AppTheme"
    tools:ignore="GoogleAppIndexingWarning" >

    <meta-data
      android:name="asset_statements"
      android:resource="@string/asset_statements" />

    <activity
      android:name="com.google.androidbrowserhelper.trusted.LauncherActivity">

      <meta-data
        android:name="android.support.customtabs.trusted.DEFAULT_URL"
        android:value="@string/config_default_url" />

      <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
      </intent-filter>

      <intent-filter>
        <action android:name="android.intent.action.VIEW"/>
        <category android:name="android.intent.category.DEFAULT" />
        <category android:name="android.intent.category.BROWSABLE"/>

        <data
          android:scheme="https"
          android:host="@string/config_host"/>
      </intent-filter>
    </activity>

  </application>
</manifest>

```