

**RAFAEL DA SILVA MOREIRA
RICARDO MEGALE BAUMGARTNER**

DESENVOLVIMENTO EM CLOUD COMPUTING

**UNIVERSIDADE DO VALE DO SAPUCAÍ
POUSO ALEGRE – MG
2013**

**RAFAEL DA SILVA MOREIRA
RICARDO MEGALE BAUMGARTNER**

DESENVOLVIMENTO EM CLOUD COMPUTING

Trabalho de conclusão de curso apresentado
ao Curso de Sistemas de Informação da
Universidade do Vale do Sapucaí para a
obtenção do título de Bacharel em Sistemas
de Informação sob a orientação do professor
Ednardo David Segura.

**UNIVERSIDADE DO VALE DO SAPUCAÍ
POUSO ALEGRE – MG
2013**

Dedicamos,

As nossas famílias, amigos e professores que foram muito importantes em nossa vida acadêmica, nos ajudando nessa conquista.

AGRADECIMENTOS

Ao Prof. Ednardo David Segura, por sua orientação competente e pelos estímulos constantes ao longo desta caminhada.

De Ricardo Megale Baumgartner:

Agradeço em primeiro lugar a Deus que me iluminou durante esta caminhada. Agradeço também de forma especial minha mãe, Sandra, pois foi quem sempre fez todo o possível para que eu chegasse aonde cheguei, minha irmã Luciana, pois com quem sei que posso contar em qualquer situação. Minha namorada Marina, que de muitas formas sempre me deu forças e coragem com apoio nos momentos de dificuldade. E meu companheiro Rafael que caminhou comigo nestes longos meses de trabalho.

De Rafael da Silva Moreira:

Agradeço em primeiro lugar a Deus que me iluminou e me deu forças e coragem durante esta caminhada. Aos meus pais (Nelson da Silva Moreira e Darcy Ribeiro de Lima Moreira), que sempre me apoiaram. Minha irmã (Juliana Lima Moreira) e minha namorada (Aline Raquel do Espírito Santo) que sempre estiveram do meu lado me apoiando e me dando forças para seguir essa caminhada. Também agradeço ao meu companheiro de trabalho (Ricardo Megale Baumgartener) e a todos os professores do curso, que foram tão importantes na minha vida acadêmica.

“O único lugar onde o sucesso vem antes do trabalho é no dicionário”.

(Albert Einstein)

BAUMGARTNER, Ricardo Megale; MOREIRA, Rafael da Silva. **Desenvolvimento em Cloud Computing**. 2013. Curso de Sistemas de Informação, Universidade do Vale do Sapucaí, Pouso Alegre, 2013.

RESUMO

A necessidade de acesso a recursos de forma rápida, com baixo custo fez surgir o cloud computing, um modelo computacional que permite a utilização de recursos apenas na quantidade e tempo que forem necessários. Pelo fato de esse modelo ser uma tendência para a área de tecnologia da informação, o objetivo desta pesquisa é mostrar os conceitos, as vantagens e desvantagens, os riscos e as possibilidades que ele oferece, documentando teorias e definições. A partir de toda a pesquisa realizada foi possível além da compreensão de seu funcionamento, também a criação de uma aplicação que utilizasse esses conceitos e teorias. Para que, além de documentadas, elas fossem demonstradas.

Palavras-chave: Modelos de serviço, Nuvem, Cloud, Cloud Computing, Computação em Nuvem, Modelos de Implantação, Modelos de Serviços, Nuvem Híbrida, Nuvem Pública, Nuvem Privada, Escalabilidade, Amazon Web Services, Windows Azure, Google AppFabric.

BAUMGARTNER, Ricardo Megale; MOREIRA, Rafael da Silva. **Desenvolvimento em Cloud Computing**. 2013. Curso de Sistemas de Informação, Universidade do Vale do Sapucaí, Pouso Alegre, 2013.

ABSTRACT

The requiring access to quickly resources, with low cost and without the need for physical equipment has raised the cloud computing, a computational model that allows the use of resources in the quantity and time needed. Because this model is an Information Technology tendency, the objective of this resource is show the concepts, the advantages and disadvantages, the risks and possibilities that it offers, documenting theories and definitions. From all the research done was possible further the understand its functioning, also creating an application that utilizes this concepts and theories. For besides documented they were demonstrated.

Palavras-chave: Modelos de serviço, Nuvem, Cloud, Cloud Computing, Computação em Nuvem, Modelos de Implantação, Modelos de Serviços, Nuvem Híbrida, Nuvem Pública, Nuvem Privada, Escalabilidade, Amazon Web Services, Windows Azure, Google AppFabric.

LISTA DE ABREVIATURAS E SIGLAS

ACL	<i>Access Control List.</i>
AWS	<i>Amazon Web Services.</i>
CSA	<i>Cloudy Security Alliance.</i>
CSS	<i>Cascade Style Sheet.</i>
DNS	<i>Domain Name System.</i>
DOM	<i>Document Object Model.</i>
EBS	<i>Elastic Block Store.</i>
EC2	<i>Elastic Compute Cloud.</i>
EMR	<i>Elastic Map Reduce.</i>
FAI	Faculdade de Administração e Informática.
GAE	<i>Google AppEngine.</i>
GLP	<i>General Public License.</i>
HTML	<i>Hypertext Markup Language.</i>
HTTP	<i>Hypertext Transfer Protocol.</i>
IAAS	<i>Infrastructure as a Service.</i>
IAM	<i>Identity and Access Management.</i>
ID	Identidade.
IDE	<i>Integrated Developement Enviroment.</i>
IFE	Instituto Federal de Educação, Ciência e Tecnologia.
IIS	<i>Internet Information Services.</i>
IP	<i>Internet Protocol.</i>
MIT	<i>Massachusetts Institute of Tecnology.</i>
NTFS	New Technology File System.
PDO	<i>PHP Data Objects.</i>
PHP	<i>Hypertext Preprocessor.</i>
PAAS	<i>Platform as a Service.</i>
RAM	<i>Random Access Memory.</i>
RDS	<i>Relational Database Service.</i>
S3	<i>Simple Storage Service.</i>
SAAS	<i>Software as a Service.</i>

SDB	<i>Simple Database.</i>
SDK	<i>Software Development Kit.</i>
SES	<i>Simple Email Service.</i>
SLA	<i>Service Level Agreement.</i>
SMTP	<i>Simple Mail Transfer Protocol.</i>
SNS	<i>Simple Notification Service.</i>
SOAP	<i>Simple Object Access Protocol.</i>
SQS	<i>Simple Queue Service.</i>
SVN	<i>Subversion.</i>
UNIFEI	Universidade Federal de Itajubá.
UNIVÁS	Universidade do Vale do Sapucaí.
VPC	<i>Virtual Private Cloud.</i>
W3C	<i>World Wide Web Consortium.</i>
XML	<i>Extensible Markup Language.</i>

LISTA DE FIGURAS

Figura 1 – Código exemplo de uma estrutura HTML	19
Figura 2 – Sintaxe CSS.....	22
Figura 3 – Exemplo de PHP inserido no HTML	23
Figura 4 – Plataforma como serviço.....	30
Figura 5 – Modelos de serviços da computação em nuvem	31
Figura 6 – Modelos de serviços.....	31
Figura 7 – Modelos de implantação	33
Figura 8 – Principais componentes e serviços do Windows Azure.....	39
Figura 9 – Funcionamento das roles	40
Figura 10 – Funcionamento dos <i>blobs</i>	41
Figura 11 – Funcionamento do <i>tables</i>	42
Figura 12 – Funcionamento das <i>queues</i>	42
Figura 13 – Visão geral do <i>Azure Virtual Network</i>	44
Figura 14 – Roteamento do <i>Traffic Manager</i>	45
Figura 15 – Principais componentes.....	46
Figura 16 – Estudo de caso com CloudFront	48
Figura 17 – Arquitetura de um sistema no GAE.	55
Figura 18 – Ciclo de vida do programa de conscientização	58
Figura 19 – Funcionamento do servidor.....	60
Figura 20 – Aplicação implantada no Elastic Beanstalk	63
Figura 21 – Arquitetura Elastic Beanstalk.....	64
Figura 22 – Instância criada no RDS.....	64
Figura 23– Configuração da IDE para acesso ao RDS.....	65
Figura 24– Efetuando copia de um <i>snapshot</i> para outra região.	66
Figura 25– Aplicação desenvolvida.	67
Figura 26– Arquitetura de usuário utilizando os recursos AWS	68
Figura 27 – Código PHP pra criação do <i>bucket</i> no S3	69
Figura 28– <i>Buckets</i> de usuários no S3.....	69
Figura 29 – Código PHP para criação de álbuns e arquivos.	70
Figura 30 – Álbuns e fotos criados pela aplicação	70
Figura 31 – Código PHP para exclusão de álbuns e arquivos.	71
Figura 32 – Painel de configuração do Elastic Beanstalk	72
Figura 33 – Instância criada no EC2.	73
Figura 34 – Painel de estatística da instância	74
Figura 35– Estatísticas de uso da aplicação com uma instância.....	74
Figura 36 – Configurações de escalabilidade.	75
Figura 37 – Estatísticas de uso da aplicação com uma instância.....	75
Figura 38 – Criação da segunda instância.	76
Figura 39 – Estatísticas de uso da aplicação com duas instâncias.....	76
Figura 40 – Segunda instância sendo terminada.	76

SUMÁRIO

INTRODUÇÃO.....	13
2. QUADRO TEÓRICO.....	17
2.1 CLOUD COMPUTING.....	17
2.2 SEGURANÇA DA INFORMAÇÃO	18
2.3 TECNOLOGIAS USADAS NA APLICAÇÃO DA PESQUISA	18
2.3.1 HYPERTEXT MARKUP LANGUAGE (HTML)	19
2.3.2 JAVASCRIPT	20
2.3.3 JQUERY	21
2.3.4 CASCADING STYLE SHEET (CSS)	22
2.3.5 PHP	23
3 QUADRO METODOLÓGICO.....	25
3.1 TIPOS DE PESQUISA.....	25
3.2 CONTEXTOS DA PESQUISA	26
3.3 INSTRUMENTOS	26
3.4 PROCEDIMENTOS.....	27
3.4.1 PLANEJAMENTO.....	27
3.5 CARACTERÍSTICAS DO CLOUD COMPUTING	27
3.6 MODELOS DE SERVIÇOS	29
3.7 MODELOS DE IMPLANTAÇÃO	32
3.8 SEGURANÇA.....	33
3.8.1 CONTROLES DE SEGURANÇA.....	34
3.8.2 PADRÕES DE SEGURANÇA DA FEDERAÇÃO	35
3.8.3 SLA.....	36
3.8.4 FATORES DE SEGURANÇA NA MIGRAÇÃO PARA NUVEM.....	36
3.8.5 OUTRAS QUESTÕES DE SEGURANÇA.....	37
3.9 PLATAFORMAS DE DESENVOLVIMENTO.....	38
3.9.1 WINDOWS AZURE	38
3.9.2 AWS AMAZON.....	45
3.9.3 GOOGLE APP ENGINE	55
3.10 ESTUDOS DE CASO	57
3.10.1 GRUPO ING	57
3.10.2 GOL LINHAS AÉREAS.....	59
3.10.3 PEIXE URBANO	61

3.10.4 FOURSQUARE	61
3.11 DESENVOLVIMENTO DO SOFTWARE	62
3.12 IMPLANTAÇÃO	63
3.13 MIGRAÇÃO ENTRE REGIÕES.....	65
3.14 RESULTADOS	67
4 DISCUSSÃO DOS RESULTADOS	73
5 CONSIDERAÇÕES FINAIS	77
REFERÊNCIAS	78

INTRODUÇÃO

Todas as áreas de conhecimento sofrem alterações ao longo do tempo, sejam pela mudança do modo em que as pessoas interagem com o mundo ou pela invenção de uma nova tecnologia, seja pela descoberta de novos recursos ou pela escassez deles. Com a área da tecnologia não é diferente e uma dessas mudanças, que está transformando o modo como consumimos a tecnologia, já começou (TAURION, 2010).

O uso de computadores principalmente com conexão à internet se tornou muito presente na vida das pessoas. Com a rápida evolução das tecnologias de comunicação, é possível encontrar conexão à internet com altas velocidades e preços acessíveis, criando um cenário favorável para a utilização da computação em nuvem.

Cloud computing, que na sua tradução para o português significa “Computação em Nuvem”, é descrita como o uso de recursos computacionais (*hardware e software*) que são oferecidos como um serviço através de uma rede, geralmente a internet (GUIMARÃES, 2012).

A grande característica que distingue o cloud computing do modelo convencional de entrega de recursos de computação (processamento, espaço em disco e etc.) é a escalabilidade deles. De acordo com a necessidade em que forem requisitados esses recursos serão entregues, podendo ser em grande quantidade, assim como em um momento de ociosidade poderão ser entregues de forma reduzida.

Esses recursos são disponibilizados através de *data centers*¹ onde as aplicações são implementadas e usam os recursos que são disponibilizados para o processamento, armazenamento e controle da aplicação. Com isso o usuário necessita apenas de um simples *hardware*² com uma boa conexão com a internet para acessar as aplicações a qualquer hora e lugar (TAURION, 2010).

Atualmente as empresas que disponibilizam esse serviço com melhor qualidade e maior diversificação são a Google com o Google Apps, Microsoft com o Azure e a Amazon com o AWS. Com destaque para a última que, além de oferecer suporte para as principais linguagens de programação, também disponibiliza seus serviços de forma independente uns dos outros.

¹ *Data Centers* são centrais de armazenamento e processamento de dados onde vários servidores são alocados.

² *Hardware* é a parte física de um computador formada pelos componentes eletrônicos.

Esse novo modelo pode reduzir custos, pois sistemas de gerenciamento mais robustos não mais necessitarão de grandes recursos físicos como servidores potentes e uma rede bem estruturada para serem implantados. O processamento dos dados será feito na nuvem pelos servidores e entregue ao computador que requisitou através do navegador de internet (TAURION, 2009).

O fato de ser possível acessar as aplicações de qualquer local pode facilitar a disposição de uma empresa espalhada geograficamente permitindo, por exemplo, a integração sem a necessidade de uma grande infraestrutura e gastos com manutenção, diminuindo os custos e riscos que podem ocorrer quando se tem uma estrutura de comunicação complexa.

Como o acesso a essas aplicações não requer compatibilidade com o sistema operacional instalado, é possível utilizar a aplicação tanto no Windows como no Linux ou IOS não sendo necessária a troca de equipamentos ou de sistemas operacionais por incompatibilidade.

A computação em nuvem é confundida às vezes com a computação em grade (*Grid Computing*), pois as duas tecnologias visam flexibilidade através de recursos ociosos de vários computadores. A grande diferença é a alocação de recursos que, na computação em nuvem, são alocados conforme a demanda e na computação em grade é feita uma distribuição por igual desses recursos (TAURION, 2009).

A segurança dos dados sempre foi uma preocupação global na área tecnológica. O risco existe em computadores pessoais, *tablets*, celulares e em grandes empresas. Todos estão vulneráveis e isso não é diferente quando se utiliza o cloud computing. Em um documento feito pela CSA (Cloud Security Alliance) foram listadas e descritas grandes ameaças à segurança da computação em nuvem na atualidade.

Dentre elas estão questões importantes, como violação e perda dos dados, maior facilidade de interceptação de informações por conta de trafegarem na internet e a perda de credenciais de acesso, que poderiam prejudicar de forma irreversível uma organização.

Essa é uma questão primordial e delicada para a computação em nuvem, pois irá influenciar no rumo desse modelo. A escolha da utilização do cloud computing deve levar em conta o grau de necessidade de uso do sistema. Quando é preciso ter certeza da localização dos dados e disponibilidade total da aplicação, sem depender da conexão com a internet ou ter níveis maiores de segurança deve-se considerar uma nuvem privada (TAURION, 2009).

Esse novo modelo já despertou o interesse de muitas pessoas, como Edin (2011) que elaborou um trabalho em que comenta as vantagens e desvantagens da computação em nuvem

explicando a arquitetura, os modelos e duas das principais plataformas de desenvolvimento em cloud computing. Também realizou um comparativo entre a Amazon AWS e o Microsoft Azure. Nesse comparativo ele constata que o Azure por ter uma plataforma onde fornece integradas todas as ferramentas necessárias para o desenvolvimento, torna mais fácil o gerenciamento das instâncias e aplicações implantadas no servidor, mas suporta uma quantidade menor de linguagens de programação. Também viu que a Amazon fornece a infraestrutura primária, com acesso sob demanda às instâncias de máquinas virtuais personalizáveis e tem suporte a um número maior de linguagens para programação.

Hsu (2009) apresentou um estudo de caso com os diversos conceitos em relação ao cloud computing, debatendo sobre a implantação desse modelo na empresa Pathwork que necessitava suprir sua necessidade de processamento. Ele ressalta os benefícios obtidos nessa implantação como economia e maior capacidade computacional sem precisar investir em infraestrutura.

Melo (2009) também mostrou os conceitos que definem o que é a computação em nuvem, destacando suas vantagens, desvantagens e os principais serviços da nuvem. Também citou sobre a segurança, confiabilidade e as empresas que oferecem esse serviço.

Muitos já documentaram e estudaram esse novo modelo computacional e pelo fato de a tecnologia evoluir de um modo muito rápido, pensou-se ser válido estudar novamente esses conceitos para criar um estudo mais recente. Pesquisas anteriores não demonstraram esse modelo em funcionamento, por isso esta pesquisa tem o objetivo geral de estudar os principais conceitos do cloud computing e demonstrar esse modelo em uso. Para o alcance dele foram seguidos alguns passos:

- Mostrar as vantagens e desvantagens desse modelo computacional em relação ao modelo tradicional;
- Estudar os modelos de implantação e de serviços;
- Analisar casos de uso para mostrar algumas possibilidades da computação em nuvem;
- Analisar os aspectos de segurança dos dados nesse modelo;
- Desenvolver uma aplicação para demonstrar o uso desse modelo computacional.

O estudo para o conhecimento dessa plataforma justifica-se pela redução de recursos materiais e financeiros necessários para empresas de qualquer segmento, pois permitirá a

contratação de sistemas mais robustos utilizando menos recursos computacionais, barateando várias tecnologias, deixando-as mais acessíveis para a população.

Esse modelo computacional permite ainda que projetos carentes, de inclusão social, escolas ou órgãos públicos, possam ter acesso a várias tecnologias com um gasto muito menor, possibilitando que eles tenham melhor qualidade, beneficiando a sociedade. Com a redução da infraestrutura de *hardware* necessária para esse modelo, a quantidade de lixo tecnológico gerado será drasticamente reduzida, sendo um aliado para um desenvolvimento e uma sociedade sustentável.

Com a documentação dos principais conceitos e da utilização das principais ferramentas do cloud computing como o uso dos recursos sob demanda, utilização de banco de dados, criação de instâncias, redes virtuais dentre outras, espera-se que este projeto possa ser utilizado como auxílio em estudos ou pesquisas sobre o assunto abordado.

O trabalho se divide em quatro capítulos. Sendo o primeiro capítulo uma introdução sobre o tema que será abordado mostrando conceitos básicos e o objetivo dessa pesquisa. O segundo capítulo apresenta as teorias e descrições de ferramentas utilizadas para a demonstração da pesquisa. O terceiro capítulo demonstra o tipo e o contexto da pesquisa, os instrumentos para produção de dados e os procedimentos. No quarto e quinto capítulo encerrou-se com a discussão de resultados que traz, como conteúdo, o que foi alcançado com os estudos por trás desta pesquisa. Por fim são apresentadas as referências utilizadas para a formação deste documento.

2. QUADRO TEÓRICO

Neste tópico serão abordadas as principais ferramentas, conceitos, características, modelos de implantação, modelos de serviços, questões de segurança e estudos de caso, assim como as linguagens PHP, JavaScript, HTML e CSS suportadas pelos servidores para o desenvolvimento em nuvem e foram utilizadas na aplicação deste projeto.

2.1 Cloud Computing

Quando o termo computação em nuvem apareceu pela primeira vez, descobriu-se uma forma de computação rápida. Uma analogia muito utilizada foi a da eletricidade que é gerada por vários fornecedores, entregue sob demanda e paga pela quantidade usada. Esse foi o conceito que deu origem às nuvens públicas que fornecem recursos de computação na forma comumente chamada *infrastructure-as-a-service* (IaaS). As nuvens públicas não atendem a todos os clientes principalmente em questões de segurança, por esse motivo surgiram nuvens privadas e híbridas (HADHAT, 2012).

O preço das máquinas virtuais é muito baixo, por isso usuários e desenvolvedores podem, por meio de pagamento eletrônico, ter acesso e alocar esses recursos instantaneamente não sendo necessária a espera por meses para um novo servidor ser montado e estar disponível (HADHAT, 2012).

Muitas organizações ainda não estão prontas pra mover todas suas aplicações para provedores de nuvens públicas. Em geral, isso ocorre por causa de conceitos em torno de conformidade e governança, por isso existem vários aspectos de segurança e modelos de serviços que devem ser analisados para a escolha da utilização ou não dessa tecnologia (HADHAT, 2012).

2.2 Segurança da informação

Quando as informações eram armazenadas apenas em papel, a segurança era mais simples, pois trancar os documentos e restringir o acesso físico àquele local era relativamente simples. Com o avanço tecnológico e o uso de computadores, a segurança ficou um pouco mais complicada. Com isso a segurança atingiu uma complexidade muito grande e a necessidade de desenvolvimento de equipes e de métodos de segurança cada vez mais sofisticados se tornou essencial. Junto a isso, os sistemas de informação também adquiriram uma grande importância para a maioria das organizações, já que, sem computadores e redes de comunicação, a prestação de serviços de informação torna-se praticamente impossível (ARTHUR, 2009).

Por isso o uso de alguns métodos para garantir a segurança dos dados trafegados nessas redes se tornou indispensável. Existem alguns aspectos a serem analisados como:

- Análise de riscos;
- Política de segurança;
- Controle de acesso físico e lógico;
- Treinamento e conscientização dos funcionários;
- Plano de contingência;

A informação é o patrimônio mais valioso de uma empresa e levar a sério a segurança dessas informações é fundamental. Se todos os envolvidos no processo de manipulação desse patrimônio seguirem as regras, os riscos diminuem consideravelmente (ARTHUR, 2009).

2.3 Tecnologias usadas na aplicação da pesquisa

Aqui serão abordadas as linguagens de desenvolvimento usadas para o desenvolvimento dessa pesquisa explicando-as e mostrando suas principais ferramentas.

2.3.1 Hypertext Markup Language (HTML)

HTML (*Hypertext Markup Language*) que, traduzindo, significa linguagem para marcação de hipertexto, foi criada na década de noventa por Tim Berners-Lee. As especificações da linguagem são controladas pela W3C³ a fim de padronizar o uso dela (SILVA, 2011).

Essa linguagem foi criada à princípio, com objetivo de divulgação, porém não se imaginava que futuramente a utilização cresceria de maneira exorbitante. Essa evolução do HTML fez com que surgisse uma serie de *bugs* que necessitaram de correções. Com isso foram incorporadas novas tecnologias como o CSS e JavaScript, que serão abordadas adiante (ALVAREZ, 2004)

O HTML é uma linguagem usada para a estruturação de uma página *web* e limita-se a criar os rótulos e campos de um formulário para serem preenchidos pelos usuários e nada mais, não sendo usada para empregar estilos, funcionalidades ou efeitos visuais. A Figura 1 mostra um exemplo de uma estrutura HTML (SILVA, 2011).

```
<!DOCTYPE html>
<html>
<body>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=ISO-88541">
<title>Estrutura HTML</title>

</head>

<h1>Titulo</h1>
<h2>Subtitulo</h2>
<p>Parágrafo.</p>

</body>
</html>
```

Fig. 1 – Código exemplo de uma estrutura HTML **Fonte:** Elaborado pelo autor.

Nesse exemplo podemos identificar algumas das principais *tags* para documentos HTML.

³ W3C – Abreviação para *World Wide Web Consortium*.

- <html> define o início do documento. Indica ao navegador que o conteúdo entre essas *tags* devem ser tratadas como código;
- <head> são etiquetas e conteúdo do cabeçalho;
- <body> são etiquetas e conteúdo do corpo, contendo parte do documento que será mostrado pelo navegador, como textos e imagens;

Uma página HTML é um arquivo que contém escrito todo o código em modo texto. Esse arquivo possui a extensão de .html ou .htm, não existindo diferença alguma entre qual delas utilizar, porém é aconselhado utilizar sempre a mesma extensão em todos os arquivos para que não haja confusão (ALVAREZ, 2004).

2.3.2 JavaScript

O JavaScript é uma linguagem de programação. Seu núcleo assemelha-se ao C, C++ e Java que contém em seu núcleo as construções de programação como a instrução *if*, o laço *while* e o operador **&&**. Essa linguagem é interpretada com recursos de orientação a objetos e o núcleo de utilização geral da linguagem foi incorporado ao Netscape, Internet Explorer e outros navegadores para programação *web* (GOODMAN, 2002).

Essa linguagem foi criada para dar interatividade a uma página *web*, fornecendo funcionalidades aos botões e formulários. Inicialmente foi desenvolvida para rodar no lado cliente, mas atualmente existem interpretadores que possibilitam seu uso no lado servidor também (SILVA, 2010).

2.3.3 jQuery

O jQuery é uma biblioteca JavaScript de *software* livre e aberto, desenvolvida por John Reisig. Essa biblioteca tem seu uso regido conforme as regras estabelecidas pelo MIT⁴ e pela GPL⁵. Ela pode ser utilizada para projetos pessoais e comerciais (SILVA, 2012).

Essa biblioteca simplifica e muito o uso do JavaScript. Com ela, efeitos e funcionalidades que gerariam confusas e numerosas linhas de códigos, são possíveis em poucas linhas. Essa biblioteca é bem descrita por Silva (2010, p. 25).

JQuery destina-se a adicionar interatividade e dinamismo às páginas *web*, proporcionando ao desenvolvedor funcionalidades necessárias à criação de *scripts* que visem a incrementar, de forma progressiva e não obstrutiva, a usabilidade, a acessibilidade e o *design*, enriquecendo a experiência do usuário.

O jQuery pode ser utilizado em um *site* para as seguintes funcionalidades:

- Adicionar efeitos visuais e animações;
- Acessar e manipular o DOM⁶;
- Buscar informações no servidor sem necessidade de recarregar a página;
- Prover interatividade;
- Alterar conteúdos;
- Modificar apresentação e estilização;
- Simplificar tarefas específicas de JavaScript;

Com isso entende-se que o jQuery é uma biblioteca de extrema importância, pois traz recursos que minimizam as dificuldades em programação e geram grandes benefícios em um site como adicionar interatividade e dinamismo as páginas *web*.

⁴ MIT – Abreviação de Massachusetts Institute of Technology.

⁵ GPL – Abreviação de *General Public License*.

⁶ DOM – Abreviação para *Document Object Model*

2.3.4 Cascading Style Sheet (CSS)

O CSS é a sigla para Cascading Style Sheet que, traduzido, significa efeitos de estilo em cascata. É definido como um modo simples e organizado de adicionar estilos, fontes e posição para as marcações, deixando o HTML com sua função original de exclusivamente fazer a marcação e estruturação dos conteúdos. Cabe ao CSS todas as funções de apresentação da página *web* (SILVA, 2012). Segundo Miyagusku (2007, p. 8);

A função principal das CSS é, justamente, extrair a formatação de uma página do código HTML, separando-a do conteúdo propriamente dito (informações). Além de aumentar o nível de organização, isso indica que elas podem definir, de antemão, a formatação de todos os elementos de uma ou várias páginas.

O CSS foi criado proveniente das deficiências e limitações que o HTML passou a apresentar. Com a necessidade de *sites* mais complexos, repletos de recursos e informações, a questão da formatação e manutenção de arquivos HTML passou a ficar complicada para os programadores. Para resolver esse problema surgiu o CSS que separa as configurações de formatações do conteúdo do arquivo HTML. A configuração de formatação fica em um arquivo à parte. Abaixo a Figura 2 apresenta a sintaxe do CSS (MIYASGUSKU, 2012).

```
01. p {  
02.   font-size: 12px; /* ponto-e-vírgula é facultativo */  
03. }  
04.  
05. body {  
06.   color: #000;  
07.   background: #fff;  
08.   font-weight: bold; /*ponto-e-vírgula é facultativo  
09. */  
10. }
```

Fig. 2 – Sintaxe CSS **Fonte:** <http://www.maujor.com/tutorial/sintaxetut.php>

A sintaxe de uma regra CSS está descrita no seletor apresentado acima. O elemento antes das chaves “{ }” (parágrafos, cabeçalhos, componentes de lista de tabelas entre outros)

indica o elemento da página HTML. Os atributos que estão dentro das chaves e seus respectivos valores definem a formatação do elemento.

A utilização do CSS fornece a certeza de que todo o *site* estará padronizado e formatado, proporcionando uma enorme economia de trabalho, pois dispensa a repetição de código em cada arquivo HTML. Também o mesmo código CSS pode ser reaproveitado para definir o layout em outros arquivos HTML.

2.3.5 PHP

O PHP (Hypertext Preprocessor) é uma linguagem de script de uso geral utilizada especialmente para o desenvolvimento de aplicações *web* dentro do HTML. A Figura 3 ilustra um exemplo do PHP inserido no HTML.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN"
 "http://www.w3.org/TR/html4/loose.dtd">
<html>
  <head>
    <title>Exemplo</title>
  </head>
  <body>

    <?php
      echo "Olá, Eu sou um script PHP!";
    ?>

  </body>
</html>
```

Fig. 3 – Exemplo de PHP inserido no HTML
Fonte: http://www.php.net/manual/pt_BR/intro-whatcando.php

Ao invés de muitos comandos para mostrar HTML, as páginas PHP contém o HTML juntamente com seus códigos, neste caso, mostra na tela a frase "Olá, Eu sou um script PHP!". O código é delimitado por *tags* iniciais e finais (`<?php` e `?>`) que permitem mudar do HTML para a linguagem PHP.

O que distingui o PHP de algo como JavaScript no lado do cliente é que o código é executado no servidor, gerando HTML que é então enviado para o cliente. O cliente recebe os resultados da execução desse *script*, mas não sabe como é o código fonte.

O PHP é uma linguagem de *script* do lado do servidor, portanto, é possível coletar dados de formulários, gerar páginas com conteúdo dinâmico ou enviar e receber *cookies*⁷. Existem alguns campos onde os *scripts* podem ser utilizados:

- *Script* no lado do servidor (*server-side*);
- *Script* de linha de comando;
- Utilizado para aplicações *desktop*;

Ele pode ser utilizado na maioria dos sistemas operacionais, incluindo Linux, variantes Unix (incluindo HP-UX, Solaris e OpenBSD), Microsoft Windows, Mac OS X, RISC OS, e outros. Também é suportado pela maioria dos servidores *web* atuais, incluindo Apache, Microsoft Internet Information Server, Personal Web Server, Netscape and iPlanet Servers, Oreilly Website Pro Server, Caudium, Xitami, OmniHTTPd, e outros.

⁷ Cookies são grupos de dados trocados entre o navegador e o servidor de páginas, colocado num arquivo de texto criado no computador.

3 QUADRO METODOLÓGICO

Neste capítulo são abordados os métodos adotados para conduzir a pesquisa. Nele são definidos o tipo de pesquisa, o contexto da pesquisa, participantes e procedimentos para desenvolvimento do projeto.

3.1 Tipos de pesquisa

Para atingir os reais objetivos desta pesquisa, foi utilizado um design que a caracteriza como exploratória. Esse tipo de pesquisa permitiu um estudo para obter os conhecimentos necessários sobre o tema proposto.

Para Andrade (2007, p.111), “Pesquisa é o conjunto de procedimentos sistemáticos, baseado no raciocínio lógico, que tem por objetivo encontrar soluções para problemas propostos, mediante a utilização de métodos científicos”.

A pesquisa exploratória pode ser entendida como um estudo que tem como principal objetivo descobrir ideias, percepções, gerar hipóteses precisas sobre um determinado assunto. Caracteriza-se por sua flexibilidade e versatilidade, pois não são empregados procedimentos formais de pesquisa (HONORATO, 2004).

Gil (2007, p.43) afirma que:

As pesquisas exploratórias têm como principal finalidade desenvolver, esclarecer, modificar conceitos e ideias tendo em vista, a formulação de problemas mais precisos ou hipóteses pesquisáveis para estudos posteriores. De todos os tipos de pesquisa, estas são as que apresentam menor rigidez no planejamento. Habitualmente envolve levantamento bibliográfico e documental, entrevistas não padronizadas e estudos de caso.

Ainda Gil (2007) diz que o objetivo de pesquisa exploratória é fornecer a visão geral de modo aproximativo, sobre determinado assunto. Esse tipo de pesquisa é bastante utilizado quando o tema proposto é pouco conhecido e explorado.

Com base nessas informações o tipo de pesquisa que melhor se define nesse trabalho é a pesquisa exploratória, com objetivo de explorar as documentações bibliográficas existentes

em *sites* e livros, pesquisar as teorias já definidas, explorar os recursos da nuvem escolhida e abordar de maneira clara tudo que foi pesquisado.

3.2 Contextos da pesquisa

O objetivo desse projeto é definir e comparar as principais plataformas de desenvolvimento para o cloud computing. Para isso realizou-se uma breve definição de algumas empresas que provém esse serviço. O provedor a ser utilizado para demonstração prática dos recursos será o AWS Amazon, que disponibiliza a maior variedade de serviços na nuvem.

Esta pesquisa, por ter informações comparativas entre os servidores mais conhecidos em cloud computing, servirá de base de conhecimento para estudantes, profissionais da área de tecnologia da informação e empresas que pretendem contratar e utilizar esse serviço.

3.3 Instrumentos

Para a obtenção e organização das informações, divisão das tarefas e desenvolvimento desta pesquisa foram feitas reuniões entre os participantes. Os dados foram conseguidos através de pesquisas bibliográficas, conhecimento dos provedores de cloud computing existentes e compreensão dos serviços oferecidos. Após isso foi possível analisar e organizar as informações.

Os encontros dos membros da equipe tiveram o propósito de discutir a linha a ser seguida, pois o tema escolhido é muito abrangente, em busca de definir o que seria necessário ser pesquisado e compreendido. Essas reuniões permitiram que fossem propostas situações e tomadas decisões quanto ao desenvolvimento do projeto, como as tecnologias abordadas e também o modo como seriam demonstradas.

3.4 Procedimentos

Durante a fase de desenvolvimento foi necessário adquirir conhecimentos das tecnologias utilizadas na pesquisa para a utilização no projeto proposto.

O primeiro passo foi o estudo da Amazon Web Services, em que será implantado o *software* desenvolvido para demonstração do modelo cloud computing. Esse estudo aconteceu por meio de tutoriais e da documentação oficial oferecida pela empresa, além de vídeos de palestras e *blogs*.

A seguir foram explorados alguns recursos como a criação de uma máquina virtual, configuração de um banco de dados e o armazenamento de arquivos.

Após o conhecimento do AWS foi pesquisado outros dois importantes servidores de cloud computing, o Windows Azure e Google App Engine. Os estudos foram feitos em cima de vídeos, tutoriais e da documentação oferecida pelas empresas em seus respectivos *sites*.

3.4.1 Planejamento

Foram escolhidas as tecnologias para uso no desenvolvimento da aplicação. O banco de dados usado foi o MySQL, o Xampp foi utilizado para simular um servidor local e permitir que a aplicação pudesse ser desenvolvida localmente, evitando gastos com o provedor de serviços.

A IDE⁸ escolhida foi o NetBeans e junto dele foi usado um controlador de versão para que os desenvolvedores trabalhassem juntos facilitando a criação e mescla dos códigos, assim como a realização de *backups*.

3.5 Características do Cloud Computing

O modelo de computação em nuvem é formado por três modelos de serviços, quatro modelos de implantação e cinco características que, juntas, são julgados essenciais para que uma aplicação se caracterize no modelo de cloud computing (BADGER, 2011). Essas

⁸ IDE—É um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar este processo.

características são as principais diferenças entre computação em nuvem e a TI⁹ tradicional (SNOWMAN, 2010). Essas características são citas a seguir.

- *On-demand self-service* (Autosserviço sob demanda) é onde o usuário pode escolher os recursos computacionais conforme a necessidade, sem necessitar de interação humana com o provedor de serviço.
- *Broad network access* (Acesso por banda larga) é quando recursos estão disponíveis através da internet e podem ser acessados por mecanismos padronizados independente da plataforma, possibilitando o acesso através de celulares, *notebooks*, *desktops*, etc.
- *Resource pooling* (Agrupamento de recursos) é quando o provedor de serviços agrupa os recursos computacionais para atender vários clientes (armazenamento, processamento, memória, banda larga e máquinas virtuais¹⁰) utilizando um modelo *multi-tenant*¹¹ em que os recursos físicos e virtuais são atribuídos e realocados dinamicamente de acordo com a demanda do cliente. Geralmente o cliente não tem controle ou conhecimento sobre a localização exata de onde os arquivos estão armazenados, mas pode especificar o local como o estado, país ou *data center* de sua preferência.
- *Rapid elasticity* (Elasticidade rápida) é quando recursos podem ser disponibilizados e configurados rapidamente ou, em alguns casos, configurados de forma automática, aumentando ou diminuindo a quantidade de recursos de acordo com a necessidade do usuário.
- *Measured service* (serviço mensurado) a nuvem controla e otimiza automaticamente a utilização dos recursos de acordo com o serviço fornecido. O uso dos recursos pode ser controlado e monitorado tanto pelo usuário quanto pelo provedor de forma transparente.

⁹ TI – Abreviação de Tecnologia da Informação.

¹⁰ Máquina Virtual é um *software* de ambiente computacional em que um sistema operacional ou programa pode ser instalado e executado.

¹¹ *Multi-Tenant* é uma arquitetura que permite que múltiplos usuários compartilhem os mesmos recursos físicos do servidor, mas permaneçam logicamente isolados.

3.6 Modelos de Serviços

O conceito de computação em nuvem é baseado na entrega dos serviços através da internet, serviços esses que são cobrados apenas pelo seu uso (*pay-per-use*), dando ao usuário a possibilidade de aumentar ou diminuir as capacidades de armazenamento e processamento conforme o necessário (TAURION, 2009).

A grande diversificação dos serviços pode ser um ponto positivo, o qual permite que o cliente adquira os serviços mais adequados às suas necessidades, porém pode também ser negativo, pois a maior parte dos serviços podem não ser compatíveis entre si (TAURION, 2009). Os modelos de serviços são divididos em três grupos (SNOWMAN, 2010).

- SaaS (*Software como um Serviço*): É uma aplicação completa sendo executada na nuvem e disponível para os clientes sem a necessidade de configurações adicionais. Os usuários deste serviço não têm contato com os serviços técnicos que ficam a cargo da empresa prestadora de serviço, os custos são reduzidos, pois não necessita de aquisição de licenças de *software*¹². Os serviços podem ser acessados pelo usuário de qualquer dispositivo com acesso a internet. Muitos destes *softwares* já são muito utilizados diariamente sem que o usuário saiba que está utilizando um serviço da nuvem. Alguns exemplos de SaaS são o Google Apps, Salesforce, Facebook, Linkedin, Sky Drive, dentre outros.
- PaaS (Plataforma como um Serviço): Permite que o usuário possa desenvolver suas aplicações e disponibilizá-las na nuvem. Esse tipo de plataforma oferece suporte para implementar e testar aplicações nas nuvens. O usuário não gerencia ou controla a camada, mas controla a aplicação desenvolvida na nuvem. É o provedor da PaaS que fornece o sistema operacional, a linguagem de programação e o ambiente de desenvolvimento para o usuário desenvolver suas aplicações. São exemplos de PaaS o Google AppEngine, Windows Azure, Aneka Software, Bungee Connect, Heroku, Force.com, Amazon S3, dentre outros. A plataforma como um serviço fica bem ilustrada na Figura 4.

¹² *Software* – conjunto de instruções a serem seguidas por uma máquina.

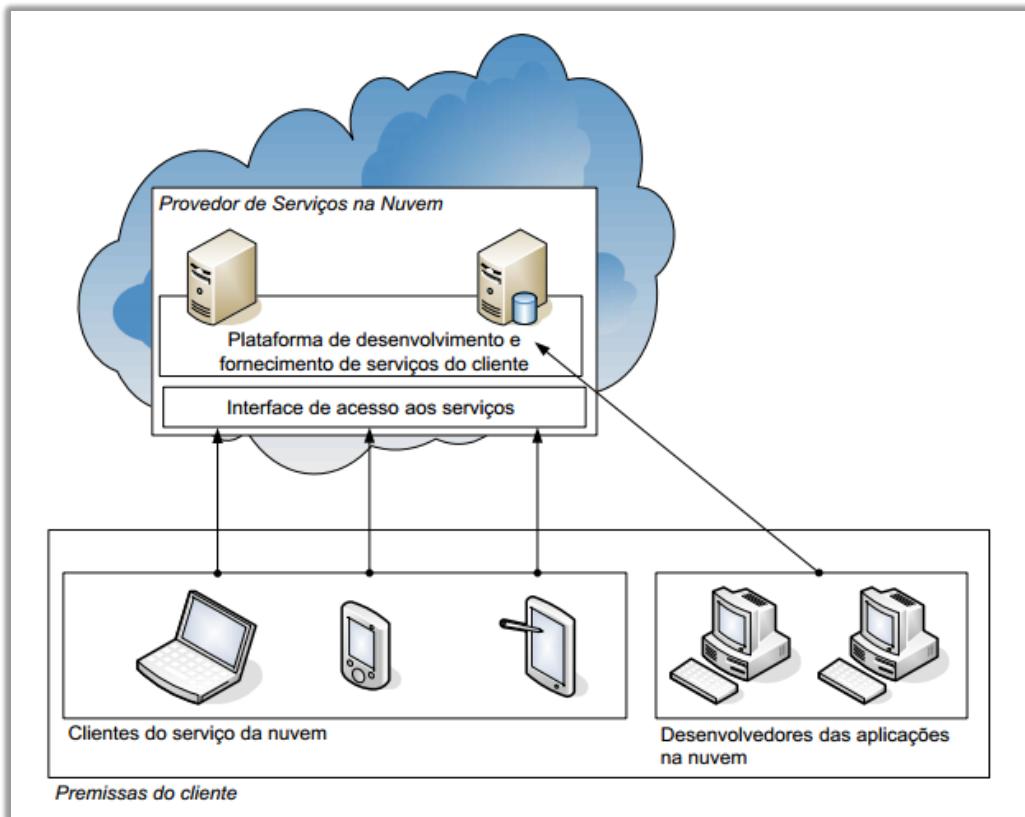


Fig. 4 – Plataforma como serviço

Fonte: <https://chapters.cloudsecurityalliance.org/brazil/files/2012/04/cap17.pdf>

- IaaS (Infra-Estrutura como um Serviço): O provedor oferece a infraestrutura de armazenamento e processamento para facilitar a entrega dos recursos de computação necessários para criar um ambiente de aplicação. O usuário não administra ou controla a infraestrutura, mas tem controle sobre os sistemas operacionais, armazenamentos e aplicativos implantados. IaaS é baseado em técnicas de virtualização. Seus recursos podem ser diminuídos ou aumentados de acordo com a necessidade do usuário. Os Exemplos de IaaS são o Amazon EC2, Flexiscale, GoGrid, Eucalyptus, Tecla Internet, Cloud Server Locaweb, dentre outros.

A Figura 5 mostra a separação das responsabilidades de cada plataforma.

Configuração Local	IaaS	PaaS	SaaS
Aplicações	Aplicações	Aplicações	Aplicações
Intermediárias	Intermediárias	Intermediárias	Intermediárias
Framework de aplicação	Framework de aplicação	Framework de aplicação	Framework de aplicação
Sistema Operacional	Sistema Operacional	Sistema Operacional	Sistema Operacional
Virtualização	Virtualização	Virtualização	Virtualização
Hardware	Hardware	Hardware	Hardware
Conectividade	Conectividade	Conectividade	Conectividade
Datacenter	Datacenter	Datacenter	Datacenter

Laranja = Gerenciado pelo cliente | Azul Marinho = Gerenciado pelo fornecedor de Nuvem

Fig. 5 – Modelos de serviços da computação em nuvem

Fonte: http://www.solidq.com/sqj/pt/Documents/2010_August_Issue/SQJ%20002.pdf

As plataformas de serviços se suportam e se completam de acordo com uma hierarquia. Cada uma delas tem um responsável diferente pela sua gerencia, isso fica bem ilustrado na Figura 6.

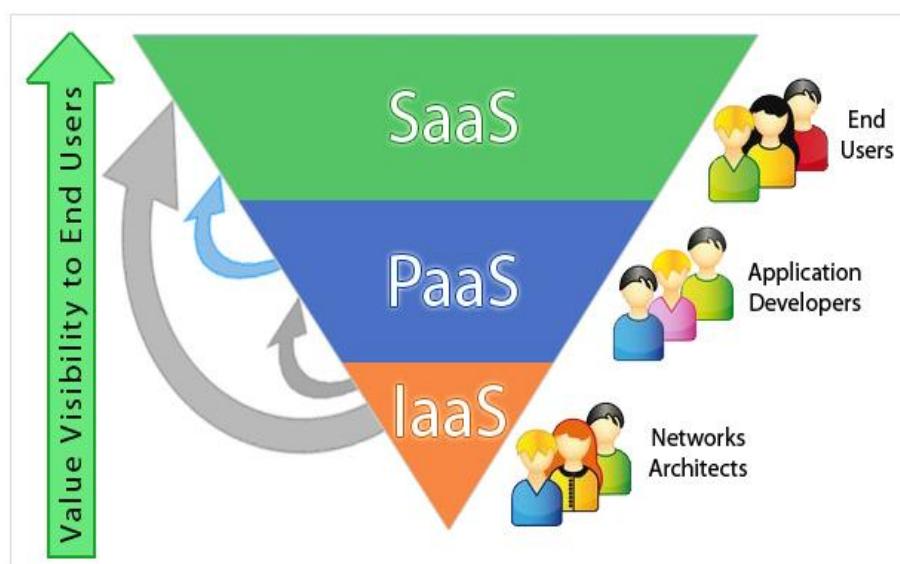


Fig. 6 – Modelos de serviços

Fonte: <http://www.qarea.com/articles/cloud-computing-outlook-iaas-paas-and-saas>

O provedor é responsável por disponibilizar, gerenciar e monitorar toda a estrutura da computação em nuvem, deixando o desenvolvedor e o usuário final sem essa responsabilidade e fornecendo serviços nos três modelos. Os desenvolvedores utilizam os recursos fornecidos e disponibilizam serviços para os usuários finais.

3.7 Modelos de Implantação

Existem alguns modelos para se seguir na hora de implantar o cloud computing, eles devem ser analisados e escolhidos cuidadosamente. Eles são divididos em três tipos e descritos a baixo (ASHLEY, 2009).

- Nuvens públicas são serviços de nuvem prestados por um terceiro (fornecedor) que existe além do *firewall*¹³ da empresa e são hospedados e gerenciados pelo provedor da nuvem. São ainda responsáveis pelo provisionamento, instalação e manutenção sejam do *software*, da infraestrutura de aplicação ou da infraestrutura física. Estes serviços são oferecidos como ‘convenção sobre configuração’, que significa que são entregues com a ideia de acomodar os casos de uso mais comuns. Outro ponto a se analisar é que, como os consumidores têm pouco controle sobre a infraestrutura, os processos que exigem estrita segurança não são aconselhados para o uso em uma nuvem pública.
- Nuvens privadas são serviços de nuvem prestados dentro da empresa. Estas nuvens existem dentro do *firewall* e são gerenciadas pela mesma. As nuvens privadas oferecem muitos dos mesmos benefícios que as nuvens públicas, porém nesse modelo, a empresa é responsável pela instalação e manutenção da nuvem. Nesse caso a grande vantagem é a segurança e a personalização da nuvem de modo a atender completamente as necessidades da empresa, além de oferecerem um controle mais detalhado sobre os vários recursos que a constituem. Em contrapartida a dificuldade de se fazer isso e o custo de se estabelecer uma nuvem interna podem ser um obstáculo. O custo da utilização contínua da nuvem pode ser maior que o de se usar uma nuvem pública. As nuvens privadas são ideais quando o trabalho que está sendo feito, por motivos de segurança ou preocupações regulamentares não é indicada para a nuvem pública.
- Nuvem híbrida é uma combinação de nuvens públicas e privadas. Estas nuvens são criadas pela empresa, e a responsabilidade da administração fica dividida entre a empresa e o provedor de nuvem pública. A nuvem híbrida aproveita serviços que estão tanto no espaço público quanto privado, por isso é a solução quando a empresa precisa tanto dos serviços da nuvem pública quanto da privada. Uma nuvem híbrida bem

¹³ Firewall é um *software* que tem como função proteger uma rede do acesso de pessoas ou outros *softwares* não autorizados.

construída poderia atender tanto os processos críticos, como aqueles que são secundários para o negócio. A grande desvantagem deste modelo é a dificuldade em se criar e administrar, pois serviços de diferentes fontes devem ser disponibilizados como se fossem originados de um único local. Deve ser levado em consideração que este é um modelo relativamente novo e as melhores práticas e ferramentas continuam a surgir. A disposição desses modelos é ilustrada na Figura 7.

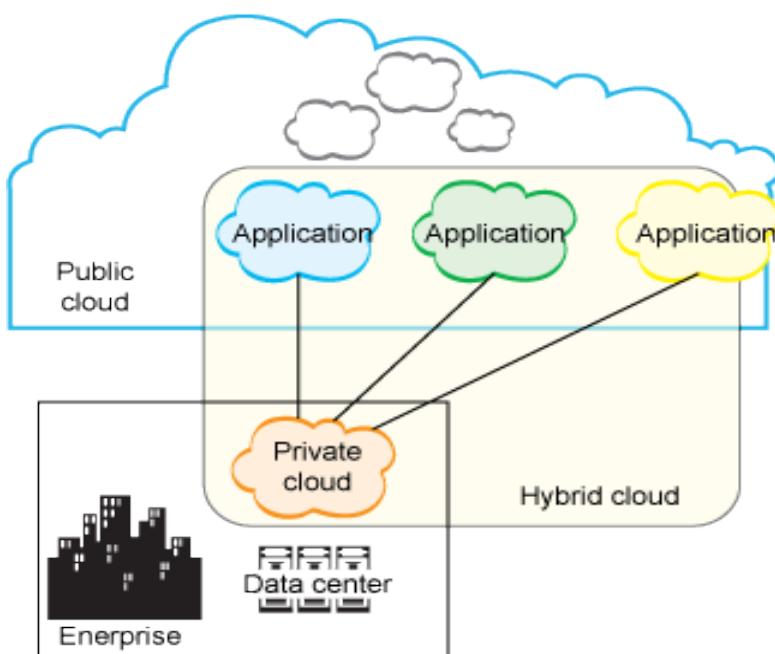


Fig. 7 – Modelos de implantação

Fonte: <http://www.ibm.com/developerworks/br/cloud/library/cl-hybridcloud1/>

Os modelos de implantação permitem ao usuário usar as tecnologias disponibilizadas na nuvem de acordo com suas necessidades, porém é necessária uma análise cautelosa para definir qual é o modelo mais apropriado para as necessidades do negócio, principalmente em questões de segurança.

3.8 Segurança

A segurança tem uma grande importância no cloud computing e a popularização da computação em nuvem está diretamente relacionada com a confiança que se tem no modelo e nas tecnologias envolvidas (TCLBRAZIL, 2010).

O fato de o cloud computing ainda ser um conceito e um modelo computacional em crescimento torna fundamental a atenção às evoluções das tecnologias e conceitos que o formam e o sustentam. Essa evolução, ainda prematura, do modelo de cloud computing pode levar gestores de TI a tomarem decisões com algum grau de risco, pois nem sempre as tecnologias envolvidas estão no nível de maturidade que seria desejável, não permitindo que se tenha certeza do resultado final. Por esses motivos, a segurança se torna um ponto crítico que deve ser muito bem analisado (TAURION, 2010).

3.8.1 Controles de segurança

Existem alguns controles de segurança, necessários para tornar um ambiente de computação em nuvem seguro de forma correta (CASES, 2010). Os principais controles seguem descritos a seguir:

- Gerenciamento de Ativos: Deve ser possível gerenciar todos os *hardwares*, redes e *softwares*, físicos ou virtuais e fornecer acesso a eles para auditorias e verificações de rotina.
- Criptografia: Gerenciamento de chaves e de certificados é muito importante, pois eles serão os responsáveis por garantir ou negar o acesso aos dados de pessoas não autorizadas.
- Segurança de dados/armazenamento: É muito importante que o provedor seja capaz de armazenar dados criptografados e também saber que alguns usuários precisarão armazenar seus dados em locais separados dos dados de outros clientes
- Segurança do terminal: Os usuários devem ter o poder de gerir a segurança de suas instâncias criadas na nuvem.
- Relatório e auditoria: Um dos fatores de segurança mais importantes é a capacidade de auditoria para se conseguir detalhes de um fato ocorrido, como falhas de sistema, invasões e ataques diretos.
- Segurança de rede: É necessário garantir o tráfego de rede em nível de comutador¹⁴, roteador¹⁵ e pacote.

¹⁴ Comutadores são semelhantes aos hubs, mas podem identificar o destino e enviam essas informações apenas para os computadores que devem recebê-las.

- Políticas de segurança: Para tornar o controle de acesso e a alocação de recursos eficaz, é preciso definir, resolver e implantar políticas de segurança de forma consistente.
- Automação de serviços: Possuir uma forma automatizada de gerenciar e analisar os fluxos e processos de controle de segurança para auxiliar as auditorias é extremamente importante para que as auditorias sejam feitas com maior agilidade.

Na escolha de um provedor para serviços de cloud computing, os controles de segurança devem ser levados em conta, o provedor deve dar suporte a esses controles permitindo ao usuário uma forma confiável de trabalhar e gerenciar o acesso aos dados.

3.8.2 Padrões de segurança da federação

Federação é o conceito que torna a computação em nuvem possível, pois ela é a capacidade de vários recursos independentes (ativos, identidades, configurações, etc.) se comportarem como um único recurso (CASES, 2010).

Uma autoridade de autenticação fornece para as duas partes uma relação de confiança, pois a empresa que presta os serviços deve trocar credenciais de segurança (certificados) e criar *tokens*¹⁶ assinados digitalmente usando essas credenciais, formando assim uma base segura para todos os padrões de federação (CASES, 2010).

O gerenciamento de identidades é o padrão que permite definir um provedor de identidade que possa aceitar as credenciais do usuário (ID, senha, certificado, etc.) e retornar um *token* de segurança assinado para o ID¹⁷ daquele usuário, fazendo com que o provedor de serviço permita o acesso.

O gerenciamento de acesso permite que políticas sejam gravadas para examinar *tokens* de segurança.

A auditoria e conformidade faz com que auditorias de dados sejam difundidas em diversos domínios para documentar a conformidade com acordos de serviço e outros regulamentos.

¹⁵ Roteador é um dispositivo que encaminha pacotes de dados entre redes de computadores.

¹⁶ Token é um dispositivo que auxilia quanto à segurança ao gerar uma senha temporária.

¹⁷ ID – Abreviação de identidade.

O gerenciamento da configuração é a capacidade de federar dados de configuração para serviços, aplicativos e máquinas virtuais.

Como as melhores práticas costumam se tornarem padrões, é importante que um desenvolvedor observe as que já existem, pois assim estará usando-as e cumprindo os padrões de federação.

3.8.3 SLA

O Service Level Agreement (SLA) ou contrato em nível de serviço é um contrato, negociado entre as partes, para um serviço de tecnologia da informação ou de telecomunicações e tem por objetivo especificar os requisitos mínimos como a qualidade do serviço, critérios de cobrança, provisionamento, processo de atendimento e relatórios fornecidos ao cliente que deverão ser cumpridos na entrega do serviço. O não cumprimento do acordo pode acarretar em penalidades para o provedor de acordo com o contrato (TUDE, 2003).

Os contratos SLA não existem com o objetivo de melhorar a qualidade do serviço e sim para servir como base legal para um eventual problema jurídico. Mais importante do que a qualidade do serviço imposta no contrato, deve-se exigir soluções para diminuir o tempo de recuperação da aplicação no caso de queda do servidor, pois isso pode trazer grandes prejuízos (GOLDEN, 2011).

3.8.4 Fatores de segurança na migração para nuvem

Realizar uma migração de uma aplicação para a nuvem pode ser uma tarefa complicada, pois a segurança nessa área traz muitos desafios, mas muitos deles podem ser superados quando se tem um planejamento de migração. Para isso é preciso ter conhecimento de quais os serviços que podem se aproveitar da migração para a nuvem e quais os problemas de segurança, privacidade e regulamentação podem ocorrer com a migração do serviço (ALLIANCE, 2012).

Existem ocasiões em que se pode optar por não realizar a migração por questões legais, em situações que exijam que os dados fiquem localizados em um *data center* no país

de origem, deverá ser realizado um estudo referente ao assunto, pois como a legislação varia conforme o país em que os dados estão hospedados, a migração para nuvem jamais deverá ser feita sem que esse aspecto seja analisado (ALLIANCE, 2012).

A maior causa de migrações fracassadas é a falta de um planejamento inicial e a não compreensão das implicações de se realizar essa migração. O modelo de serviço que se escolhe para a migração deve ser muito bem estudado, pois dependendo da situação, uma total readaptação precisa ser feita (ALLIANCE, 2012).

Em uma migração que inclua o desenvolvimento de novas aplicações já na nuvem, não só os serviços para o usuário final serão afetados, mas também a plataforma de desenvolvimento.

3.8.5 Outras questões de segurança

Mesmo que todo o contrato seja revisado e estudado por todos da empresa, no fim o contratante tem que confiar que este contrato será respeitado, pois a partir da assinatura, todos os dados da empresa serão manuseados por pessoas com quem o contratante provavelmente nunca terá contato (ALLIANCE, 2012).

Durante o processo de escolha do provedor é importante entender previamente alguns aspectos como a liberação de *logs*, pois caso o contratante necessite de algum tipo de investigação nos seus dados ele precisa saber qual a política do provedor para liberar essas informações (ALLIANCE, 2012).

É preciso ter um conhecimento prévio das condições de liberação desses dados, pois se houver a necessidade de uma ordem judicial, é preciso saber por quanto tempo esses *logs* ficam armazenados ou ainda se existe uma garantia de que eles não serão mesclados com os de outros clientes (ALLIANCE, 2012).

É necessário também lembrar de que não basta ter certificações para serviços SLA¹⁸ se a equipe interna que gerencia o *data center* não possui um treinamento adequado e não segue um padrão de segurança para acesso dos dados (ALLIANCE, 2012).

É de extrema importância que o contratante do serviço tenha um conhecimento profundo da política de segurança usada pelo provedor e saber onde ficam armazenados os dados fisicamente. Não é necessário saber o endereço do *data center*, mas precisa-se saber a

¹⁸ SLA – Abreviação para “Service Level Agreement”.

localização geográfica do mesmo, pois em caso de falha desse servidor, os dados podem ser transferidos para fora do país de origem, com leis diferentes e por isso o contratante fica sujeito a regras diferentes do seu país de origem (ALLIANCE, 2012).

Além de todas essas preocupações contratuais e com o manuseio dos dados, outro ponto importante são os empregados mal intencionados ou despreparados que são responsáveis por 80% de ataques maliciosos dentro das empresas, sendo mais nocivos que usuários de fora dessa rede (GRIMES, 2010)

O risco aumenta se o contratante não sabe exatamente qual o tipo de treinamento que os funcionários recebem e qual o tipo de punição dada para infrações no possível vazamento de dados (ALLIANCE, 2012).

Por fim os dados são de propriedade do cliente e isso tem que ser explícito em contrato, pois o manuseio será realizado pelo provedor de serviços da nuvem, mas os dados sempre serão do cliente. Para evitar futuras complicações, o contratante tem que exigir uma documentação contratual quanto a essa propriedade final dos dados ALLIANCE, 2012).

3.9 Plataformas de Desenvolvimento

Aqui serão abordados os principais servidores que oferecem esse modelo computacional como serviço, explicando seu funcionamento e mostrando as principais ferramentas oferecidas.

3.9.1 Windows Azure

O Windows Azure é uma plataforma que suporta várias linguagens e atua como um ambiente de desenvolvimento, hospedagem de serviços e gerenciamento, fornecendo uma computação sob demanda e armazenamento para hospedar, escalar e gerenciar aplicações *web* suportando SOAP¹⁹, XML²⁰ e PHP²¹ dentre outros (MICROSOFT, 2010).

¹⁹ SOAP – Sigla de “Simple Object Access Protocol”.

²⁰ XML – Sigla de “Extensible Markup Language”.

²¹ PHP – Sigla de “Hypertext Preprocessor”.

O Azure é uma plataforma como serviço (PaaS) que disponibiliza as principais funcionalidades como processamento, armazenamento, gerenciamento e administração das aplicações, além do suporte ao modelo de software como serviço (SaaS). Para complementar essa plataforma, existem 3 serviços que podem ser considerados os mais importantes, como o SQL Azure, Windows Azure AppFabric e o Windows Azure Storage. Esses serviços podem ser utilizados pelas aplicações hospedadas na plataforma principal, como é ilustrado na Figura 8 (CAMBIUCCI, 2011).

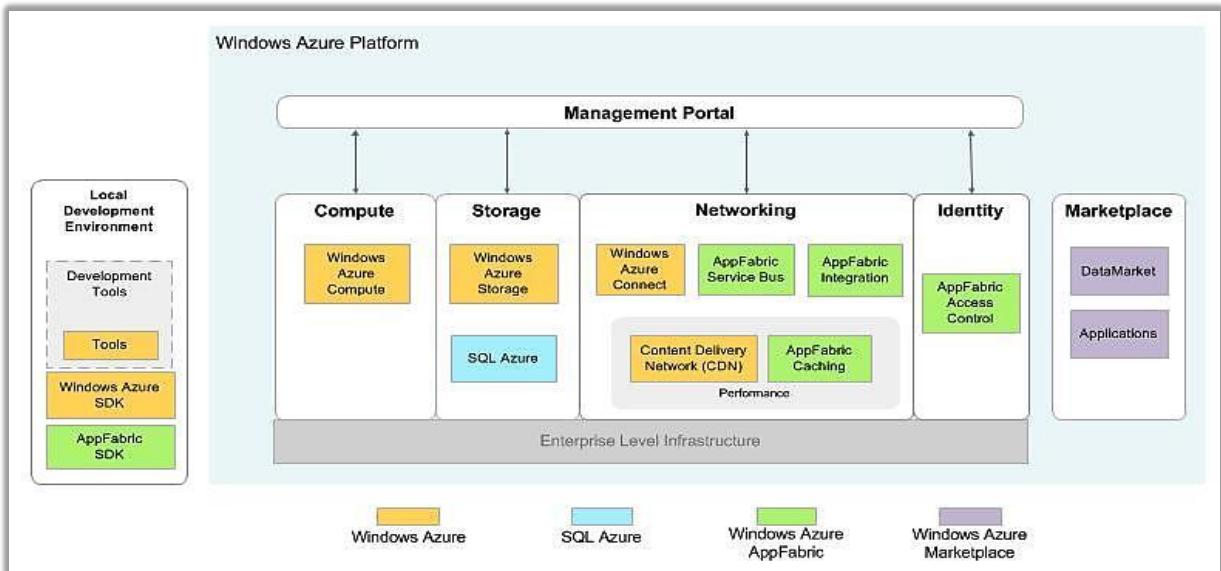


Fig. 8 – Principais componentes e serviços do Windows Azure
Fonte: <http://msdn.microsoft.com/pt-br/library/hh150078.aspx>

Para compreender melhor é necessário entender o funcionamento do Azure Services. Ele é um conceito que faz o isolamento entre os processos, utilizando as *roles* que são as instâncias da máquina virtual e são classificadas em três tipos (CAMBIUCCI, 2011).

- *Web Role* - Instância que recebe e trata requisições http. Cada *web role* é executado sobre o *Internet Information Services* (IIS). As requisições são feitas para o mesmo DNS e são enviadas para o *load balancer* (balanceador de carga), que é responsável por distribuir as requisições pelos *web roles* da aplicação, não dividindo o processamento, pois cada instância é executada em uma máquina virtual diferente. Quando um *web role* recebe uma requisição que demanda muito processamento essa tarefa é enviada para o *worker role* através de uma *queue* (fila).

- *Worker Role* – Tem como função processar todos os pedidos armazenados na fila pelos *web roles*, sendo o responsável por realizar as tarefas mais lentas. Pode em alguns casos conectar-se a outras instâncias.
- *VM Role* – São instâncias que dão acesso ao sistema operacional, pois representam a máquina virtual criada pelo usuário.

A Figura 9 representa o funcionamento mais comum das *roles*.

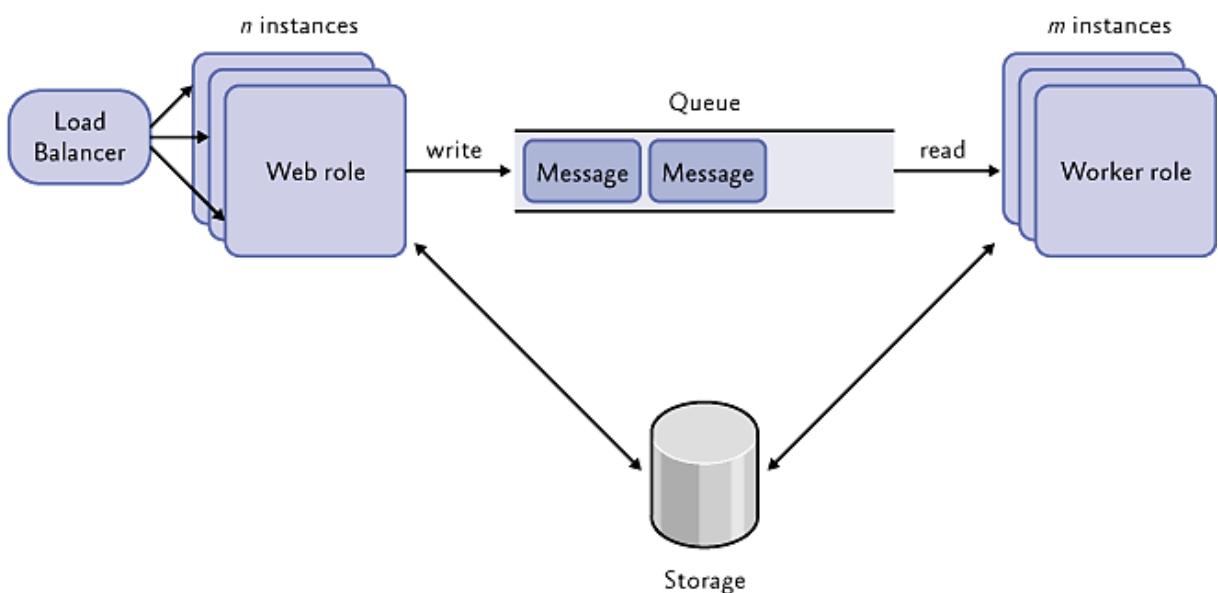


Fig. 9 – Funcionamento das roles

Fonte: http://www.gta.ufrj.br/ensino/eel879/trabalhos_vf_2011_2/valverde/NotesImages/Topic7NotesImage5.gif

Os *web roles* e *worker roles* permitem um maior grau de abstração onde o usuário se preocupa apenas com a aplicação, sem ser responsável por atualizações de infraestrutura ou sistema operacional. Se for necessário maior controle e customização o *VM role* permite que o usuário fique responsável por essas atualizações (CAMBIUCCI, 2011).

3.9.1.1 Azure Storage

Os serviços de *storage* do Azure fornecem acesso a um sistema escalável de armazenamento que mantém três cópias dos dados em segundo plano garantindo a disponibilidade dos dados (SOUZA, 2012). Existem três tipos de *storage* que são citados a seguir (CAMBIUCCI, 2011).

- *Blobs* -- São arquivos binários ou textos armazenados em *containers*. Podem conter *tags* que descrevem o arquivo ou adicionam informações sobre seu conteúdo. Cada container pode conter um ou mais arquivos de qualquer tipo (vídeo, imagens, áudio, etc.), podendo ser público ou privado. A Figura 10 mostra o funcionamento dos *blobs*.

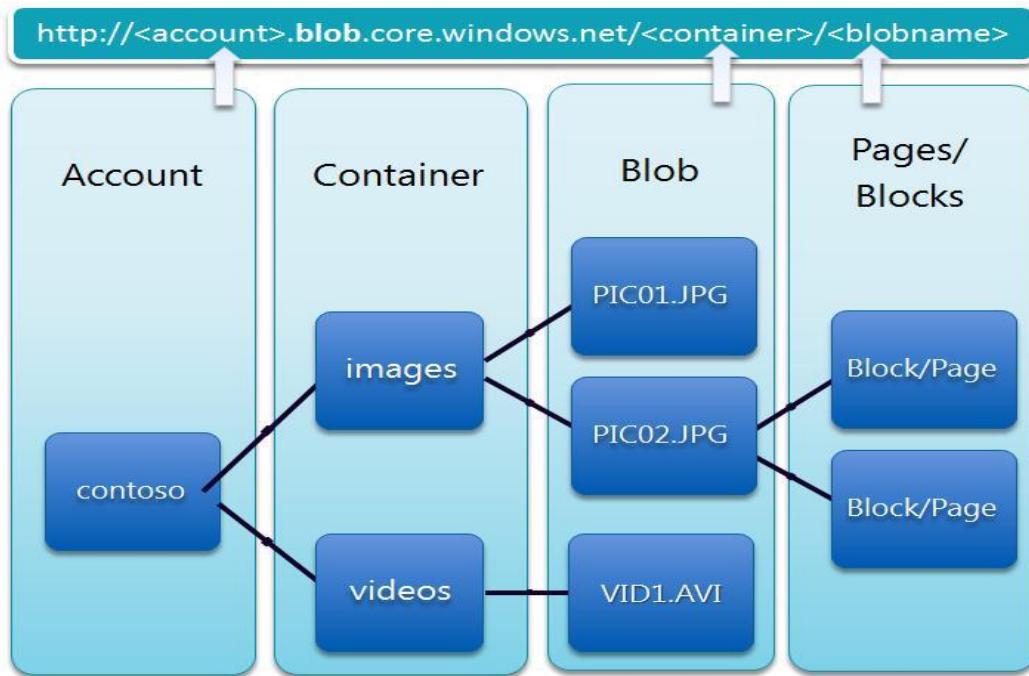


Fig. 10 – Funcionamento dos blobs **Fonte:** <http://i.microsoft.com/dynimg/IC501803.jpg>

- *Tables* – É uma persistência baseada em tabelas de forma não relacional. Para os tipos relacionais o Windows Azure oferece o SQL Azure. A Figura 11 mostra o funcionamento deste tipo de armazenamento.

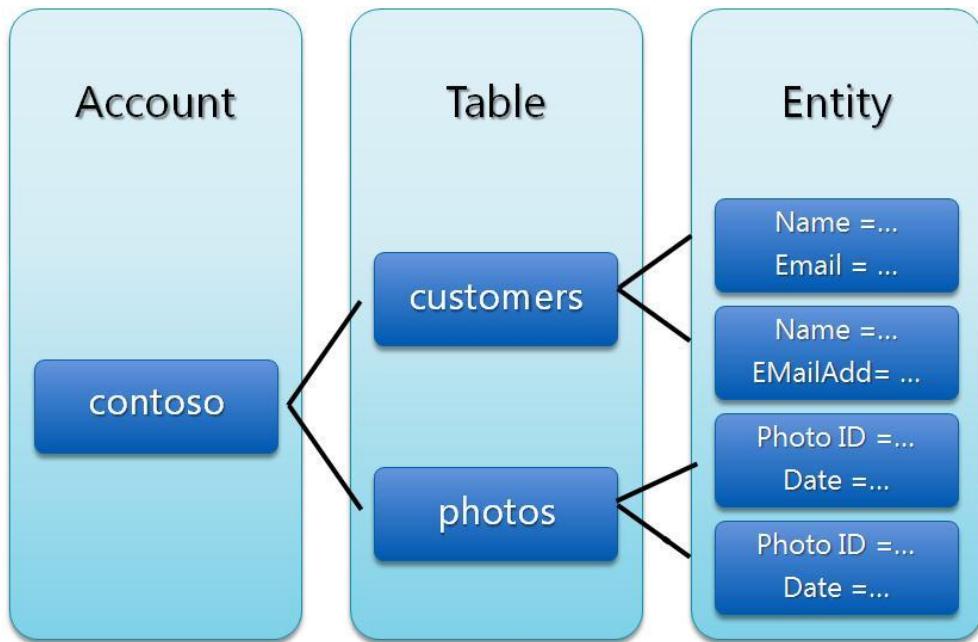


Fig. 11 – Funcionamento do *tables* **Fonte:** <http://i.microsoft.com/dynimg/IC501804.jpg>

- *Queues* – É um mecanismo de comunicação assíncrona que permite a comunicação entre processos, *web roles* e *worker roles* por meio de mensagens colocadas na fila. A Figura 12 mostra o funcionamento das *queues*.

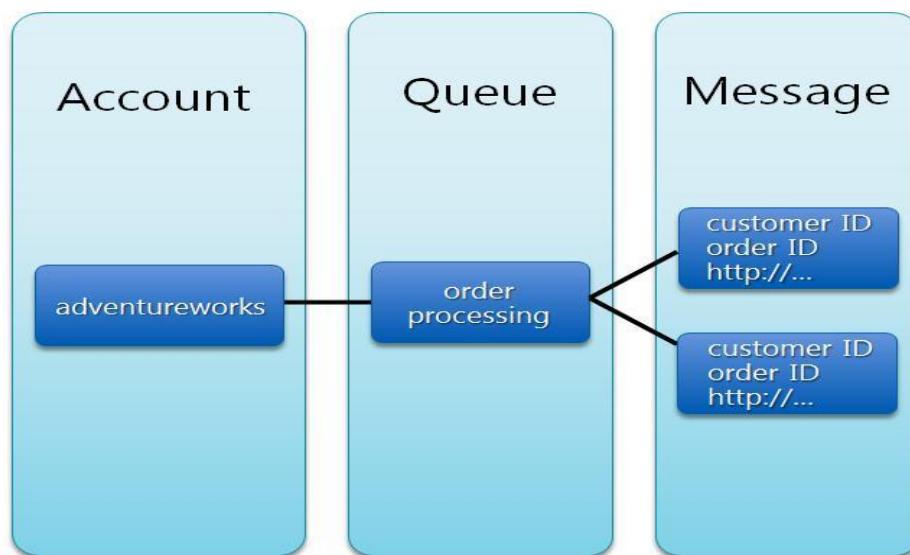


Fig. 12 – Funcionamento das *queues*
Fonte: <http://i.microsoft.com/dynimg/IC501804.jpg>

Existe ainda um quarto tipo de armazenamento chamado *Drive* que oferece uma unidade NTFS²², permitindo a manipulação de arquivos. Ele é montado a partir de um *blob* ampliando as funcionalidades das aplicações na nuvem (CAMBIUCCI, 2011).

3.9.1.2 SQL Azure

O banco de dados do SQL Azure, permite que se crie, dimensione e estenda aplicativos na nuvem. Permite também a criação de *blobs*, que são um jeito simples de armazenar grandes volumes de texto ou dados binários, com dimensionamento automático para atender a grandes taxas de transferência e volumes (MICROSOFT, 2013).

As aplicações podem aproveitar recursos como persistência de dados, *business intelligence*²³, sincronização, replicação de dados, relatórios, *data mining*, dentre outras, a partir de uma infraestrutura de alta escalabilidade e provisionamento dinâmico (MICROSOFT, 2013).

3.9.1.3 Windows Azure AppFabric

O Windows Server AppFabric da plataforma Windows Azure permite que aplicações sejam criadas e gerenciadas de maneira fácil, ajudando os desenvolvedores a conectar aplicativos e serviços nas nuvens ou em um ambiente local. Isso inclui aplicativos executados no Windows Azure, Windows Server e várias outras plataformas, incluindo Java, Ruby, PHP e outros (MICROSOFT, 2013).

O *Service Bus* ajuda a fornecer conectividade segura entre serviços e aplicativos, permitindo navegar em *firewalls* ou nas fronteiras da rede e usar uma variedade de padrões de comunicação. Os serviços que se registram no *service bus* podem ser descobertos e acessados facilmente em qualquer parte da rede. (MICROSOFT, 2013).

O Microsoft Access Control ajuda a criar autorização federada para o seus aplicativos e serviços, sem a programação complicada que normalmente é necessária para proteger aplicativos que vão além dos limites organizacionais (MICROSOFT, 2013).

²² NTFS - New Technology File System

²³ *Business intelligence* – Tradução “inteligência de negócios” em inglês.

3.9.1.4 Windows Azure Virtual Network

O Virtual Network é um serviço que permite aumentar a capacidade das redes locais utilizando os serviços do Azure, fazendo com que ele seja uma extensão da rede com criação de máquinas virtuais, atribuição endereços e configuração de DNS²⁴ (CONDÉ, 2013). A Figura 13 ilustra esse serviço em funcionamento.

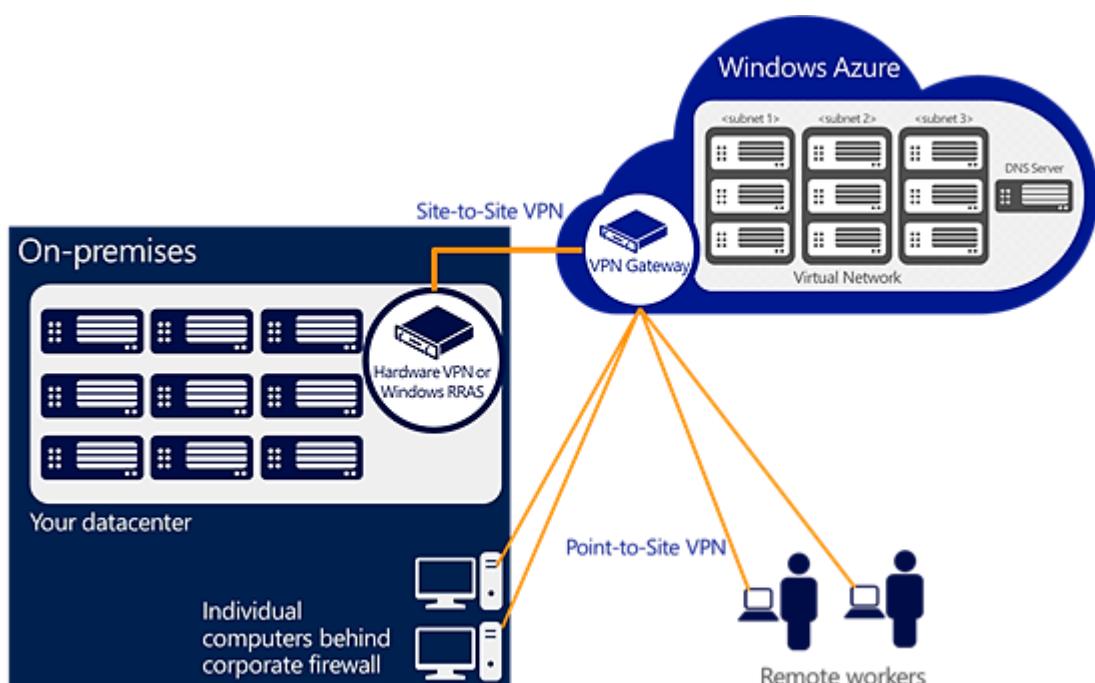


Fig. 13 – Visão geral do Azure Virtual Network

Fonte: <http://blogs.msdn.com/b/conde/archive/2013/05/06/an-250-ncio-virtual-networks-agora-suporta-windows-rras-e-conex-227-o-ponto-para-rede-virtual.aspx>

3.9.1.5 Windows Azure Traffic Manager

O Azure Traffic Manager permite controlar a distribuição do tráfego de acesso aos serviços do Azure, através de políticas de acessos (MICROSOFT, 2013). A Figura 14 exemplifica como é feito esse controle.

²⁴ DNS – Domain Name System

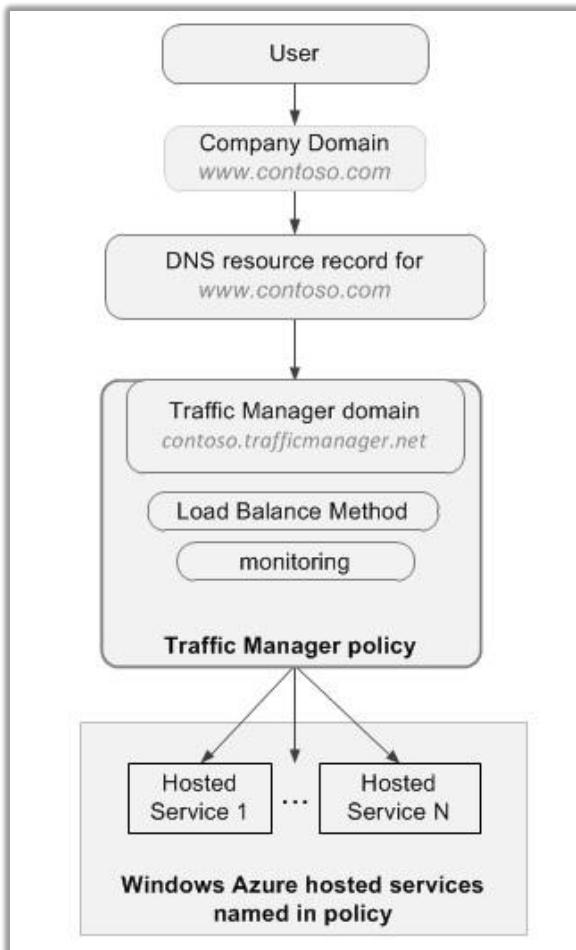


Fig. 14 – Roteamento do *Traffic Manager*

Fonte: <http://msdn.microsoft.com/pt-br/library/windowsazure/hh744833.aspx>

O usuário solicita o serviço através de um domínio, que aponta para o DNS do Traffic Manager que usa o método de平衡amento de carga previamente definido e o status do monitoramento para escolher qual serviço deve atender à solicitação. Depois de feita a escolha, ele resolve o DNS do serviço que irá atender a requisição e retorna o IP²⁵ que o usuário irá usar para ter o acesso direto (MICROSOFT, 2013).

3.9.2 AWS Amazon

A Amazon Web Services (AWS) oferece um conjunto completo de serviços de aplicativos e infraestrutura que permite executar desde aplicativos empresariais e grandes

²⁵ IP – Sigla de “Internet Protocol”.

projetos de dados a jogos sociais e aplicativos móveis, fornecendo uma ampla infraestrutura. (AMAZON, 2013). Os principais componentes desse servidor estão ilustrados na Figura 15.

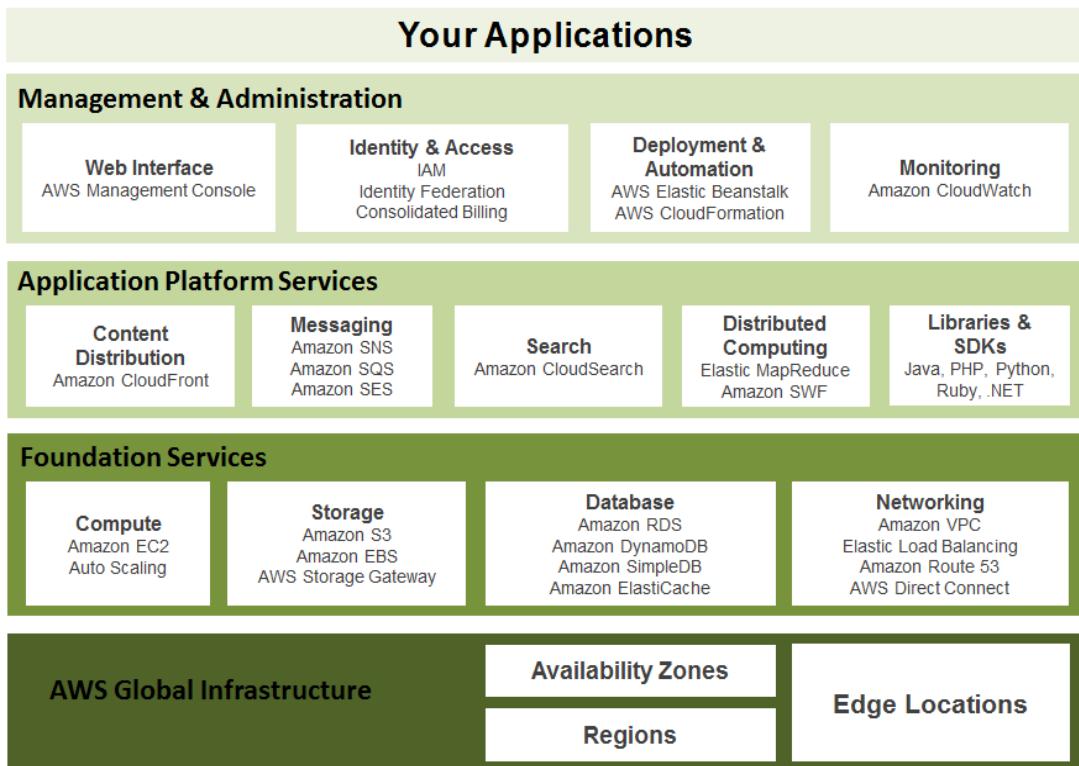


Fig. 15 – Principais componentes do AWS

Fonte: <http://docs.aws.amazon.com/gettingstarted/latest/awsgsg-intro/images/services.png>

A Amazon oferece serviços distribuídos nas seguintes categorias: computação/processamento, distribuição de conteúdo, banco de dados, instalação e gerenciamento de serviços, *e-commerce*, mensagens, monitoramento, rede, pagamento e faturamento, armazenamento, suporte, análise de tráfego *web* e força de trabalho (BARBOLO, 2011).

Pelo fato de a Amazon Web Services ser a pioneira e mais experiente do mercado decidiu-se utilizá-la como base para esse projeto. Outro motivo foi o fato de este provedor oferecer seus serviços de forma independente uns dos outros, diferente dos concorrentes que oferecem um conjunto de serviços para a contratação.

3.9.2.1 Elastic Compute Cloud (EC2)

O Elastic Compute Cloud (EC2) é um serviço que fornece recursos computacionais escaláveis e redimensionáveis de forma simples, permitindo que se altere esses recursos, para mais ou para menos, à medida que for necessário (AMAZON, 2013).

O EC2 é um ambiente de computação virtual que permite a utilização de *interfaces* de serviços *web* para iniciar instâncias de sistemas operacionais e gerenciar permissões de acesso de rede usando o número de sistemas que o usuário desejar.

3.9.2.2 Elastic MapReduce (EMR)

O Elastic MapReduce (EMR) é um serviço que permite às empresas, pesquisadores, analistas de dados e desenvolvedores processar, de um modo mais prático e econômico, grandes quantidades de dados (AMAZON, 2013).

O EMR utiliza o *framework*²⁶ Apache Hadoop que fica nos servidores do Simple Storage Service (S3) e do EC2. O EMR é mais utilizado para tarefas como indexação, mineração de dados, análise de *logs*, *data warehousing*²⁷, aprendizagem de máquina, análise financeira, simulação científica e pesquisas em bioinformática (BARBOLO, 2011).

3.9.2.3 Auto Scaling

O Auto Scaling permite que os servidores do EC2 aumentem ou diminuam os recursos automaticamente de acordo com definições estabelecidas previamente para a aplicação. Tendo a alocação de recursos em períodos de pico de forma automática como o principal benefício desse serviço (BARBOLO, 2011).

²⁶ Framework é uma abstração que une códigos comuns entre vários projetos de software provendo uma funcionalidade genérica.

²⁷ Data Warehousing é um sistema de computação utilizado para armazenar informações relativas às atividades de uma organização em bancos de dados.

3.9.2.4 CloudFront

O CloudFront é um serviço que permite distribuição de conteúdo com baixa latência e alta taxa de transferência, através de *caches* em provedores de *internet* de várias regiões do mundo, sendo utilizado para distribuir conteúdos estáticos ou *streamings* de mídias (BARBOLO, 2011).

Em um estudo de caso realizado na Frost & Sullivan, empresa multinacional que realiza consultoria de inteligência de mercado e é patrocinada pela Amazon, os resultados encontrados com o CloudFront estão ilustrados na Figura 16.

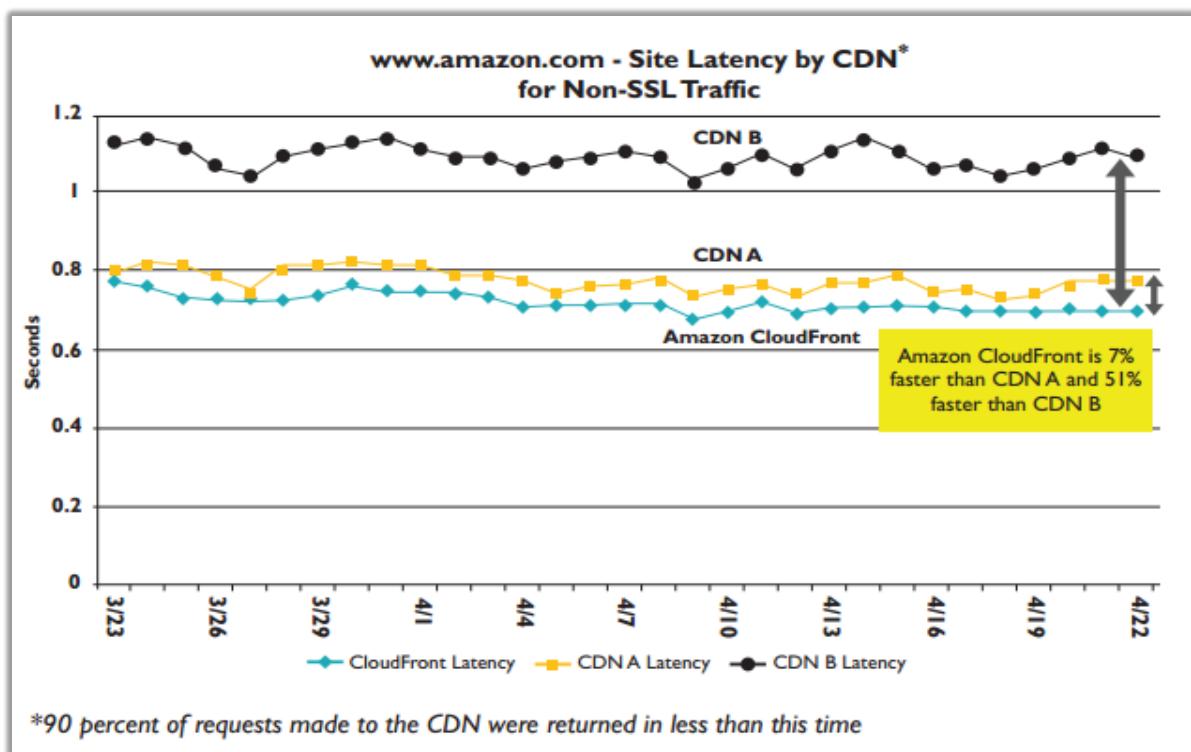


Fig. 16 – Estudo de caso com CloudFront

Fonte: http://media.amazonaws.com/FS_WP_AWS_CDN_CloudFront.pdf

Como nos mostra o resultado do estudo de caso, o CloudFront se mostrou muito mais eficiente do que os serviços das outras duas empresas (cujo os nomes não foram revelados) em que foram realizados os mesmos testes, mostrando assim ser uma boa opção de serviço para esse tipo de necessidade.

3.9.2.5 Simple DB (SDB)

O SimpleDB (SDB) é um armazenamento de dados altamente disponível, flexível e não relacional (AMAZON, 2013). O SDB foi criado para trabalhar em conjunto com o S3 e o EC2. Um exemplo é uma aplicação de compartilhamento de fotos e vídeos sendo executada no EC2, enquanto fotos e vídeos ficam armazenados no S3 e os comentários e opiniões dos usuários ficam no SDB (TAURION, 2009).

Esse serviço deve ser usado por aplicações que precisam apenas indexar e consultar dados, sem envolver transações ou comandos mais complexos (BARBOLO, 2011).

3.9.2.6 Relational Database Service (RDS)

O Relational Database Service (RDS) é um serviço que facilita a configuração e a implementação de um banco de dados relacional e os recursos são semelhantes aos do banco de dados MySQL (AMAZON, 2013). É possível configurar *backups* automáticos, adicionar ou remover recursos, e configurar réplicas que aumentam a confiabilidade e velocidade de acesso aos dados (BARBOLO, 2011).

3.9.2.7 Elastick Beanstalk

O Elastic Beanstalk permite implantar e gerenciar aplicações que estejam na nuvem de modo simples e rápido trabalhando automaticamente com os detalhes de distribuição da capacidade de provisionamento, balanceamento de carga, dimensionamento automático e monitorando a aplicação. Utiliza serviços como o EC2, S3, Simple Notification Service e o Auto Scaling (AMAZON, 2013).

3.9.2.8 Fulfillment Web Service (FWS)

O Fulfillment Web Service permite que um vendedor terceirize a entrega de produtos através da Amazon. Ele envia seus produtos para a Amazon, onde eles são estocados e catalogados, colocando-os como disponíveis. Quando um consumidor realiza a compra de um produto, a Amazon empacota o pedido e faz a entrega. Esse serviço permite que aplicações de vendedores se integrem com o serviço de entrega da Amazon (AMAZON, 2013).

3.9.2.9 Simple Queue Service

O Simple Queue Service (SQS) permite criar filas de mensagens no servidor de serviços com isso o desenvolvedor pode mover dados entre componentes distribuídos de seus aplicativos que executem tarefas diferentes, sem perder mensagens ou sendo necessário que cada componente esteja sempre disponível (AMAZON, 2013).

No SQS uma mensagem pode ser armazenada por até quatorze dias e não pode ser lida por mais de uma instância simultaneamente, sendo bloqueada enquanto é lida. Se o processamento da mensagem falhar, ela voltará a ficar disponível na fila e se for processada com sucesso ela será excluída da fila (BARBOLO, 2011).

3.9.2.10 Simple Notification Service

O Simple Notification Service (SNS) funciona como um serviço que facilita a configuração, operação e o envio de notificações a partir do provedor. O SNS é um serviço que permite enviar notificações facilitando o envio de mensagens via http²⁸, smtp²⁹ ou outro protocolo por aplicações, integrando módulos ou mantendo assinantes de conteúdo atualizados (AMAZON, 2013).

²⁸ HTTP é um protocolo de comunicação.

²⁹ SMTP é o protocolo padrão para envio de e-mails através da Internet.

3.9.2.11 Simple Email Service

O Simple Email Service (SES) é um serviço de envio de *e-mails* para empresas e desenvolvedores. Ele elimina o trabalho de criar uma solução de *e-mail* interna ou instalar e operar um serviço terceirizado, se integrando com outros serviços da AWS, facilitando o envio de *e-mails* de aplicativos hospedados (AMAZON, 2013).

3.9.2.12 CloudWatch

O CloudWatch permite monitorar o desempenho operacional e a utilização dos recursos de aplicativos que estejam sendo executados na nuvem. As versões mais simples são gratuitas, já as com maiores detalhes de monitoramento são taxadas (AMAZON, 2013).

3.9.2.13 Route 53

O Route 53 é um serviço de consulta de DNS³⁰ que resolve endereços de domínios públicos. Com ele é possível instâncias do EC2 ou no S3 e direcionar o acesso de um domínio externo a esses serviços, facilitando o acesso a eles (AMAZON, 2013).

3.9.2.14 Virtual Private Cloud

O Virtual Private Cloud (VPC) é um serviço que permite aproveitar uma seção privada e isolada da nuvem onde é possível executar recursos em uma rede virtual. Possibilita também definir uma topologia de rede que se assemelhe a uma rede real, com o controle total sobre o ambiente de rede virtual, incluindo a seleção do seu próprio intervalo de endereços IP³¹,

³⁰ DNS é um sistema de gerenciamento de nomes hierárquico.

³¹ IP é uma identificação de um dispositivo (computador, impressora, etc) em uma rede local ou pública.

criação de sub-redes e configuração de tabelas de roteamento e *gateways*³²de rede (AMAZON, 2013).

3.9.2.15 Elastic Load Balancing

O Elastic Load Balancing é o serviço que distribui requisições de acesso para as aplicações acessarem as instâncias ativas do EC2, balanceando a carga pelos diversos servidores, tornando possível aumentar a tolerância a falhas, a velocidade de acesso e a disponibilidade de uma aplicação (BARBOLO, 2011).

3.9.216 Simple Storage Service

O Simple Storage Service (S3) é um armazenamento para a *internet*, oferecendo quatro mecanismos de controle de acesso diferentes como o Identityand Access Management (IAM), Access Control Lists (ACLs), políticas de *bucket*³³ e autenticação de sequência de caracteres de consulta (AMAZON, 2013).

Ele também possui armazenamento de dados para várias instalações e dispositivos, armazenando de forma síncrona seus dados em várias instalações antes de confirmar a troca de dados (AMAZON, 2013).

3.9.2.17 Amazon Elastic Block Store

O Amazon Elastic Block Store (EBS) fornece volumes de armazenamento em bloco. Eles são vinculados à rede e podem ou não ser vinculados a uma instância do EC2 em execução e expostos como um dispositivo dentro da instância (AMAZON, 2013).

³² *Gateway* – é uma máquina intermediária geralmente destinada a interligar redes, separar domínios de colisão, ou mesmo traduzir protocolos.

³³ *Bucket* – é um objeto abstrato cujo papel é conter dados próprios de um usuário.

3.9.2.18 Import/Export

O Import/Export é um serviço usado para transferência de uma grande quantidade de dados para os servidores da Amazon. A partir de um dispositivo de armazenamento os dados são enviados de um dispositivo para os servidores ou dos servidores para o dispositivo. Este serviço é considerado relevante quando a transferência de dados pela internet é muito intensa e demorada (BARBOLO, 2011).

3.9.2.19 Suporte Premium

O Suporte Premium atua como um canal de suporte fornecido em tempo integral por engenheiros de suporte técnico. O serviço ajuda clientes a utilizarem os produtos e recursos fornecidos de forma correta (AMAZON, 2013).

3.9.2.20 RedShift

Até a data de pesquisa deste projeto, o Amazon Redshift se encontra em versão *beta*. Ele é um serviço para processamento de dados que variam de *gigabytes* até *petabytes* com custos abaixo de mil dólares por *terabyte* por ano, ou seja, um décimo do custo das soluções tradicionais de *data warehouse*³⁴.

É possível utilizar o Redshift diretamente com o S3 e com o DynamoDB para retirar dados do MapReduce, RDS e dos bancos de dados do EC2, permitindo combinar e analisar dados de várias fontes (AMAZON, 2013).

³⁴ *Data Warehouse* - Depósito de dados digitais que serve para armazenar informações detalhadas relativamente a uma empresa, criando e organizando relatórios.

3.9.2.21 OpsWorks

O OpsWorks se encontra em sua versão *beta* até a data em que essa pesquisa foi feita. Ele é uma plataforma para o gerenciamento de qualquer escala ou complexidade na nuvem, oferecendo uma facilidade no gerenciamento do ciclo de vida dos aplicativos, incluindo provisionamento de recursos, gerenciamento de configuração, implementação de aplicativos, atualizações de *software*, monitoramento e controle de acesso (AMAZON, 2013).

O OpsWorks permite modelar e visualizar o aplicativo em camadas que definem como configurar um conjunto de recursos gerenciados simultaneamente. Também é possível definir a configuração de *software* para cada camada, como *scripts* de instalação e tarefas de inicialização (GARCIA, 2013).

Ele é muito parecido com o Elastic Beanstalk, onde ele oferece *containers* (IIS, Node.JS, PHP, Ruby, Python e Tomcat) e é preciso customizar esses *containers*, porém existem limites para isso (GARCIA, 2013).

O OpsWorks, tem outro conceito (*stacks* e *layers*), em que é possível criar e configurar *containers* como o necessário. Gerenciando um ambiente com dezenas, centenas ou até milhares de servidores, o OpsWorks é como uma "receita de bolo", uma vez feita e funcional, é possível replicá-la quantas vezes for preciso (GARCIA, 2013).

3.9.2.22 Elastic Transcoder

O Amazon Elastic Transcoder é a transcodificação de vídeo na nuvem. Ele foi desenvolvido para ser uma forma escalável para que desenvolvedores e empresas convertam arquivos de vídeo de seu formato de origem em versões que serão reproduzidas em dispositivos como *smartphones*, *tablets* e computadores. Com ele é possível adicionar Marca d'água visual, parâmetros de taxa máxima de bits do vídeo, parâmetros de taxa máxima de quadros dentre outros (AMAZON, 2013).

3.9.3 Google App Engine

O Google App Engine (GAE) é uma plataforma como serviço que fornece um ambiente de desenvolvimento para que aplicações possam ser desenvolvidas e implementadas na nuvem, acabando com a necessidade de manter servidores ativos. O ambiente de desenvolvimento contém suporte ao Java, permitindo a criação do aplicativo usando tecnologias Java padrão, incluindo JVM³⁵, *servlets*³⁶ Java ou qualquer outra linguagem que utilize um compilador com base na JVM. O GAE também disponibiliza um ambiente para desenvolvimento em Python, PHP e GO³⁷, essas duas últimas de forma experimental (GOOGLE, 2013). A Figura 17 ilustra a arquitetura de um sistema GAE.

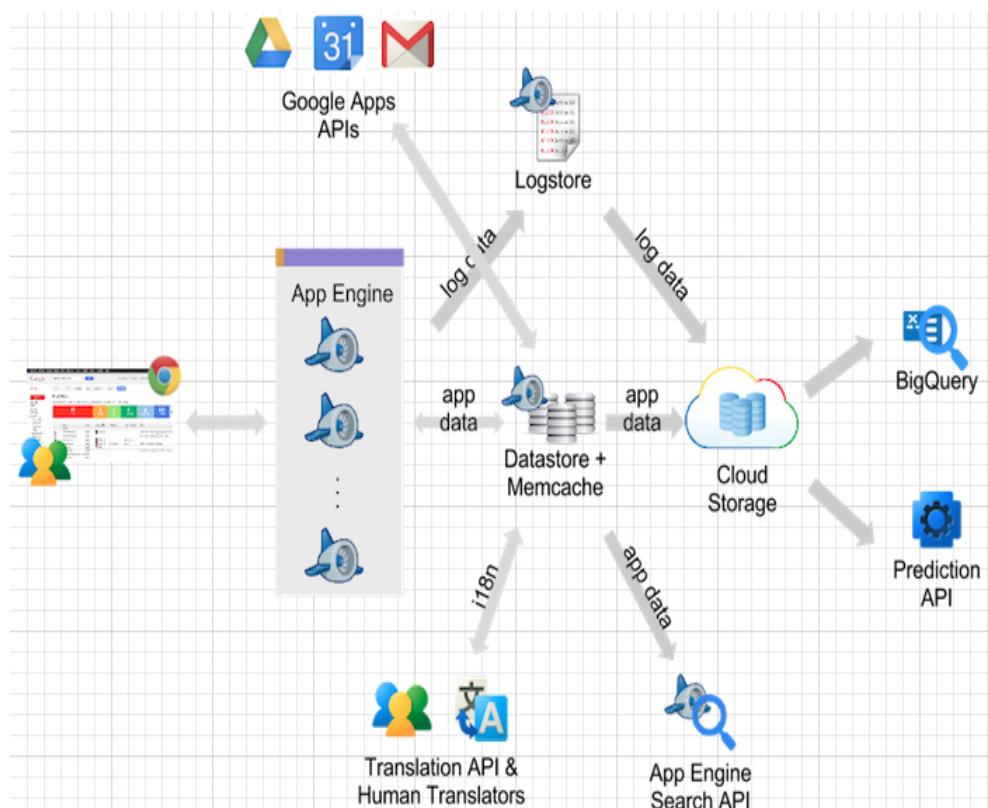


Fig. 17 – Arquitetura de um sistema no GAE.

Fonte: <http://thecustomizewindows.com/wp-content/uploads/2012/12/PaaS-Model-and-Architecture-of-Google-App-Engine.jpg>

³⁵ JVM – Java Virtual Machine.

³⁶ *Servlets* – é um componente do lado servidor que gera dados HTML e XML para a camada de apresentação de um aplicativo web.

³⁷ GO – Linguagem de programação lançada em 2009 pela Google.

No GAE não se tem a transparência de onde sua aplicação foi implementada, ao contrário do AWS onde se tem a liberdade de escolha dos servidores a serem utilizados. As interações com essa plataforma são feitas através de *servlets* e chamadas a recursos externos a sua aplicação, podem ter duração de no máximo trinta segundos por questões de escalabilidade (GUIMARÃES, 2012).

3.9.3.1 Serviços de Storage

O Google App Engine fornece algumas formas para que os armazenamentos de informações e arquivos sejam feitos afim de serem utilizados pelas aplicações hospedadas no GAE que são descritas a seguir (GUIMARÃES, 2012).

- GAE Blobstore - Serviço de armazenamento fornecido através de uma biblioteca própria, onde chaves de identificação apontam o arquivo. O local onde ele se encontra armazenado não é conhecido. Esse não é um serviço que possa ser contratado a parte, pois ele já vem incluso na plataforma de desenvolvimento do GAE, diferentemente dos serviços da AWS.
- GAE Datastore – É uma solução NoSql que utiliza um sistema chamado “*Big Table*” que consiste em armazenar os registros de todas as aplicações em uma única tabela gigante, em que as entidades são representadas pelo tipo da informação armazenada, por um identificador e seu conteúdo. Existem regras de segurança por trás desse sistema que permite apenas que as aplicações acessem suas informações que lá estão armazenadas.
- GAE CloudSql – Solução que permite utilizar bancos de dados relacionais, permitindo criar, importar e exportá-los. Oferece um ambiente *MySQL* com acesso para as plataformas Java (via JDBC) e Python (via DB-API). As operações de administração, *backup* e escalonamento são feitos automaticamente.

3.9.3.2 Google App Engine Task Queue

O GAE Task Queue é um serviço que permite que sejam criadas filas de processamento em segundo plano. A plataforma aloca o quanto for necessário de recursos para realizar o que foi configurado na *task*. O serviço é gratuito, porém o tempo de processamento utilizado para a realização da tarefa será cobrado (GUIMARÃES, 2012).

3.10 Estudos de caso

Para demonstrar algumas possibilidades que já são utilizadas por grandes empresas, alguns estudos de casos serão utilizados para saber como o cloud computing beneficiou o modo como elas gerenciam seus processos e entregam seus serviços.

3.10.1 Grupo ING

O grupo ING é uma organização financeira que atua em todo o mundo, atende a 85 milhões de clientes privados, corporativos e institucionais contando com uma mão de obra em torno de 107 mil pessoas em mais de quarenta países na Europa, América do Norte, América Latina, Ásia e Austrália e possui 1.261 trilhão de euros em ativos. O objetivo do ING é ser o banco preferencial para seus clientes, sendo a primeira escolha para todas as necessidades bancárias e financeiras (ING, 2013).

O ING definiu sua estratégia para economias substanciais nos custos e, ao mesmo tempo proporcionar um maior controle sobre infraestrutura de TI, maior estabilidade, e uma melhor organização da empresa. Para isso, o ING percebeu que o papel dos profissionais de TI tinha que mudar com a adoção de um novo conceito para a gestão dessa área, e que a nuvem tem um impacto sobre todas as áreas da empresa. Com isso foi dado um treinamento com os princípios básicos da virtualização e computação em nuvem, garantindo que todos os envolvidos tivessem habilidade para permitir que a organização conseguisse realizar essa transição com sucesso (ING, 2013). Alguns pontos que pretendiam ser alcançados com isso são:

- Transformar a área de TI de um departamento tradicional para um ‘provedor de serviços’ da própria empresa;
- Conseguir um baixo custo com o fornecedor de cloud computing;
- Reduzir o risco e a complexidade que existe quando os dados não estão todos no mesmo local;
- Conseguir a virtualização de plataformas, visando o avanço da tecnologia;
- Construir um modelo operacional em nuvem;

O cloud computing promete um modelo mais flexível para os sistemas que colocam a unidade de negócios ou o cliente no centro do processo. No entanto, isso não vai acontecer repentinamente, pois as pessoas, processos e tecnologia precisam acompanhar esta mudança. O processo para a implantação incluiu alterações na forma de trabalho das pessoas, nos tipos de habilidades necessárias para a organização e a relação dos usuários com os gestores de TI. O sucesso na preparação da organização para lidar com o cloud computing está no fato de que o ING não tratou a mudança para a nuvem como uma questão apenas de TI, mas sim envolveu toda a organização. Para conseguir isso, iniciou-se uma conscientização em larga escala e em torno desse objetivo (ING, 2013). A Figura 18 mostra um gráfico de consciência do ING e do ciclo de vida do programa de educação ao longo do tempo.

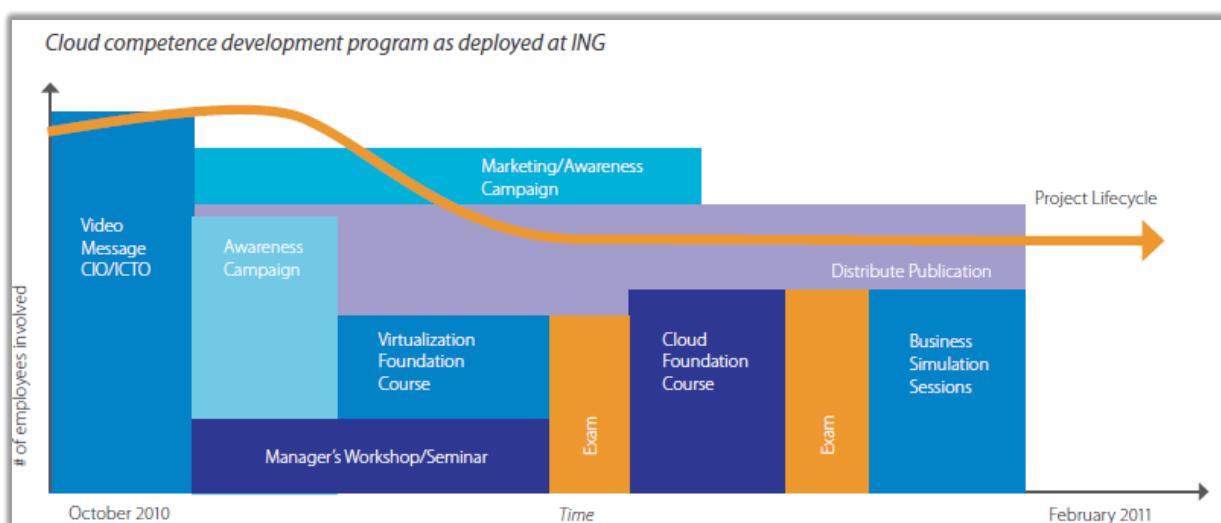


Fig. 18 – Ciclo de vida do programa de conscientização
Fonte: <http://www.itpreneurs.com/document/case-study.pdf>

O programa foi iniciado com um vídeo explicativo, mostrando que a empresa estava passando por uma mudança tecnológica e de processos em longo prazo. Ao mesmo tempo iniciaram uma série de atividades para divulgar esse treinamento aos funcionários. Enquanto a campanha de marketing interno estava em andamento, foi dado inicio à transição de modelo de TI. Os cursos de formação foram disponibilizados *on-line*, permitindo que os funcionários pudessem fazer os cursos em horários convenientes para eles (ING, 2013).

Cada funcionário, cujo cargo foi diretamente afetado por essa transição, participou do programa de conscientização e treinamento. Os gerentes de projeto, gerentes-chave de negócios e gerentes de relacionamento participaram de um programa de conscientização, focando a necessidade do negócio e os principais conceitos. Assim como os profissionais de TI, bem como usuários de negócios com um impacto muito grande em suas funções também participaram do programa (ING, 2013).

O ING treinou cerca de mil e quinhentas pessoas para garantir a capacidade de gerenciar a nova nuvem. Um livro contendo material de leitura adicional e estudos de caso foi feito e entregue para aqueles que completaram o treinamento. Finalmente, uma simulação de negócios virtual, permitiu aos participantes colocarem os conhecimentos obtidos em prática (ING, 2013).

3.10.2 Gol Linhas Aéreas

Com cerca de 63 destinos em toda a América do Sul, a GOL Linhas Aéreas se tornou uma das maiores companhias aéreas do Brasil. Como a conexão à internet ainda não está disponível nos aviões, a companhia percebeu que os passageiros necessitavam de entretenimento de bordo e criou uma solução alinhada com a estratégia da empresa (AMAZON, 2011).

O maior desafio foi determinar como gerenciar o conteúdo que é controlado automaticamente. Desde informações sobre os acessos realizados pelo passageiro e programação que conta com mais de 14 mil opções como vídeos, artigos de revistas, jogos, conteúdo de esportes e notícias (AMAZON, 2011).

Um servidor é instalado no avião, possui um ponto de acesso *wireless*³⁸ e trabalha em conjunto com os servidores centrais implementados na nuvem. Sempre que um avião aterrissa, o servidor dele conecta-se a uma rede *wireless* do aeroporto, que se conecta ao servidor para sincronizar o conteúdo. Esta arquitetura é revolucionária na indústria aérea (AMAZON, 2011).

A empresa conseguiu parceiros de negócios, permitindo que eles façam e gerenciem a publicidade remotamente de acordo com o destino dos passageiros (AMAZON, 2011). A Figura 19 mostra um diagrama de arquitetura a bordo e em terra funciona.

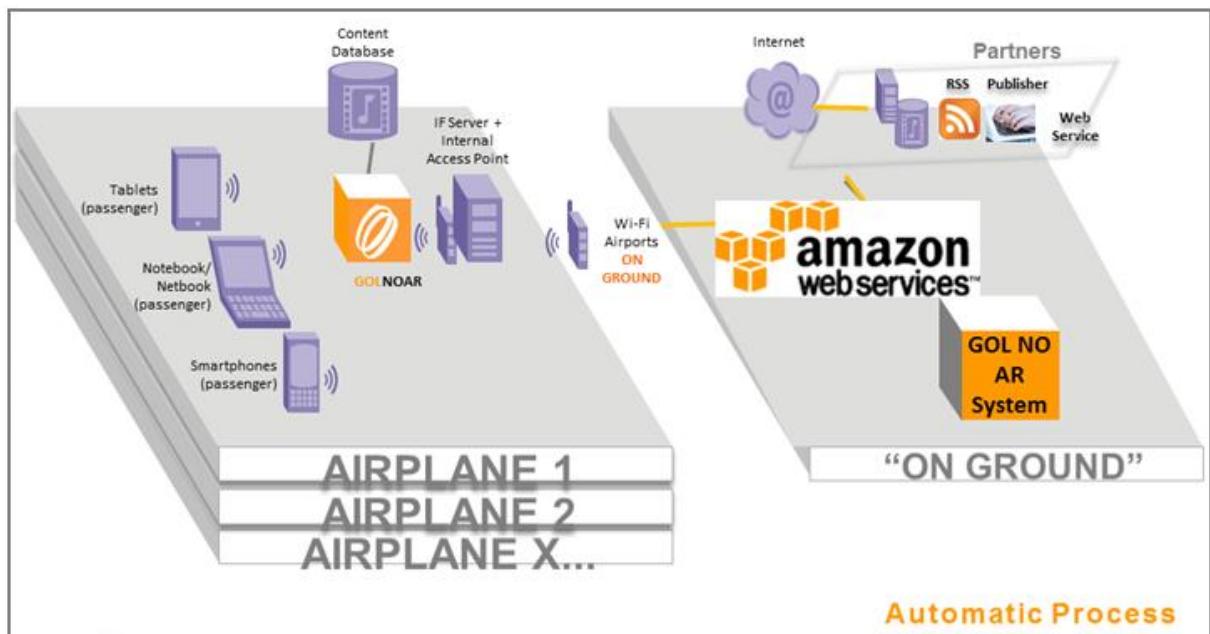


Fig. 19 – Funcionamento do servidor **Fonte:** <http://aws.amazon.com/pt/solutions/case-studies/gol-airlines/>

Quando a GOL decidiu criar esse sistema, a empresa percebeu que precisava de um provedor *web* para que o projeto tivesse sucesso, pois se não houver tempo suficiente para atualizar o servidor a bordo com um arquivo grande, ele pode ser carregado parcialmente no aeroporto atual e o restante na próxima parada (AMAZON, 2011).

³⁸ *Wireless* – rede de computadores sem a necessidade do uso de cabos.

3.10.3 Peixe Urbano

O Peixe Urbano é um portal que auxilia na descoberta dos melhores produtos, serviços e atividades de cada cidade. Para as empresas locais, funciona como uma ferramenta de *marketing* que permite realizar uma grande divulgação além da fidelização de novos clientes. Cada cidade tem o seu próprio *site* do Peixe Urbano, com ofertas selecionadas a sua região, cobrindo mais de 80 cidades do Brasil, Argentina, México e Chile (AMAZON, 2011).

O *site* começou com recursos limitados para investir em infraestrutura, mas já visavam o potencial de rápido crescimento caso o serviço fizesse sucesso. Com isso escolheram o cloud computing pela necessidade de uma solução que tivesse custo baixo e fosse altamente escalável. Atualmente o Peixe Urbano executa uma média de 180 instâncias no EC2, serviço de instâncias do AWS, simultaneamente. A capacidade de escalabilidade garante a estabilidade do site mesmo com usos extremos, em que em alguns casos a variação de pico ultrapassa os 1000%. Utilizando o cloud computing a empresa evitou a despesa investir em *hardware*, se concentrando apenas na procura de ofertas para seus usuários (AMAZON, 2011).

3.10.4 Foursquare

O Foursquare, que é uma rede social baseada na localização dos usuários, conta hoje com de 10 milhões de contas ativas. É através de um aplicativo para *smartphones* que os usuários trocam dicas compartilham lugares por meio de sua localização, registrando em torno de 5 milhões de *check-ins* por dia. Utilizando o Amazon Elastic MapReduce, serviço do AWS, é possível realizar análises como relatórios diários sobre o uso de cada cliente, análise de tendência a longo prazo, avaliação de novos recursos dentre outros para melhor entender e atender seus clientes (AMAZON, 2012).

A facilidade de se fazer toda essa análise em um provedor de nuvem simplifica o processo de implantação e gerenciamento principalmente por causa da flexibilidade, permitindo que os grupos de dados que tenham maior necessidade de serem analisados sejam colocados rapidamente para análise. Os engenheiros podem executar um *cluster* dedicado a testar novas funcionalidades e a equipe de análise de dados pode expandir dinamicamente seu conjunto de relatórios durante os períodos de maior necessidade (AMAZON, 2012).

O fato de a cobrança ser feita sobre os recursos utilizados faz com que os custos sejam reduzidos significativamente usando clusters transitórios para análises. Além disso, a nuvem permite que o Foursquare evite utilizar recursos caros para manter seu grupo de análises utilizando a instâncias disponíveis (AMAZON, 2012).

Recentemente o Foursquare decidiu que iria gastar mais de 1 milhão de dólares com instâncias reservadas por ano, conseguindo com isso reduzir os custos em 35%. Devido baixa nos preços praticados pela AWS a economia foi ainda maior, pois a queda nos preços aliada com o uso de instâncias fará o total de redução dos custos chegar a 53%. Com o uso da nuvem o Foursquare conseguiu não apenas diminuir seus gastos como também o tempo de processamento de dados sem a necessidade de desenvolver aplicativos adicionais para análise dos dados mais urgentes (AMAZON, 2012).

3.11 Desenvolvimento do software

Após ter as informações necessárias sobre o provedor de cloud computing escolhido, se deu início ao desenvolvimento do *software* seguindo alguns passos:

- Definição e configuração do controlador de versão;
- Definição do compilador;
- Configuração do ambiente de desenvolvimento;
- Requisitos;
- Modelagem;
- Criação de manuais de implantação e de configuração;

A utilização do controlador de versão se faz necessário para o controle do desenvolvimento, para realização de *backup* e para facilitar a integração entre os desenvolvedores.

3.12 Implantação

A implantação foi feita pelo serviço Elastic Beanstalk, que permite criar uma nova aplicação e fazer o *deploy* do projeto compactado de maneira bem prática. Existem outras maneiras de se fazer o *deploy*, como por exemplo, utilizar o GIT e o AWS Elastic Beanstalk APIs. Essa por envolver linhas de comando, o *deploy* se torna um pouco mais complexo, portanto optou-se em não utilizar. Outra maneira de fazer o *deploy* seria utilizar o *plugin* de código aberto para o Eclipse Java IDE. Porém esse seria apenas para aplicações *java web* que não foi o caso da aplicação desenvolvida.

Ao iniciar uma nova aplicação foi escolhida a plataforma PHP, pois foi a linguagem utilizada no desenvolvimento da aplicação. No ambiente criado, foi instanciado uma instância Linux 64 bits no EC2 e foi criado um *bucket* no S3 onde a aplicação foi armazenada. A Figura 20 ilustra o ambiente criado.

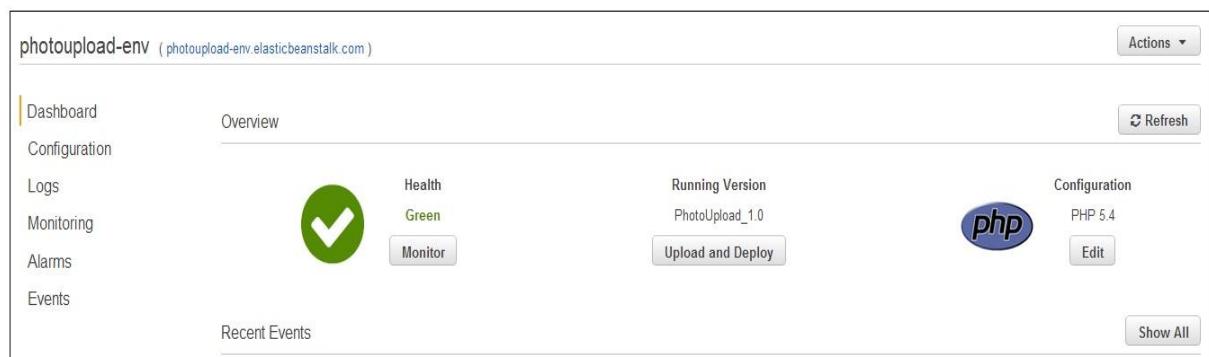


Fig. 20 – Aplicação implantada no Elastic Beanstalk **Fonte:** Elaborado pelo autor.

O Elastic Beanstalk torna a implantação bastante simples, pois possui uma interface amigável e sem complicações. No entanto a *url* oferecida para acesso à aplicação não é adequada para o uso do usuário final. Para esse problema utilizou-se outro serviço da AWS chamado Route 53. Com ele foi possível associar um domínio criado em qualquer outro site a aplicação que foi implantada. A Figura 21 ilustra a arquitetura do Elastic Beanstalk.

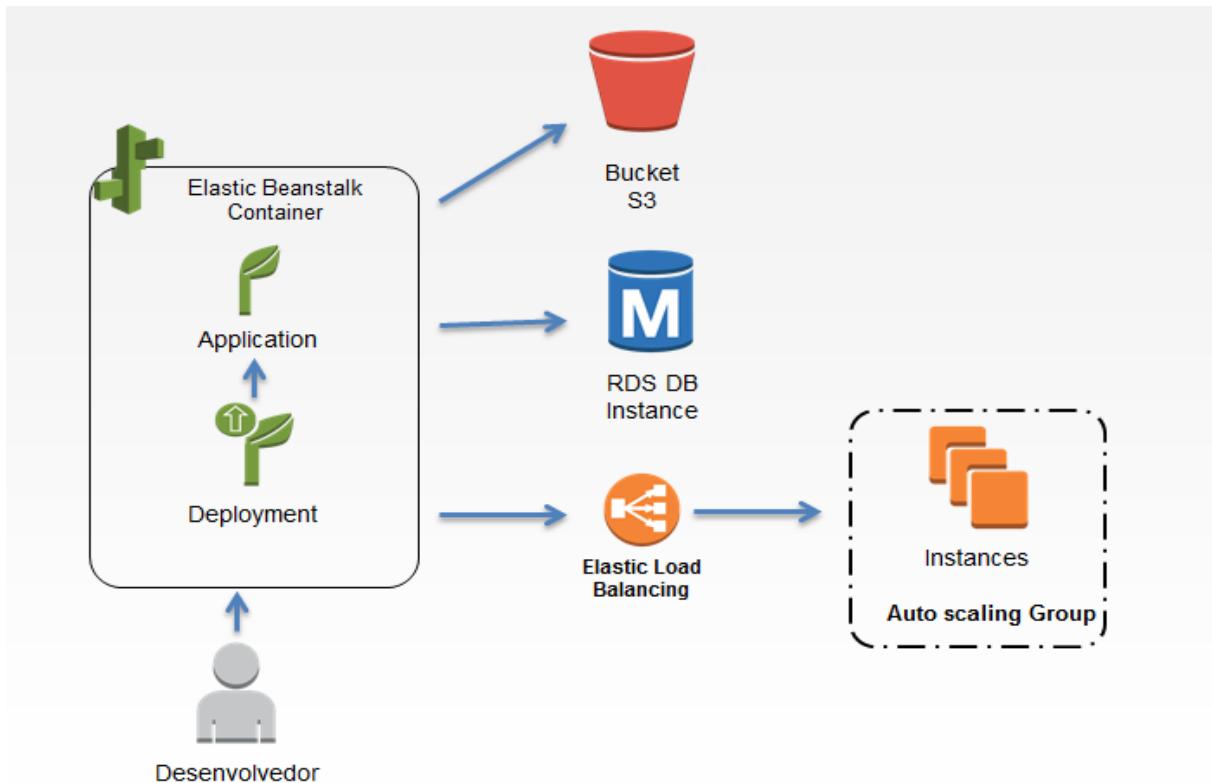


Fig. 21 – Arquitetura Elastic Beanstalk **Fonte:** Elaborado pelo autor.

Pode-se observar na Figura 21 que o Elastic Beanstalk torna-se a implantação bastante simples, pois ele se encarrega de controlar os recursos que a aplicação irá utilizar.

O RDS foi o serviço escolhido para ser usado com o banco de dados da aplicação. A conexão foi realizada pelo link obtido diretamente da instância. A Figura 22 mostra a instância criada no RDS e o link que foi utilizado.

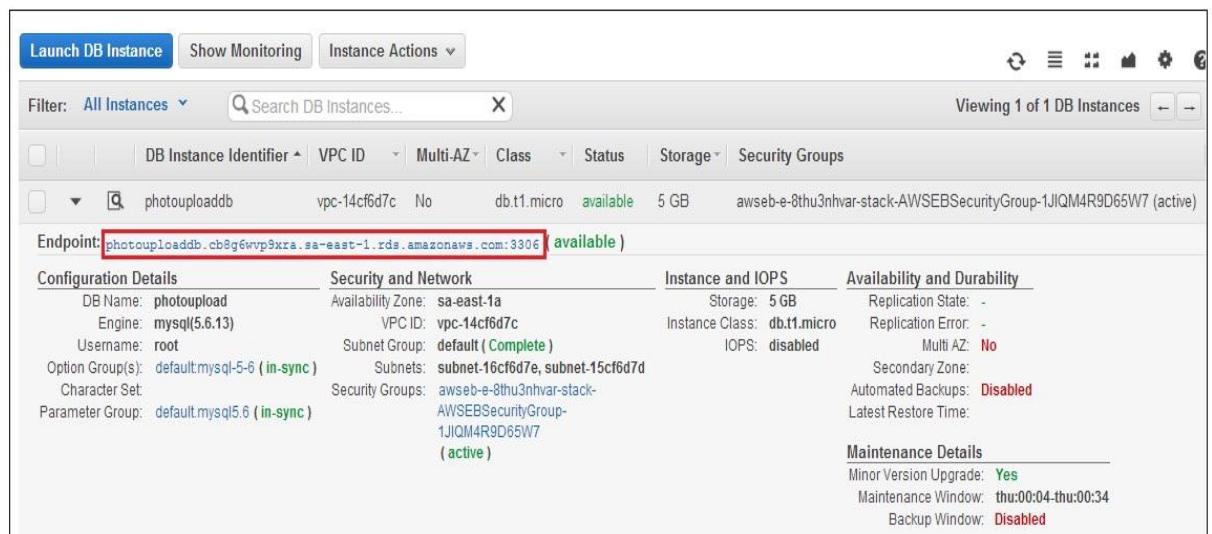


Fig. 22 – Instância criada no RDS **Fonte:** Elaborado pelo autor.

Para conexão ao banco foi necessário informar o usuário e senha que foi definida inicialmente na criação da instância e a manipulação do banco e criação das tabelas foram feitos através do NetBeans, por meio de uma conexão remota. A figura 22 mostra essa configuração da IDE para o acesso.

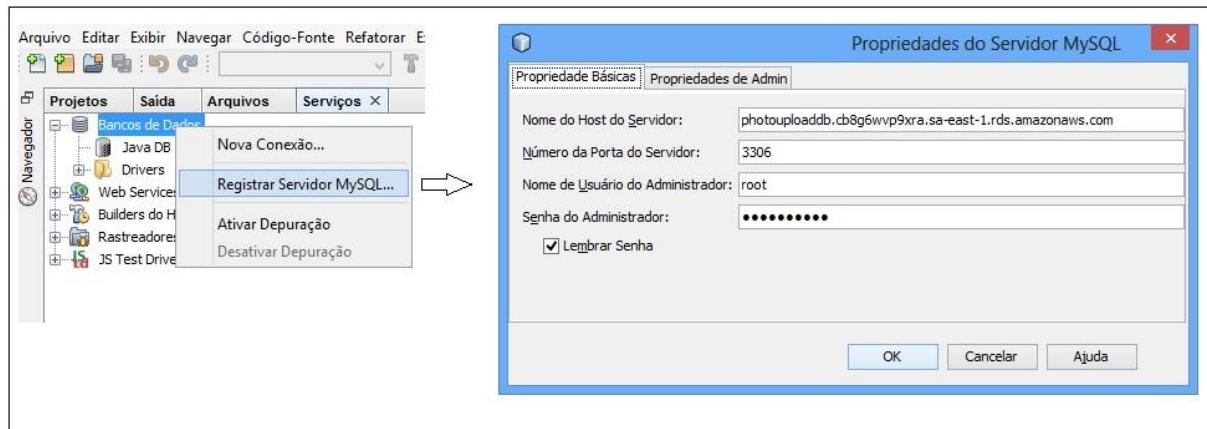


Fig. 23– Configuração da IDE para acesso ao RDS. Fonte: Elaborado pelo autor.

Após realizar as configurações acima foi possível ter acesso e criar as tabelas necessárias para uso da aplicação. As funcionalidades oferecidas pelo RDS permitem criar as tabelas de duas formas. A primeira por meio da *interface* do Netbeans e a segunda por meio de *scripts SQL*.

Para um melhor entendimento sobre a estrutura do Elastic Beanstalk, foi elaborado um diagrama para demonstração. A Figura

3.13 Migração entre regiões

A migração entre regiões é importante quando se deseja utilizar alguma outra região que esteja mais próxima ou que ofereça uma latência menor.

A migração do Elastic Block Store (EBS) pode ser feita por um recurso chamado EBS Snapshot Copy que permite copiar os *snapshots* de discos EBS entre regiões diferentes. Esse recurso facilita o uso de múltiplas regiões AWS para realizar uma expansão geográfica, migração de *data centers* e *disaster recovery* (AMAZON, 2013). A Figura 24 ilustra uma cópia de EBS.

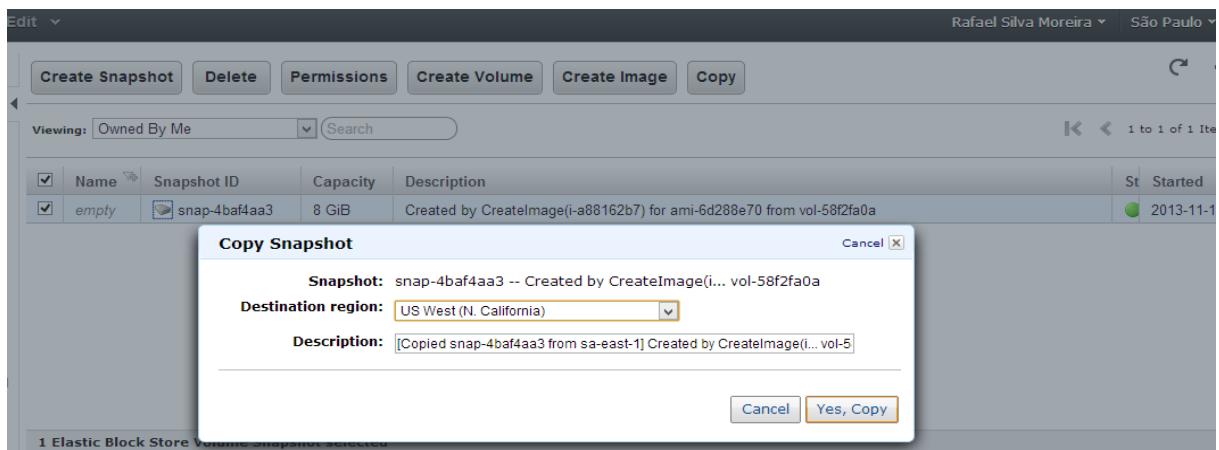


Fig. 24– Efetuando cópia de um *snapshot* para outra região. **Fonte:** Elaborado pelo autor.

Pode-se observar que um *snapshot* de uma instância de São Paulo pode ser copiado para qualquer outra região.

3.14 Resultados

Após muito estudo e pesquisa foi possível enfim compreender mais a fundo o que é cloud computing e como realizar a escolha do provedor de serviços. A partir desse conhecimento foi desenvolvida uma aplicação para a demonstração prática dos recursos oferecidos por esse modelo computacional. A Figura 25 ilustra a aplicação criada.



Fig. 25– Aplicação desenvolvida. **Fonte:** <http://awstcc.tk/home/home.php>.

Foi desenvolvido um sistema de armazenamento e manipulação de fotos, onde é possível cortá-las, criar álbuns personalizados e exibi-las *online*. O principal objetivo da criação desse aplicativo é utilizar os principais recursos da computação em nuvem como escalabilidade de recursos, máquinas virtuais, replicação de banco de dados dentre outros. Para isso, alguns serviços foram utilizados como o Elastic Compute Cloud (EC2), Simple Storage Service (S3), Elastik beanstalk, Route 53 e o Relational Database Service (RDS). A Figura 26 ilustra a arquitetura desses recursos sendo utilizada pela aplicação.

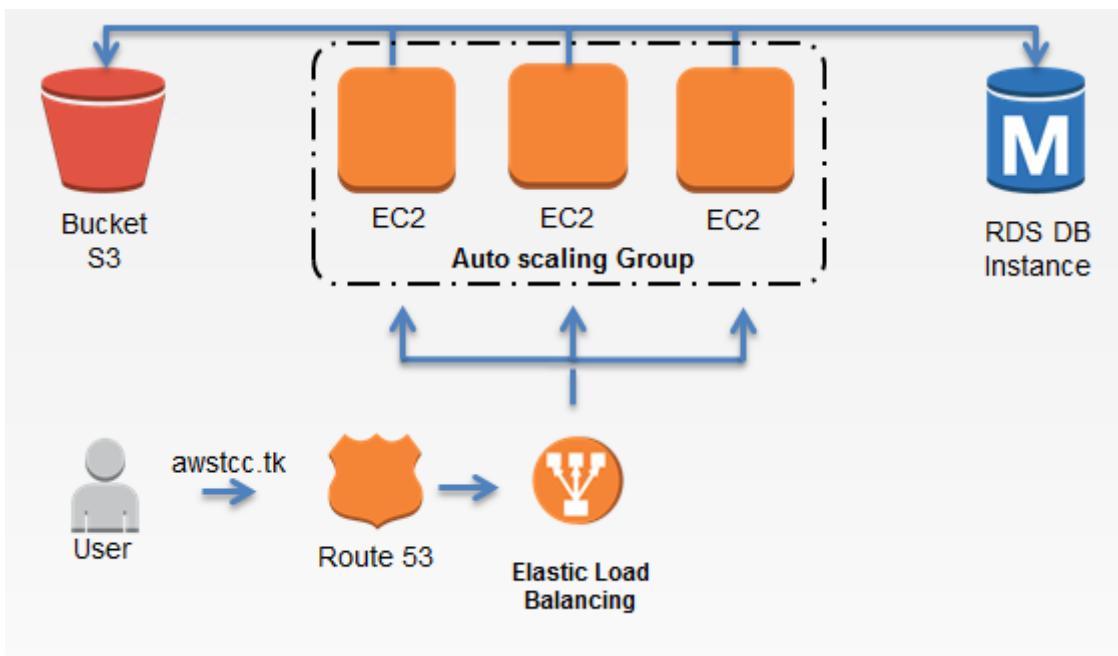


Fig. 26– Arquitetura de usuário utilizando os recursos AWS **Fonte:** Elaborado pelo autor

Pode-se observar que o usuário entra com o endereço da aplicação, o Route 53 faz o direcionamento do endereço externo para a aplicação, o Elastic Load Balancing faz a distribuição de carga entre as instâncias EC2 e o Auto Scaling finca encarregado de criar e excluir instâncias conforme a demanda da aplicação. Outros recursos que utilizados também foi o S3 e o banco de dados RDS.

. As linguagens utilizadas para o desenvolvimento da aplicação foram o PHP, para processar as requisições do aplicativo, o JavaScript para validação dos formulários, o JQuery e seu *plugin* JCrop para localização dinâmica dos álbuns e recorte de fotos.

Com o uso do SDK³⁹ para PHP fornecido pela AWS foi possível que a aplicação realizasse o acesso ao S3. Esse SDK é um conjunto de classes PHP que elimina a complexidade de código em se conectar aos serviços da AWS.

A figura 24 mostra como acessar esse serviço para a criação de um *bucket* no S3.

³⁹ SDK- Sigla de “Software Development Kit”.

```

require '../vendor/aws-autoloader.php';
use Aws\S3\S3Client;

//CRIAÇÃO DO BUCKET USUÁRIO
public function createBucketUser($bucketUser) {
    //ARQUIVO DE CONFIGURAÇÃO
    require '../home/s3_config.php';
    //RECEBE AS CREDENCIAIS DE ACESSO AO S3
    $clientS3 = S3Client::factory(array(
        'key' => $key,
        'secret' => $secretkey
    ));

    //PERMITE TRATAR AMAZON S3 COMO UM SISTEMA DE ARQUIVOS
    $clientS3->registerStreamWrapper();
    //FUNÇÃO PARA CRIAÇÃO DO BUCKET
    return mkdir("s3://$bucketUser");
}

```

Fig. 27 – Código PHP pra criação do *bucket* no S3 **Fonte:** Elaborado pelo autor.

Ao criar uma instância no S3 foram geradas as credenciais de acesso. Para manipular os diretórios S3 é necessário informar essas chaves em todas as requisições. A Figura 28 mostra os *buckets* criados a partir da aplicação pelos códigos demostrados na Figura 27.

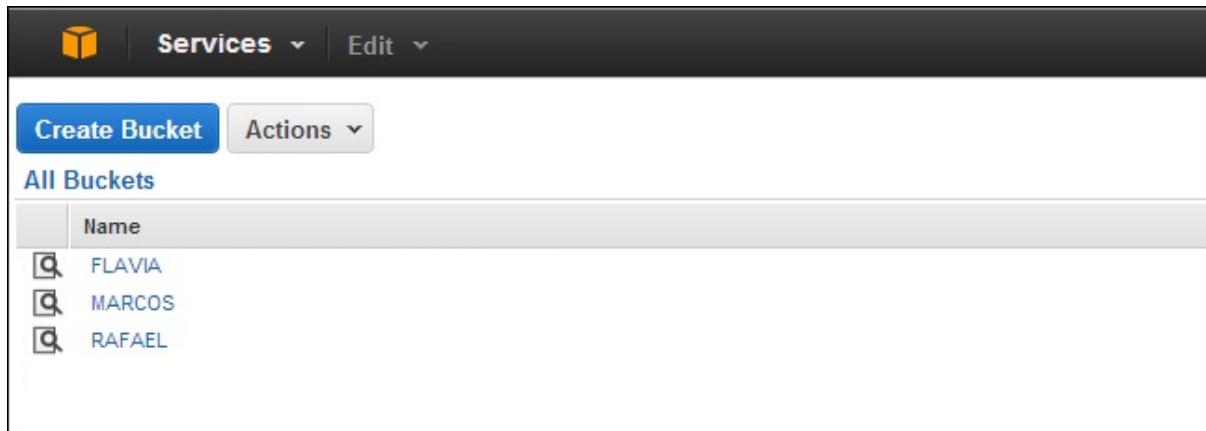


Fig. 28– *Buckets* de usuários no S3. **Fonte:** Elaborado pelo autor.

Com isso é possível que cada usuário tenha seus arquivos protegidos e separados dos demais, pois cada usuário registrado na aplicação tem seu próprio *bucket*. Dentro deles é possível a criação de álbuns se assim solicitados pelos usuários

Para a criação do álbum, foi utilizado o método ‘*putObject*’ do SDK. Ele faz a verificação da existência desse álbum; caso ele exista, os arquivos são gravados nele, caso contrário, o álbum é criado. A Figura 29 ilustra a utilização método.

```

require '../vendor/aws-autoloader.php';
use Aws\S3\S3Client;

//CRIA ALBUM E ARQUIVOS
public function createS3Album($tmp, $bucket, $albumName, $fotoName) {
    //ARQUIVO DE CONFIGURAÇÃO
    require '../home/s3_config.php';
    $directoryBucket = $bucket . '/' . $albumName;
    //RECEBE AS CREDENCIAIS DE ACESSO AO S3
    $clientS3 = S3Client::factory(array(
        'key' => $key,
        'secret' => $secretkey
    ));

    //SE ALBUM NÃO EXISTIR CRIA E SALVA OS ARQUIVOS
    $response= $clientS3->putObject(array(
        'Bucket' => $bucket.'/'.$albumName,
        'Key' => $fotoName,
        'SourceFile' => $tmp
    ));

    return $response;
}

```

Fig. 29 – Código PHP para criação de álbuns e arquivos. **Fonte:** Elaborado pelo autor.

Esse método retorna uma resposta, pois caso o arquivo não seja gravado é retornada uma resposta de falha e a quantidade de fotos que não foram gravadas, do contrário uma resposta positiva é devolvida. A Figura 30 ilustra o resultado conseguido com esse método.

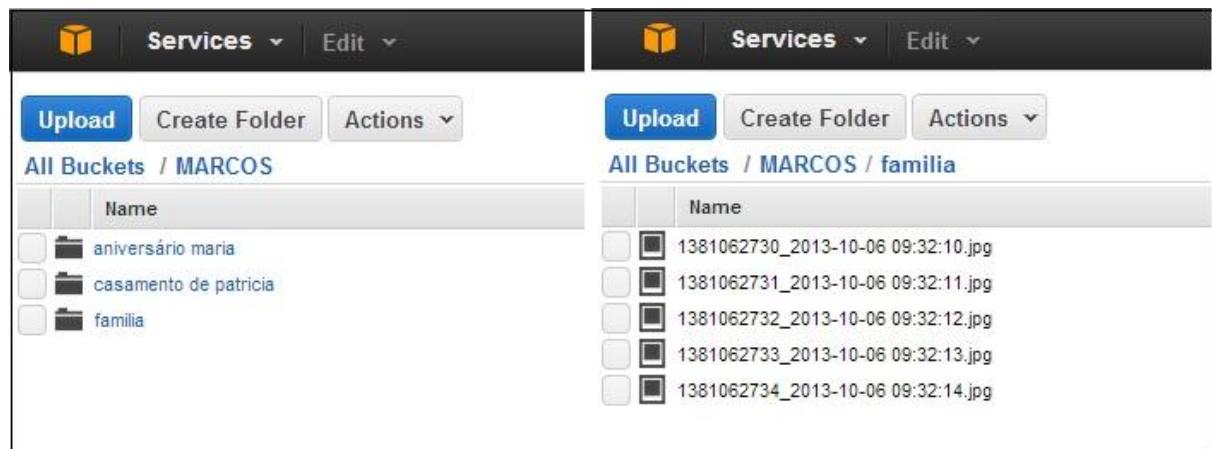


Fig. 30 – Álbuns e fotos criados pela aplicação **Fonte:** Elaborado pelo autor.

Para realizar a exclusão dos arquivos foi utilizada a função ‘*unlink*’. Essa função recebe o diretório e nome do arquivo a ser excluído e retorna uma reposta de sucesso ou falha. O código é ilustrado a na Figura 31.

```
require '../vendor/aws-autoloader.php';
use Aws\S3\S3Client;

//DELETA FOTO DO ALBUM
public function deleteS3AlbumPhoto($bucket, $albumName, $fotoName) {
    //ARQUIVO DE CONFIGURAÇÃO
    require '../home/s3_config.php';

    $directoryBucket = $bucket . '/' . $albumName;
    //RECEBE AS CREDENCIAIS DE ACESSO AO S3
    $clientS3 = S3Client::factory(array(
        'key' => $key,
        'secret' => $secretkey
    ));
    //PERMITE TRATAR AMAZON S3 COMO UM SISTEMA DE ARQUIVOS
    $clientS3->registerStreamWrapper();
    //DELETA O ARQUIVO
    return unlink("s3://$bucket/$albumName/$fotoName");
}
```

Fig. 31 – Código PHP para exclusão de álbuns e arquivos. **Fonte:** Elaborado pelo autor.

O uso do SDK diminui a complexidade da manipulação dos diretórios no S3, economizando tempo e oferecendo segurança aos arquivos, pois para ter acesso é preciso informar a *key pair*.

Outro recurso utilizado na aplicação foi o RDS, que é a instância de banco de dados da aplicação onde ficarão armazenados os cadastros dos usuários. Ao criar uma instância, foi definido um usuário, uma senha e gerado um *link* para acesso a ela.

A conexão foi feita através do PDO⁴⁰, por ele foram passados os parâmetros como usuário, senha, nome da instância, *link* de acesso e a porta de conexão.

O Elastic Beanstalk foi utilizado para implantação da aplicação na nuvem. Ele oferece o ambiente de desenvolvimento e faz a escalabilidade dos recursos conforme a necessidade. A Figura 32 mostra como foi feita a configuração do escalonamento automático da aplicação implantada.

⁴⁰ PDO – O PHP *Data Objects* é um modo de acesso à base de dados em PHP.

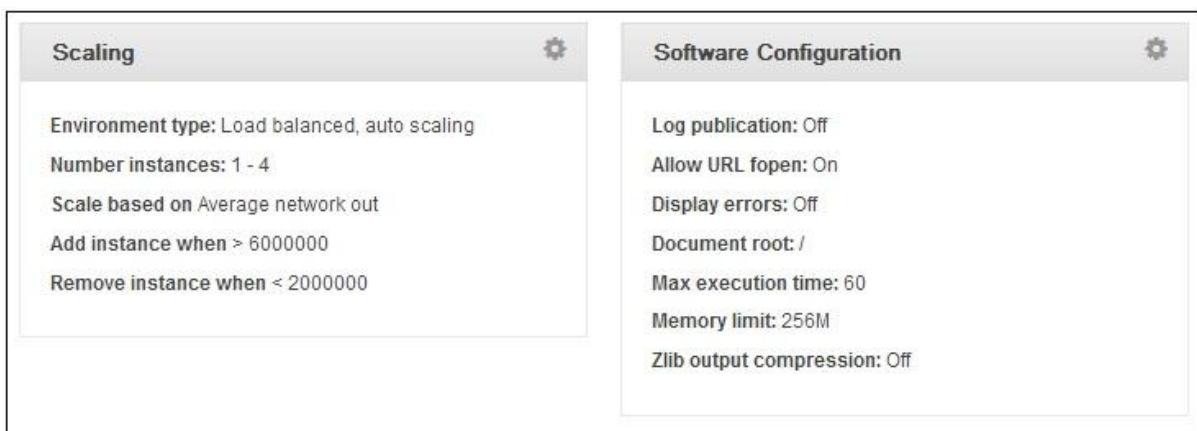


Fig. 32 – Painel de configuração do Elastic Beanstalk **Fonte:** Elaborado pelo autor.

Nesse painel de configuração do ambiente de desenvolvimento é possível configurar a escalabilidade de acordo com os limites que forem colocados, podendo ser medidos pelo uso de processamento, quantidade de tráfego na rede ou pela latência.

4 DISCUSSÃO DOS RESULTADOS

Com o estudo dos conceitos e serviços da computação em nuvem, foi possível criar uma aplicação que utilizasse ambos. Configurando o Elastic Beanstalk para a criação de instâncias adicionais no EC2 conseguimos garantir que o servidor ficasse sobrecarregado e houvesse o risco de a aplicação ficar inoperante.

Pelo fato de a aplicação desenvolvida não ser pública e por isso não contar com um grande número de acessos, a escalabilidade foi configurada com um valor máximo de 25% e mínimo de 10% de utilização da CPU. Lembrando que a instância escolhida é a t1.micro e possui uma baixa capacidade de processamento, sendo que a quantidade de processamento pode ser facilmente atingida.

No começo foi simulado um acesso à aplicação com a configuração feita para não criar instâncias adicionais. O resultado foi que o processamento ultrapassou os 25% estipulados e continuou subindo, como ocorre atualmente no modelo convencional de computação. Na Figura 33 é ilustrada a instância.

Viewing: All Instances All Instance Types Search										
	Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status	Monitoring	Security Group
<input type="checkbox"/>	UploadPhoto	i-e13120fd	ami-58538945	ebs	t1.micro	● running	✓ 2/2 checks passed	none	basic	awseb-e-8thu3nh
No EC2 Instances selected.										
Select an instance above										

Fig. 33 – Instância criada no EC2. **Fonte:** Elaborado pelo autor.

Observa-se que existe apenas uma instância EC2 criada e executando na nuvem, por isso a aplicação corre o risco de ficar indisponível caso aconteça uma falha nela e não exista outra para assumir o processamento. As Figuras 34 e 35 ilustram as estatísticas de utilização desta instância.

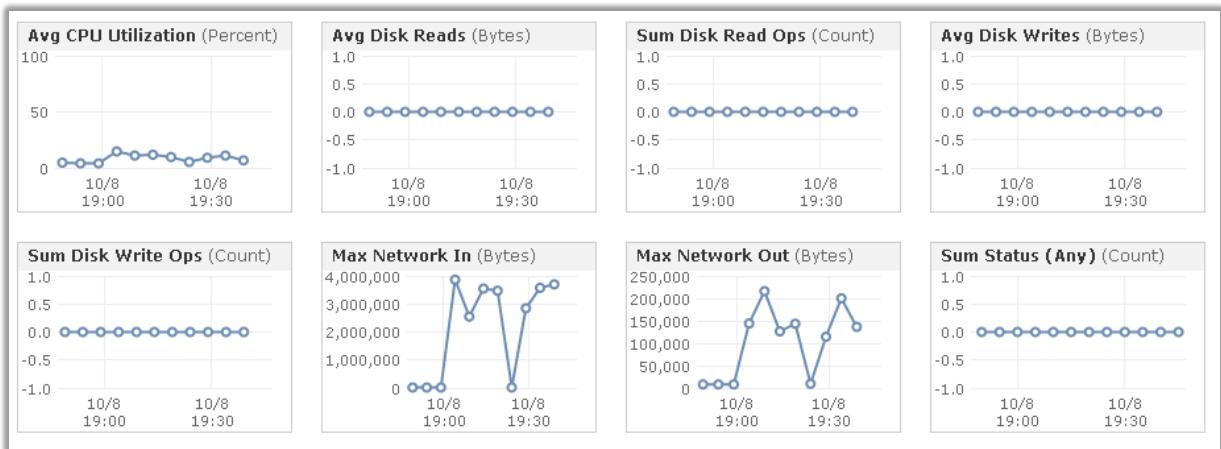


Fig. 34 – Painel de estatística da instância **Fonte:** Elaborado pelo autor.

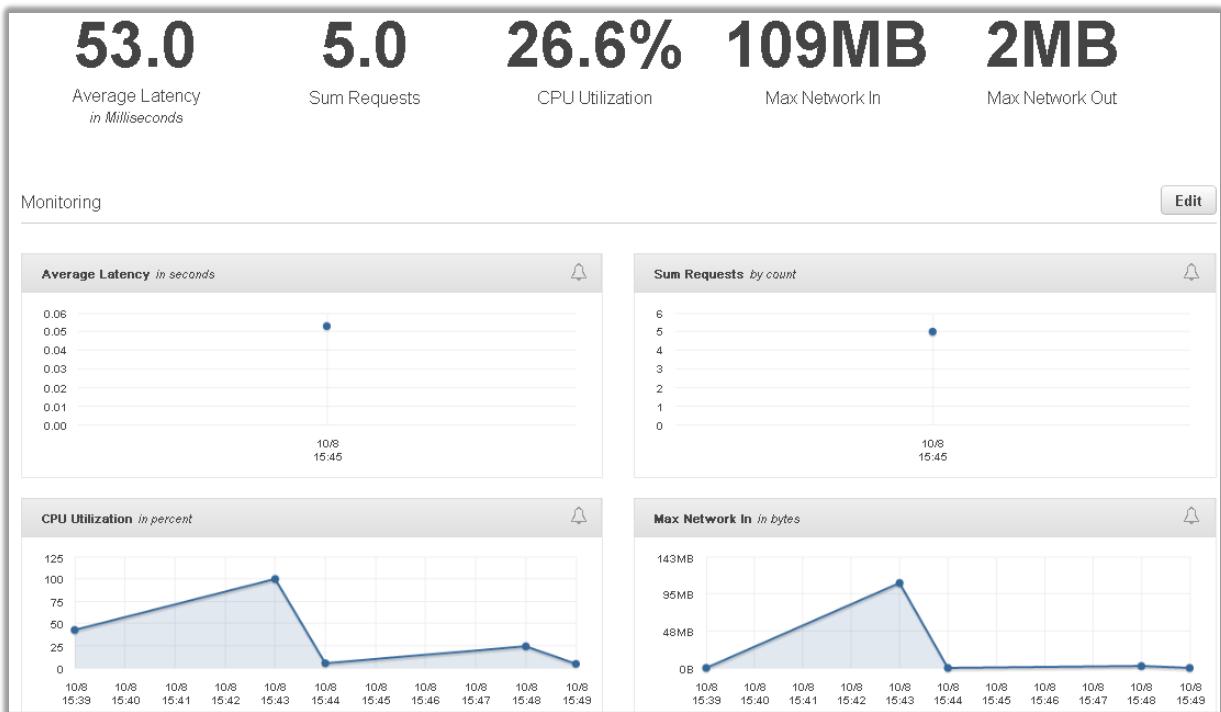


Fig. 35– Estatísticas de uso da aplicação com uma instância. **Fonte:** Elaborado pelo autor.

Em seguida a configuração foi alterada para que, quando o limite de processamento da CPU tivesse sido ultrapassado, automaticamente fosse criada uma segunda instância, conforme mostra a Figura 36.

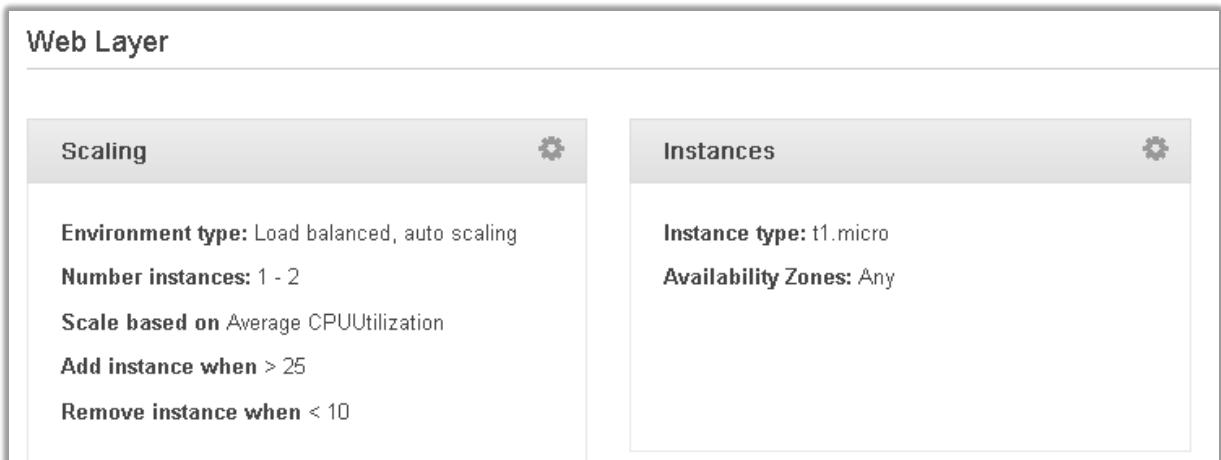


Fig. 36 – Configurações de escalabilidade. **Fonte:** Elaborado pelo autor.

Em seguida foi realizado um *upload* de vários arquivos de fotos que geraram um uso da CPU de 31% conforme a figura 37.

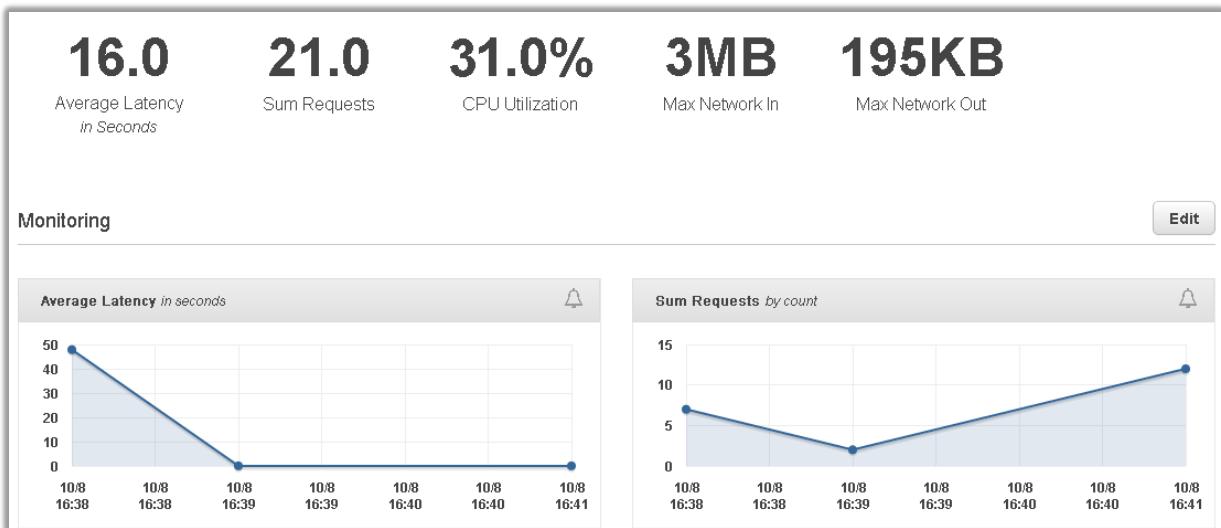


Fig. 37 – Estatísticas de uso da aplicação com uma instância **Fonte:** Elaborado pelo autor.

Imediatamente uma nova instância começou a ser inicializada no EC2, para que esse valor se mantivesse dentro do estipulado na configuração anterior.

O recurso responsável para que esse escalonamento acontecesse foi o *Auto Scaling*, que permite que os servidores do EC2 aumentem ou diminuam os recursos automaticamente de acordo com definições estabelecidas previamente para a aplicação. A Figura 38 ilustra a criação automática desta segunda instância.

	Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status	Monitoring	Security Group
<input type="checkbox"/>	UploadPhoto	i-e13120fd	ami-58538945	ebs	t1.micro	running	2/2 checks passed	none	basic	awseb-e-8thu3
<input type="checkbox"/>	UploadPhoto	i-8bcc590	ami-58538945	ebs	t1.micro	running	initializing...	none	basic	awseb-e-8thu3

Fig. 38 – Criação da segunda instância. **Fonte:** Elaborado pelo autor.

Quando esse valor fica, por um certo período, dentro dos valores escolhidos de processamento, automaticamente a instância é eliminada. A Figura 39 mostra as estatísticas do uso das duas instâncias.

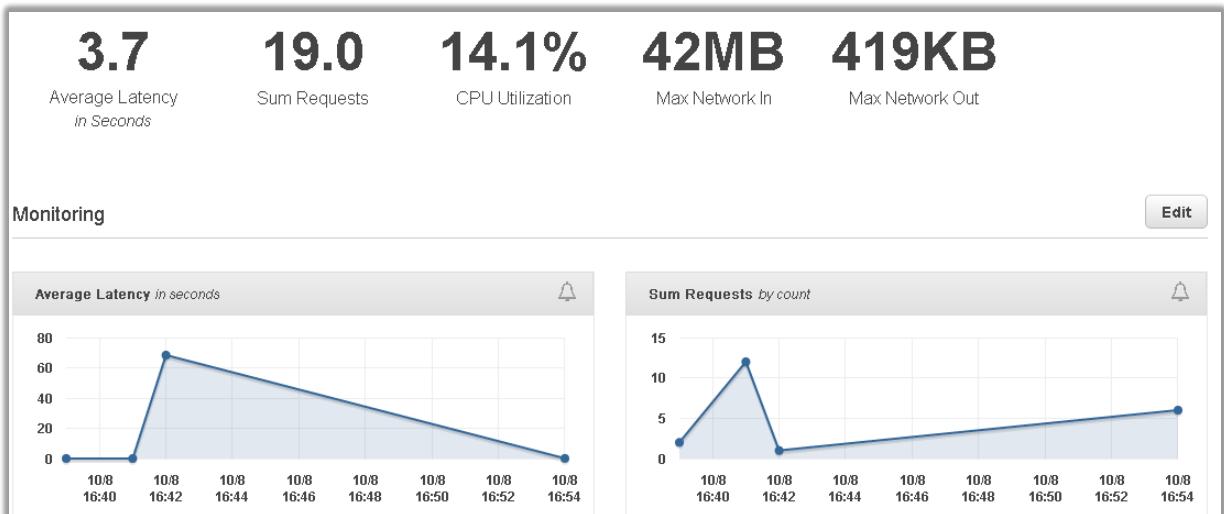


Fig. 39 – Estatísticas de uso da aplicação com duas instâncias. **Fonte:** Próprio autor.

A Figura 40 mostra a segunda instância terminada automaticamente após o uso do processador ser normalizado.

	Name	Instance	AMI ID	Root Device	Type	State	Status Checks	Alarm Status	Monitoring	Security Group
<input type="checkbox"/>	UploadPhoto	i-e13120fd	ami-58538945	ebs	t1.micro	running	2/2 checks passed	none	basic	awseb-e-8thu3
<input type="checkbox"/>	UploadPhoto	i-8bcc590	ami-58538945	ebs	t1.micro	terminated		none	basic	

Fig. 40 – Segunda instância sendo terminada. **Fonte:** Elaborado pelo autor.

Com a utilização do modelo computacional em nuvem, conseguimos uma aplicação escalável, que possibilita a alocação de recurso rapidamente. Se esse procedimento tivesse sido feito com uma aplicação de grande porte, com certeza ocorreriam problemas nos testes iniciais, pois no modelo convencional seria necessário investir em infraestrutura sem saber se esse pico na utilização da CPU ocorreria novamente, mas com o cloud computing em poucos minutos um novo servidor estava à disposição.

5 CONSIDERAÇÕES FINAIS

Neste trabalho foi explorado e apresentado o conceito de cloud computing, os modelos de serviços, as questões de segurança e gerenciamento, as principais empresas que oferecem esse serviço e quais os recursos que cada uma delas oferece. Foi feita uma pesquisa mais aprofundada sobre o provedor de serviços da Amazon e demonstrado, de forma prática, como realizar o *deploy* de uma aplicação PHP e os principais recursos que a aplicação utilizou na nuvem.

Pode-se observar que o fator segurança é o ponto chave que as empresas precisam analisar cuidadosamente antes de fechar qualquer contrato e que precisam confiar na política de segurança da nuvem escolhida. Talvez pelo fato de o cloud computing ser um novo modelo, as empresas não confiam totalmente em deixar dados importantes armazenados em *web services* espalhados pelo mundo, por isso ainda utilizam as nuvens privadas e híbridas.

Com a pesquisa realizada sobre os provedores Windows Azure, Google App Engine e AWS, pode-se constatar que o Azure e a AWS são infraestruturas como serviço (*Infrastructure as a Service*) e o Google App Engine é uma plataforma como serviço

Com os resultados obtidos, descobrimos que o cloud computing já está revolucionando o modelo tradicional de TI. O escalonamento de recursos, a facilidade na implantação de novas máquinas virtuais e principalmente por não haver a necessidade de configurações avançadas dessas VM, é possível ter uma grande agilidade quando se trata em atender as demandas, possibilitando isso de forma quase instantânea e sem desperdício de recursos. Esses fatores pesam e muito a favor do cloud computing, além da redução de custos com manutenção e aquisição de equipamentos físicos.

Também foi possível entender que o futuro do uso desse modelo está diretamente ligado à questão de segurança. Por ser uma área delicada, ainda necessita de muito estudo e avanço.

O objetivo geral e os específicos foram alcançados, proporcionando um grande conhecimento sobre o cloud computing. Nesta pesquisa o foco esteve no entendimento e demonstração deste modelo computacional criando um documento que poderá servir como referencial para futuros estudos, pesquisas e projetos realizados nesta área, como por exemplo, um estudo avançado sobre a segurança na utilização deste modelo, pois é um assunto de extrema importância para o futuro do cloud computing.

REFERÊNCIAS

ALVAREZ, M.A. Introdução ao HTML. Disponível em:
<http://www.criarweb.com/artigos/10.php>. Acesso em: 25 de maio 2013.

AMAZON; Amazon Auto Scaling. 2013. Disponível em:
<http://aws.amazon.com/pt/autoscaling>. Acesso em 19 de julho de 2013.

AMAZON; Facilite migração entre regiões, expansão geográfica e disaster recovery com o EBS Snapshot Copy. Disponível em: <<http://aws.typepad.com/brasil/2012/12/facilite-migracao-entre-regioes-expansao-geografica-e-disaster-recovery-com-o-ebs-snapshot-copy.html>> Acesso em 14 de novembro de 2013.

AMAZON; Amazon Cloud Front. 2013. Disponível em:
<http://aws.amazon.com/pt/cloudfront>. Acesso em 19 de julho de 2013.

AMAZON; Amazon Cloud Watch. 2013. Disponível em:
<http://aws.amazon.com/pt/cloudwatch>. Acesso em: 21 de julho de 2013.

AMAZON; Amazon Elastic Beanstalk. Disponível em: <<http://aws.amazon.com/pt/ebs>>
acesso em 19 de julho de 2013.

AMAZON; Amazon Elastic Block Store. 2013. Disponível em:
<http://aws.amazon.com/pt/ebs>. Acesso em: 07 de março de 2013.

AMAZON; Amazon Elastic Compute Cloud. 2013. Disponível em:
<http://aws.amazon.com/pt/ec2>. Acesso em: 07 de março de 2013.

AMAZON; Amazon Elastic MapReduce. 2013. Disponível em:
<http://aws.amazon.com/pt/elasticmapreduce>. Acesso em: 19 de julho de 2013.

AMAZON; Amazon Elastic Transcoder. 2013. Disponível em: <<http://aws.amazon.com/pt/elastictranscoder>>
Acesso em: 02 de outubro de 2013.

AMAZON; Amazon Fulfilment. 2013. Disponível em:
<http://services.amazon.co.uk/services/fulfilment-by-amazon/how-it-works.html>. Acesso em 19 de julho de 2013.

AMAZON; Amazon Premium Support. 2013. Disponível em:
<http://aws.amazon.com/pt/premiumsupport>. Acesso em: 21 de junho de 2013.

AMAZON; Amazon RedShift. 2013. Disponível em: <<http://aws.amazon.com/pt/redshift>>. Acesso em: 02 de outubro de 2013.

AMAZON; Amazon Relational Database Service. 2013. Disponível em:
<http://aws.amazon.com/pt/rds>. Acesso em: 07 de março de 2013.

AMAZON; **Amazon Route 53**. 2013. Disponível em: <<http://aws.amazon.com/pt/route53/>> Acesso em 21 de julho de 2013.

AMAZON; **Amazon Simple DB**. 2013. Disponível em: <<http://aws.amazon.com/pt/simpledb/>> Acesso em 19 de julho de 2013.

AMAZON; **Amazon Simple Email Service**. 2013. Disponível em: <<http://aws.amazon.com/pt/ses/>> Acesso em 20 de julho de 2013.

AMAZON; **Amazon Simple Notification Service**. 2013. Disponível em: <<http://aws.amazon.com/pt/sns/>> Acesso em: 20 de junho de 2013.

AMAZON; **Amazon Simple Queue Service**. 2013. Disponível em: <<http://aws.amazon.com/pt/sqs/>> Acesso em 20 de junho de 2013.

AMAZON; **Amazon Simple Storage Service**. 2013. Disponível em: <<http://aws.amazon.com/pt/s3/>>. Acesso em: 07 de março de 2013.

AMAZON; **Amazon Virtual Private Cloud**. 2013. Disponível em: <<http://aws.amazon.com/pt/vpc/>>. Acesso em: 07 de março de 2013.

AMAZON; **AWS Amazon**. 2013. Disponível em: <<http://aws.amazon.com/pt/what-is-aws/>>. Acesso em: 07 de março de 2013.

AMAZON; **Estudo de caso: Foursquare**. 2012. Disponível em: <<http://aws.amazon.com/pt/solutions/case-studies/foursquare/>> Acesso em: 11 de setembro de 2013.

AMAZON; **Estudo de caso: Gol Linhas Aéreas**. 2011. Disponível em: <<http://aws.amazon.com/pt/solutions/case-studies/gol-airlines/>> Acesso em: 11 de setembro de 2013.

AMAZON; **Estudo de caso: Peixe Urbano**. 2011. Disponível em: <<http://aws.amazon.com/pt/solutions/case-studies/peixe-urbano/>> Acesso em: 11 de setembro de 2013.

ANDRADE, M. M.; **Introdução à Metodologia do Trabalho Científico**. 8º Ed. São Paulo. Atlas. 2007.

ARTHUR; Luiz; **Segurança da Informação**. 2009. Disponível em: <http://www.slideshare.net/lui_z_arthur/securanca-da-informao-conceitos>. Acesso em: 14 de novembro de 2013.

ASHLEY, Brent. **Computação em nuvem para empresas: Parte 1. Capturando a Nuvem**, 2009. Disponível em: <http://www.ibm.com/developerworks/br/java/library/0904_amrhein/>. Acesso em: 25 de maio de 2013.

BADGER, L.; **DRAFT Cloud Computing Synopsis and Recommendations**. 2011. Disponível em: <<http://csrc.nist.gov/publications/drafts/800-146/Draft-NIST-SP800-146.pdf>>. Acesso em: 28 de maio de 2013.

BARBOLO, Rafael. **Conheça os serviços em nuvem oferecidos pela Amazon.** 2011. Disponível em: <<http://www.bitabit.eng.br/2011/02/15/conheca-os-servicos-em-nuvem-oferecidos-pela-amazon-web-services/>> acesso em 19 de julho de 2013.

CAMBIUCCI, Waldemir.; **Introdução ao Windows Azure.** 2011. Disponível em: <<http://msdn.microsoft.com/pt-br/library/hh150078.aspx>> Acesso em: 21 de julho de 2013.

CLOUD SECURITY ALLIANCE. 2012. **Da Prática Para o Exame.** Disponível em: <<https://chapters.cloudsecurityalliance.org/brazil/files/2012/04/cap17.pdf>>. Acesso em: 03 de junho de 2013.

CLOUD COMPUTING USE CASES GROUP; **Use Cases.** 2010. Disponível em: <http://cloudusecases.org/Cloud_Computing_Use_Cases_Whitepaper-4_0.odt>. Acesso em: 03 de junho de 2013.

CONDÉ, Luciano.; **Virtual Network agora tem suporte para Windows RRAS e Conexão Ponto para Rede Virtual.** 2013. Disponível em: <<http://blogs.msdn.com/b/conde/archive/2013/05/06/an-250-ncio-virtual-networks-agora-suporta-windows-rras-e-conex-227-o-ponto-para-rede-virtual.aspx>> Acesso em: 27 de agosto de 2013.

EDIN, C. S. B.; **Computação em nuvem e as diferenças tecnológicas entre amazon e o azure.** 2011. Disponível em: <http://repositorio.roca.utfpr.edu.br:8080/jspui/bitstream/1/648/1/CT_TELEINFO_XIX_2011_04.pdf>. Acesso em: 06 de março de 2013.

FROST & SULLIVAN; **Comparing CDN Performance: Amazon Cloud Front's Last Mile Testing Results.** 2013. Disponível em: <http://media.amazonwebservices.com/FS_WP_AWS_CDN_CloudFront.pdf> Acesso em 19 de julho de 2013.

GARCIA, Felipe.; **OpsWorks x Elastic Beanstalk.** 2013. Disponível em: <<http://awshub.com.br/forum/index.php?/topic/3823-opsworks-x-elastic-beanstalk/>> Acesso em: 02 de outubro de 2013.

GIL, A.C.; **Métodos e Técnicas de Pesquisa Social.** 5º. Ed. São Paulo. Atlas. 2007.

GODINHO, Rafael.; **Windows Azure Compute: Roles.** 2013. Disponível em: <<http://msdn.microsoft.com/pt-br/windowsazure/hh500326>> Acesso em: 24 de julho de 2013.

GOLDEN, Bernard.; **SLA: Todo cuidado é pouco em computação na nuvem.** 2011. Disponível em: <<http://computerworld.uol.com.br/tecnologia/2011/11/14/slais-todo-o-cuidado-e-pouco-em-computacao-em-nuvem-sera/>> Acesso em: 19 de julho de 2013.

GOODMAN, D.; **JavaScript - Guia definitivo.** 4º Ed. O'Reilly & Associates, Inc. 2002.

GOOGLE. **Whats is Google AppEngine.** 2013. Disponível em: <<https://developers.google.com/appengine/docs/whatisgoogleappengine>> Acesso em: 27 de agosto de 2013.

GUIMARÃES, Leandro; Sua aplicação Java nas nuvens: de conceitos a exemplos em AWS e Google App Engine. 2012. Disponível em:
<<http://www.infoq.com/br/presentations/java-nuvem-aws-gae>> Acesso em: 11 de setembro de 2013.

HONORATO, G.; Conhecendo o marketing. São Paulo. Manole. 2004.

HSU, S.S.; As Bases do Cloud Computing. 2009. Disponível em:
<<http://fatecjl.edu.br/TCC/2009-2/tcc-57.pdf>>. Acesso em: 01 de março de 2013.

ING; Estudo de Caso: Grupo ING Disponível em:
<<http://www.itpreneurs.com/document/case-study.pdf>> Acesso em: 11 de setembro de 2013.

MELO, T. R.; Computação em nuvem. Universidade do Vale do Sapucaí, 2009.

MICROSOFT. Visão geral do Traffic Manager. 2013. Disponível em:
<<http://msdn.microsoft.com/pt-br/library/windowsazure/hh744833.aspx>> Acesso em 27 de agosto de 2013.

MICROSOFT. Windows Azure. 2010, Disponível em:
<<http://www.microsoft.com/brasil/servidores/windowsserver2008/editions/windowsazure.mspx>>. Acessado em: 07 de março 2013.

MICROSOFT; Windows Azure. 2013. Disponível em: <<http://www.windowsazure.com/en-us/develop/net/fundamentals/intro-to-windows-azure/?fb=pt-br>>. Acesso em: 27 de abril 2013.

MIYAGUSKU, R. Desenvolvendo os Recursos do CSS. São Paulo. Digerati Books, 2007

READHAT. Why the future of the cloud is open. 2012. Disponível em:
<<http://www.redhat.com/rhecm/restrhecm/jcr/repository/collaboration/jcr:system/jcr:versionStorage/7dc14d8f0a0526016894ca8fec26281a/4/jcr:frozenNode/rh:pdfFile.pdf>>. Acesso em: 03 de junho de 2013.

ROGER, Grimes; Rogue employees and clueless users can cause even more damage than the bad guys outside the gate. 2010. Disponível em: <<http://www.infoworld.com/d/security-central/the-true-extent-insider-security-threats-281>>. Acesso em: 03 de junho de 2013.

SILVA, Mauricio Samy; HTML 5 - A Linguagem de Marcação que Revolucionou a Web. NOVATEC. 2011.

SILVA, Mauricio Samy; JavaScript – Guia do Programador. NOVATEC. 2010.

SILVA, Mauricio Samy; JQuery – A Biblioteca do Programador JavaScript. NOVATEC. 2012.

SNOWMAN, G. 2010. Diferença nos tipos de Computação nas Nuvens. Disponível em:
<http://www.solidq.com/sqj/pt/Documents/2010_August_Issue/SQJ%20002.pdf>. Acesso em: 28 de maio de 2013.

TAURION, C.; **Cloud Computing: Transformando o mundo da Tecnologia da Informação**. Editora Brasport. Rio de Janeiro 2009.

TAURION, C.; **Bate papo sobre Cloud Computing**. 2009. Disponível em: <https://www.ibm.com/developerworks/community/blogs/ctaurion/entry/seguran_c3_a7a_em_cloud_computing3?lang=en>. Acesso em: 06 de junho de 2013.

TAURION, C.; **Vantagens da computação em nuvem**. 2010. Disponível em: <<http://www.youtube.com/watch?v=HJre77TPpSw>>. Acesso em: 05 de abril de 2013.

TLCBrazil. Disponível em: <https://www.ibm.com/developerworks/community/blogs/tlcbr/entry/seguranca_cloud_computing?lang=en>. Acesso em: 03 de junho de 2013.

TUDE, Eduardo.; **Service Level Agreement (SLA)**. 2003. Disponível em: <<http://www.teleco.com.br/DVD/PDF/tutorialsbla.pdf>> acesso em 19 de julho de 2013.

WALDEMIR, Cambiucci.; **Introdução sobre o SQL Azure**. 2011. Disponível em: <<http://msdn.microsoft.com/pt-br/library/hh147515.aspx>>. Acesso em: 27 de abril de 2013.

APÊNDICES

APÊNDICE 1
CONFIGURAÇÃO DO REPOSITÓRIO SUBVERSION

Configuração do repositório *subversion*.

Para que seja possível o uso do controle de versão é necessário configurar um repositório, no caso desse projeto, o Google Code foi escolhido por já ter sido utilizado anteriormente. Abaixo seguem os passos para a sua configuração.

1. É necessário ter uma conta no Google. Após a criação dela, acessar o endereço <https://code.google.com/intl/pt-BR/> e fazer o *login*.
2. Após isso, na tela seguinte ir até a opção ‘*Create a new project*’ como ilustrado na Figura 01.



Fig. 01 – Criação do projeto no repositório. **Fonte:** Elaborado pelo autor.

3. A seguir é necessário colocar as informações do projeto criado, como nome, licença do código fonte e o tipo de controlador de versão que será utilizado, neste caso específico escolha o *subversion* (SVN). A Figura 2 ilustra a tela para preenchimento destas informações.

The screenshot shows a web-based form for creating a new project. The fields filled in are:

- Project name:** photoupload (with an error message: "Example: my-project-name" and "Invalid project name")
- Project summary:** Photo Upload
- Description:** Upload de fotos
- Version control system:** Subversion (radio button selected)
- Source code license:** Apache License 2.0
- Project label(s):** Academic (with a link to "add another row")

At the bottom is a red-bordered "Create project" button.

Fig. 02 – Preenchimento com as informações do projeto. **Fonte:** Elaborado pelo autor.

4. Após isso serão apresentados o endereço do repositório criado (<https://photoupload.googlecode.com/svn/trunk/>) e o *link* (googlecode.com password), necessários para a configuração do controle de versão. Através desse *link* será possível a obtenção da senha do repositório, para ser utilizada por todos os membros do projeto. A figura 03 ilustra a tela de obtenção do *link* para acesso ao repositório.

The screenshot shows the "Source" tab of the photoupload project page. It includes the following sections:

- Checkout:** Buttons for "Browse" and "Changes". A note says: "How-to: Explore this project's source code by clicking the "Browse" and "Changes" links above."
- Command-line access:** Instructions for HTTPS checkout:


```
# Project members authenticate over HTTPS to allow committing changes.
svn checkout https://photoupload.googlecode.com/svn/trunk/ photoupload --username rafael.dsmoreira@gmail.com
```

When prompted, enter your generated googlecode.com password.
- Anonymous access:** Instructions for anonymous checkout:


```
# Non-members may check out a read-only working copy anonymously over HTTP.
svn checkout http://photoupload.googlecode.com/svn/trunk/ photoupload-read-only
```

Fig. 03 – Endereço para acesso ao repositório. **Fonte:** Elaborado pelo autor.

Após seguir esses passos, o repositório estará devidamente configurado para ser usado com o SVN.

APÊNDICE 2
CONFIGURAÇÃO DO CONTROLE D VERSÃO NO
NETBEANS

Configuração do controle de versão no NetBeans

A IDE de desenvolvimento escolhida foi o NetBeans IDE 7.3.1 por ser uma ferramenta já conhecida pelos membros do projeto. A seguir serão apresentados os passos utilizados para configuração do SNV no NetBeans, partindo do pressuposto de que o mesmo já esteja instalado. O tutorial de criação do repositório é pré-requisito para que seja possível prosseguir.

1. No menu ‘equipe > subversion > efetuar *check-out*’ são configuradas as informações necessárias para o acesso ao repositório, como mostra a Figura 04.

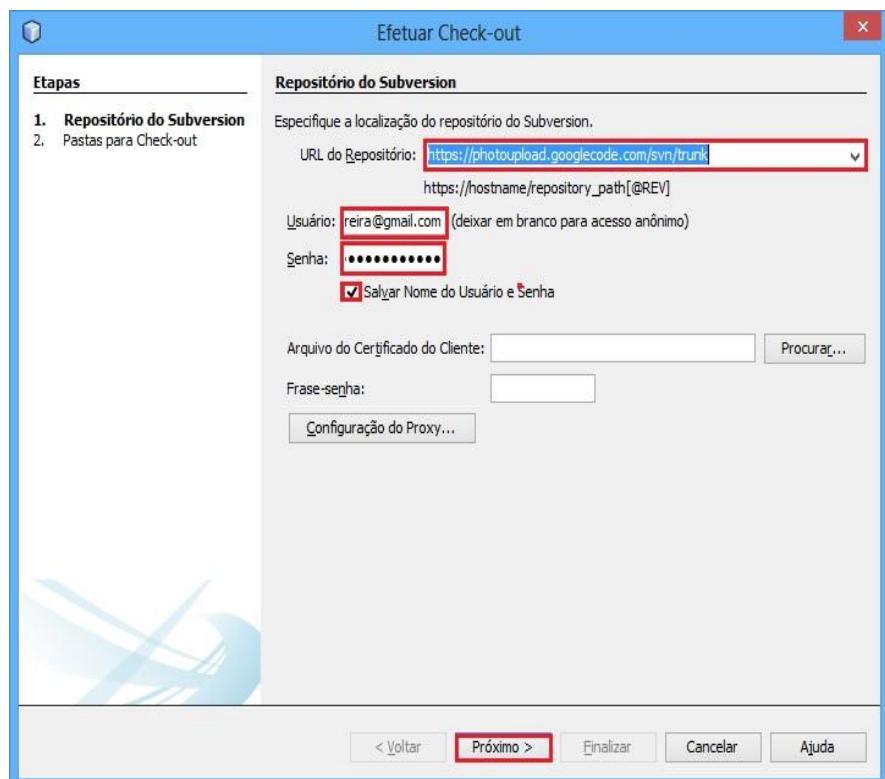


Fig. 04 – Configuração para acesso ao repositório. **Fonte:** Elaborado pelo autor

2. Após realizar a configuração anterior, clique no botão ‘Próximo’. Será apresentada uma segunda tela de configuração, nessa tela escolha no campo ‘Pasta Local’ o caminho ‘C:\xampp\htdocs’ que é a pasta servidor local como mostra a Figura 05.

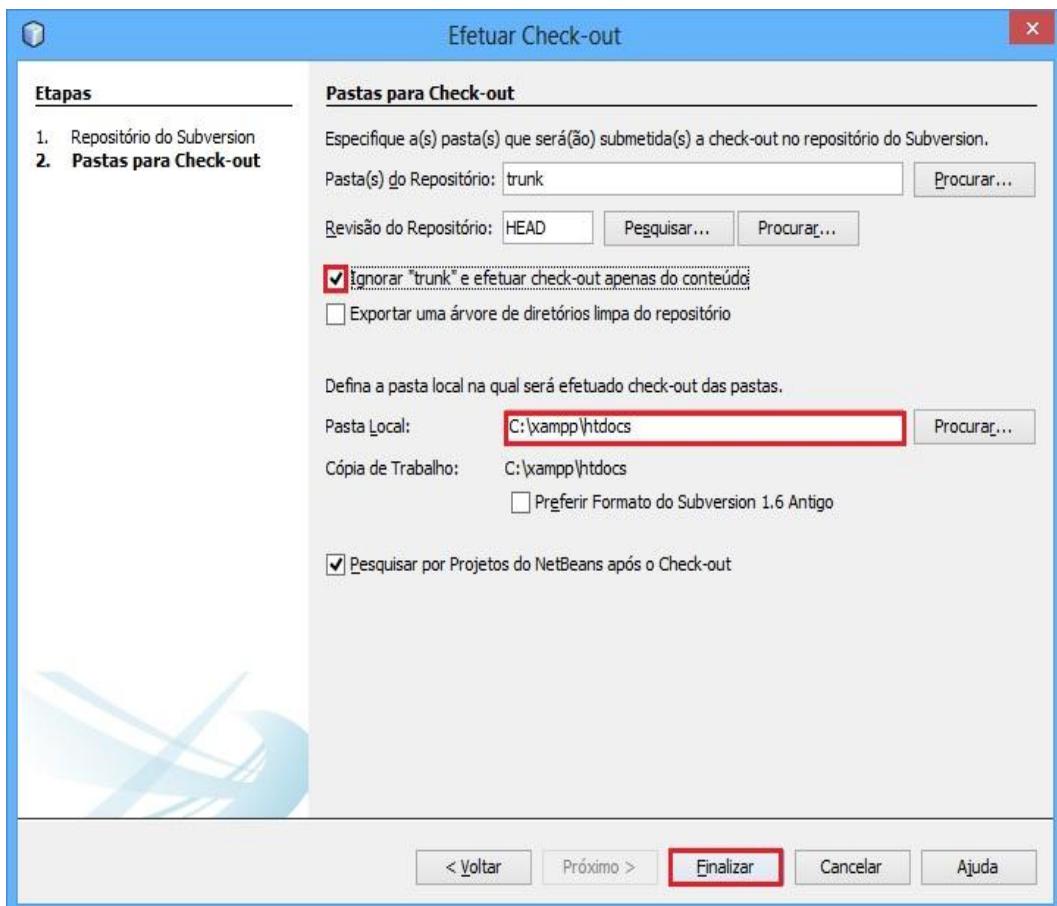


Fig. 05 – Configuração para pasta do servidor local. **Fonte:** Elaborado pelo autor.

3. Após o término das configurações anteriores, ao clicar em ‘Finalizar’, automaticamente será exibida a tela para a criação do projeto, clique em “Criar Projeto”, como mostra a Figura 06.

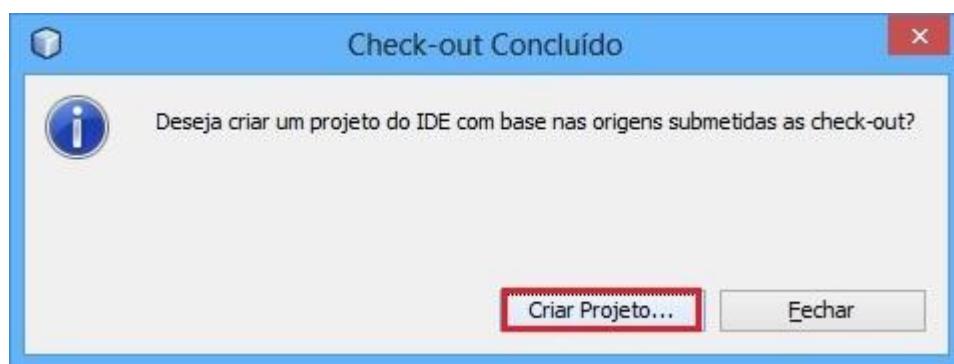


Fig. 06 – Criação do projeto na IDE. **Fonte:** Elaborado pelo autor.

4. Na Figura 07 escolha a linguagem do novo projeto, no caso desse tutorial foi escolhido o PHP.

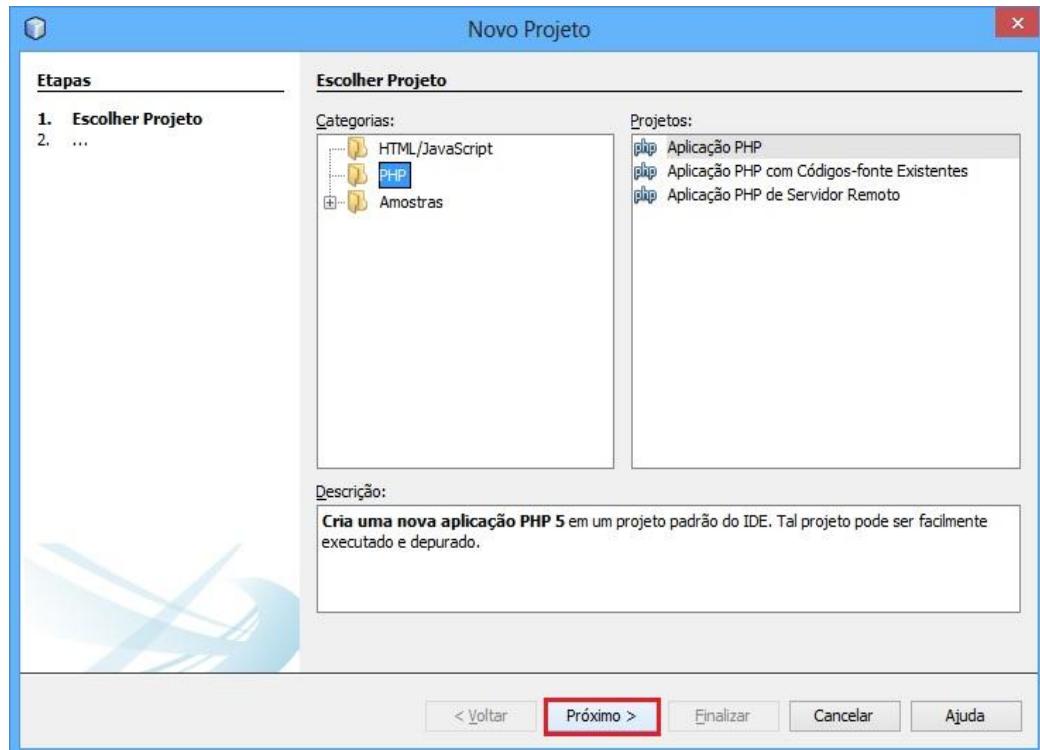


Fig. 07– Escolha da linguagem do projeto. **Fonte:** Elaborado pelo autor.

5. Por fim escolha o nome e a localização onde o projeto será salvo. Vale lembrar que precisa ser na mesma pasta onde foi configurado anteriormente o repositório (c:\xampp\htdocs). Feito isso, está concluída a parte de configuração do repositório e criação do projeto.

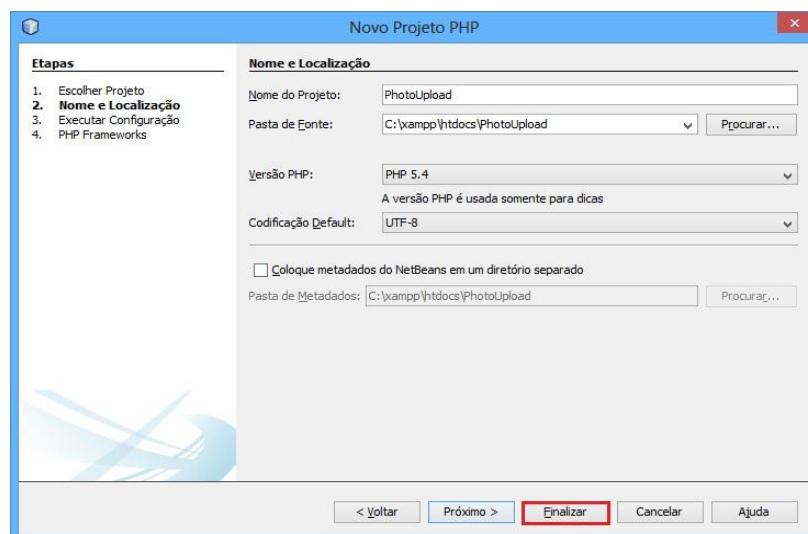


Fig. 08 – Conclusão da configuração **Fonte:** Elaborado pelo próprio autor.

Com isso as configurações do projeto e a sincronização com o repositório *Subversion* estão completas, possibilitando a todo o grupo de desenvolvimento do projeto desenvolver em conjunto.

**APÊNDICE 3
AMAZON ROUTE 53**

Amazon Route 53

Esse é o serviço que possibilita o acesso à instâncias do Elastic Beanstalk e do EC2 através de um domínio público por meio de DSN que a AWS oferece. Esse tutorial parte do pressuposto de que o usuário já tenha um domínio criado. Neste tutorial foi criado um gratuito no *dot.tk* (www.dot.tk). Vale lembrar que esse tutorial não aborda o uso deste serviço com instâncias do EC2, que é feita através de um IP elástico.

1. Clicar na aba ‘*Services*’ no console de serviços da AWS e escolher o Route 53, após a janela do serviço se abrir clicar em ‘*Create Hosted Zone*’ como ilustrado na Figura 09.

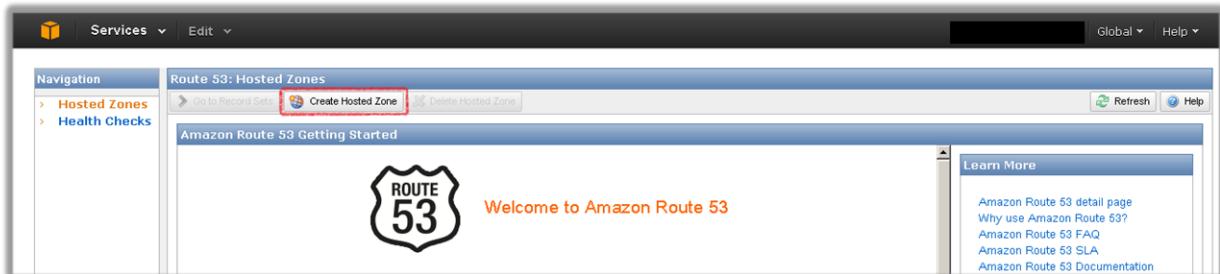


Fig. 09 – Tela inicial do Route 53. **Fonte:** Elaborado pelo autor.

2. Na Figura 10, no canto direito há o campo ‘*Domain Name*’, nele será colocado o domínio criado, no caso ‘awstcc.tk’. Após isso clicar em ‘*Create Hosted Zone*’.

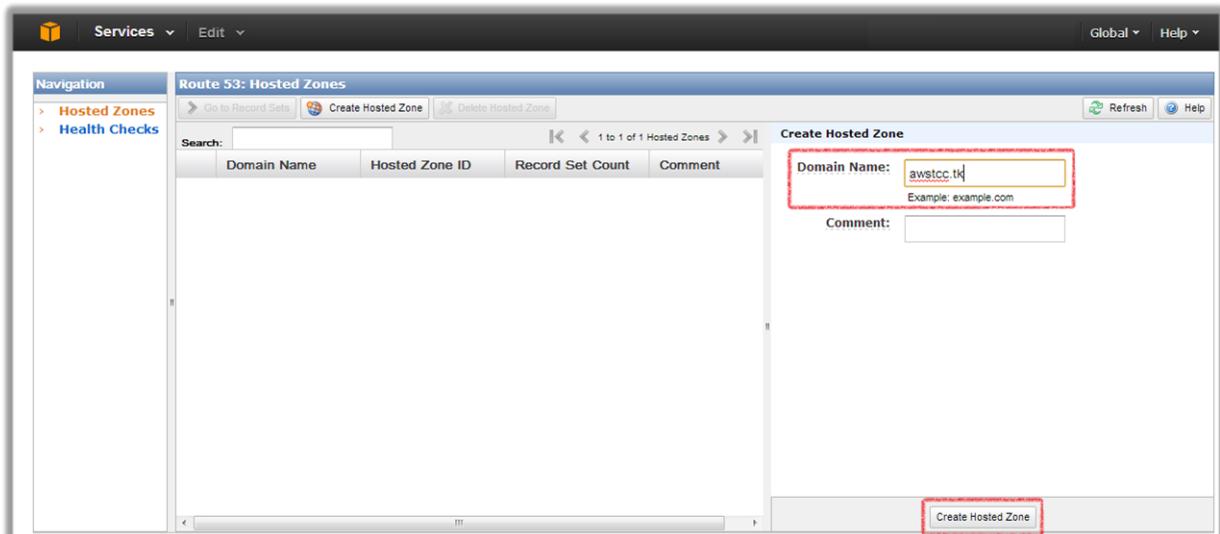


Fig. 10 – Criando a ‘Hosted Zone’. **Fonte:** Elaborado pelo autor.

- No lado direito, no campo ‘*Delegation Set*’ serão mostrados os DNS que foram disponibilizados pelo serviço. Agora selecionar o domínio que foi criado e clicar em ‘*Go to records sets*’ como ilustrado na Figura 11.

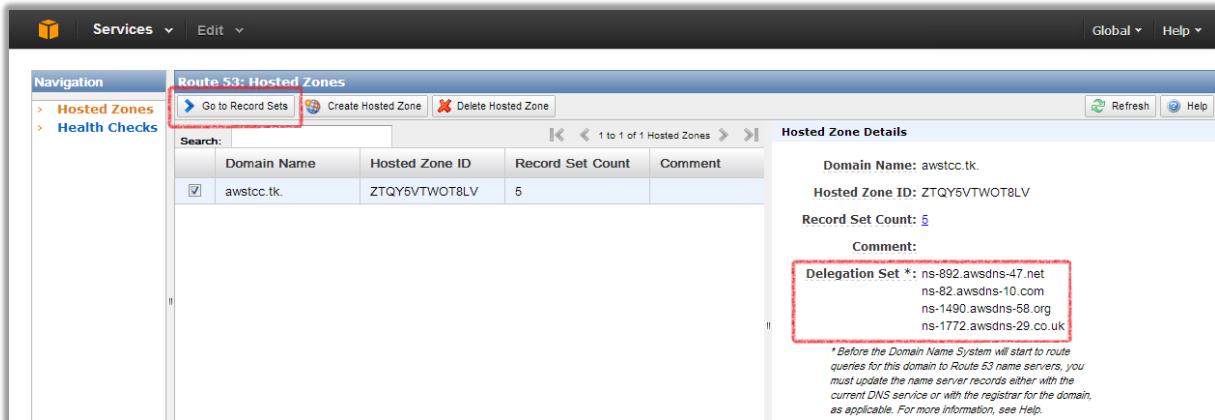


Fig. 11 – Domínios criados. **Fonte:** Elaborado pelo autor.

- Nesta tela ilustrada na Figura 12, no canto superior esquerdo, clicar em ‘*Create record set*’. Do lado direito aparecerão vários campos. O campo ‘*Name*’ é onde será colocado o prefixo para o domínio, na Figura 12 foram criados três registros, um com o prefixo ‘www’, outro com ‘*’ e outro em branco, assim o domínio será reconhecido pelo serviço de várias maneiras. No campo ‘*Alias*’, escolha a opção ‘*Yes*’, uma lista irá aparecer contendo todas as instâncias ativas da conta.

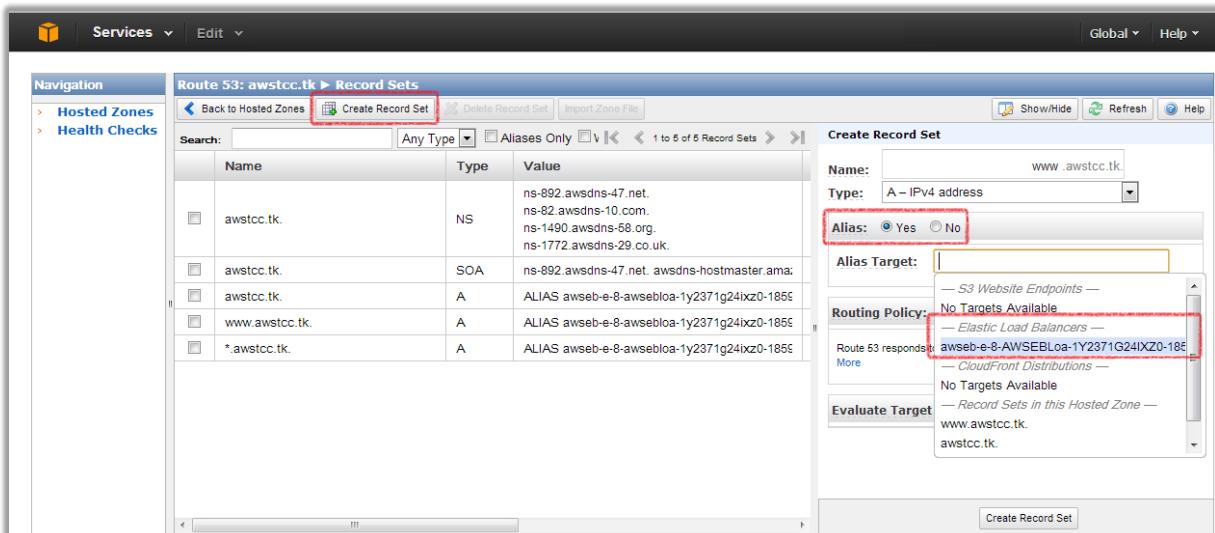


Fig. 12 – Criando um ‘Record Set’. **Fonte:** Elaborado pelo autor.

- Após escolher a instância desejada, clique em ‘Create Record Set’ no canto inferior direito e pronto, a rota está criada.
- Por fim, é preciso acessar a página de configuração do domínio criado e colocar os DSN fornecidos pelo Route 53, para que o domínio aponte para esse serviço. A Figura 13 ilustra a configuração realizada.

Host Name	IP Address	
ns-1772.awsdns-29.co.uk	n/a	Remove
ns-892.awsdns-47.net	n/a	Remove
ns-1490.awsdns-58.org	n/a	Remove
ns-82.awsdns-10.com	n/a	Remove
Add new		

Fig. 13 – Colocando DNS do Route 53 no domínio. **Fonte:**Elaborado pelo autor.

Com isso conclui-se a configuração do Route 53 para realizar o apontamento dos domínios externos para as instâncias. A Figura 14 ilustra a execução da aplicação.



Fig. 14– Acessando aplicação através do Route 53. **Fonte:** Elaborado pelo autor.

APÊNDICE 4
AMAZON ELASTIC BEANSTALK

Amazon Elastic Beanstalk

O Elastic Beanstalk é o local onde será criado o ambiente de desenvolvimento, com suporte a PHP, Python, Node.js dentre outras linguagens. A criação deste ambiente automaticamente instancia uma máquina virtual no EC2, pois é ela quem ficará encarregada de realizar todo o processamento da nuvem.

A baixo segue os passos para implantar uma aplicação utilizando o Elastic Beanstalk.

1. Clicar na aba ‘*Services*’ no console de serviços da AWS e escolha o Elastic Beanstalk, após a janela do serviço se abrir clicar em ‘*Create a New Application*’ como ilustrado na Figura 15.

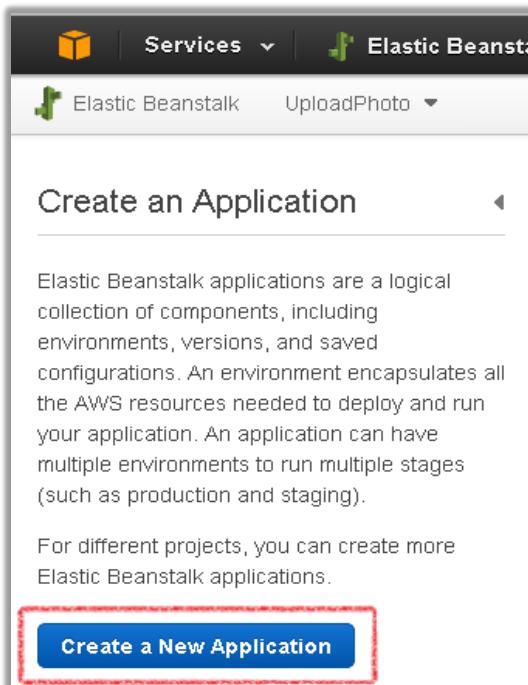


Fig. 15 – Tela inicial do Elastic Beanstalk.
Fonte: Elaborado pelo autor.

2. No campo ‘*Aplication name*’ inseri o nome da aplicação. No campo ‘*Description*’ é possível colocar uma descrição para ela, porém é facultativo. Após inserir esses dados, clicar em ‘*Create*’ como demonstrado na Figura 16.

Application Information

To create a new application, enter the details of your application. [Learn more](#)

Application name:	Must be less than 100 characters and cannot contain a /
Description:	Optional.

[Cancel](#) [Create](#)

Fig. 16 – Criando aplicação. Fonte: Elaborado pelo autor.

3. Na tela de configuração do ambiente de desenvolvimento, marcar a caixa de diálogo ‘*Launch a new environment running this application*’ para que o Beanstalk permita configurar as opções de importar um projeto já existente e a criação de instâncias no RDS.
4. No campo ‘*Predefined Configuration*’ escolha a linguagem desejada para o ambiente de desenvolvimento. Por padrão a instância criada no EC2 será um servidor Linux de 64 bits, para modificar clicar em ‘*Change Defaults*’.
5. No campo ‘*Environment type*’ existem duas opções. A primeira é a ‘*Single Instance*’, opção que utiliza apenas os recursos disponíveis pelo servidor instanciado, ou seja, não existe escalabilidade. A segunda é a ‘*Load balancing, autoscaling*’, essa permite que o Beanstalk faça o balanceamento de cargas entre instâncias e use a escalabilidade, com a criação de instâncias adicionais quando necessário. A Figura 17 demonstra a configuração realizada.

Environment Type

Choose whether to launch an environment and if so what kind.

Launch a new environment running this application

Predefined configuration: PHP

Elastic Beanstalk will create an environment running PHP 5.4 on 64bit Amazon Linux. [Change Defaults](#)

Environment type:	Load balancing, autoscaling	Learn more
	Single instance	
	Load balancing, autoscaling	

[Cancel](#) [Continue](#)

Fig. 17 – Configurando o ambiente de desenvolvimento. Fonte: Elaborado pelo autor.

6. Após clicar em ‘*Continue*’, será perguntado se o usuário quer que a Amazon coloque uma aplicação de exemplo na instância ou se quer fazer o *upload* de uma aplicação já existente. No segundo caso clicar em ‘*Choose File*’ e escolha o arquivo (tem que ser no formato .zip) e clicar em ‘*Continue*’. A Figura 18 ilustra esses passos.

The screenshot shows the 'Application Version' configuration step. The 'Source' dropdown is set to 'Upload your own' and the file 'PhotoUpload.zip' is selected. The 'Continue' button is highlighted with a red box.

Fig. 18 – Fazendo upload da aplicação. **Fonte:** Elaborado pelo autor.

7. Na Figura 19 serão pedidas as informações do ambiente de desenvolvimento. Este é o local onde o usuário irá colocar o nome do ambiente no campo ‘*Environment name*’ e a URL de acesso à aplicação no campo ‘*Environment URL*’. Clique em ‘*Check availability*’ para verificar se essa URL está disponível.

The screenshot shows the 'Environment Information' configuration step. The 'Environment name' field contains 'photoupload-env'. The 'Environment URL' field contains 'photoupload-env.elasticbeanstalk.com' and the 'Check availability' button is highlighted with a red box. The 'Description' field is optional with a maximum of 200 characters.

Fig. 19 – Nomeando o ambiente. **Fonte:** Elaborado pelo autor.

8. Na Figura 20 o Beanstalk dará duas opções. Onde a primeira permite a criação de uma instância no RDS e a segunda permite colocar esse ambiente dentro de uma rede virtual com o VPC.

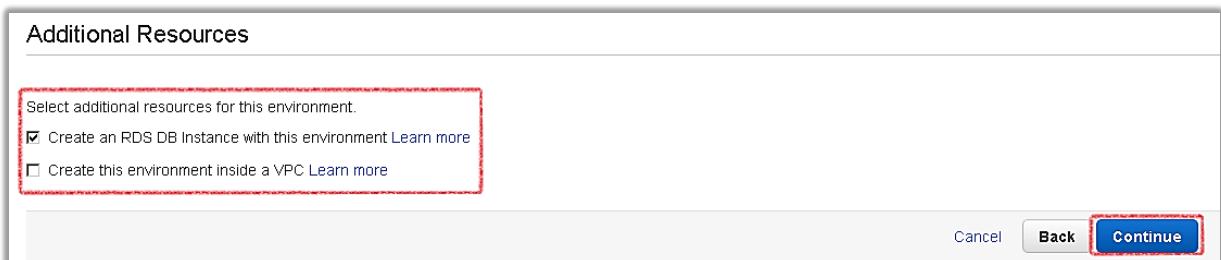


Fig. 20 – Configurando recursos adicionais. **Fonte:** Elaborado pelo autor.

9. No caso de escolha da criação de uma instância no RDS, aparecerá uma tela de configuração pedindo dados como o tipo de instância, utilização de *key pairs* e perfil da instância. A Figura 21 demostra a configuração realizada.

Configuration Details

Modify the following settings or click Continue to accept the default configuration. [Learn more](#).

Instance type: t1.micro

Determines the processing power of the servers in your environment.

EC2 key pair: Select a key pair Refresh

Optional: Enables remote login to your instances.

Email address: Optional: Get notified about any major changes to your environment.

Application health check URL: Enter the relative URL that ELB continually monitors to ensure your application is available.

Instance profile: aws-elasticbeanstalk-ec2-role Refresh

Grants your environment specific permissions under your AWS account. [Learn more](#).

Fig. 21 – Configurando a instância usada pelo Elastic Beanstalk. **Fonte:** Elaborado pelo autor.

10. Após os passos anteriores o ambiente está configurado. A Figura 22 mostra a tela inicial do Beanstalk, onde serão listados todos os ambientes existentes.

Elastic Beanstalk photo.upload UploadPhoto

Create an Application All Applications photo.upload

Elastic Beanstalk applications are a logical collection of components, including environments, versions, and saved configurations. An environment encapsulates all the AWS resources needed to deploy and run your application. An application can have multiple environments to run multiple stages (such as production and staging).

For different projects, you can create more Elastic Beanstalk applications.

[Create a New Application](#)

Filter by Application Name:

Actions

Running versions: PhotoUploadv7.0
Last modified: 2013-10-01 21:22:01 UTC-0300
URL: uploadphoto.elasticbeanstalk.com

Fig. 22 – Ambiente de desenvolvimento criado. **Fonte:** Elaborado pelo autor.

Entrando no ambiente configurado, serão encontradas várias opções de monitoramento e configuração, saúde do serviço e versão da aplicação que está em uso, assim como o *link* para acessar a aplicação no Beanstalk. A Figura 23 ilustra o ambiente que foi configurado

The screenshot shows the AWS Elastic Beanstalk console for the 'photo.upload' application. The left sidebar has links for Dashboard, Configuration, Logs, Monitoring, Alarms, and Events. The main area has tabs for Overview, Health (Green), Running Version (PhotoUploadV7.0), Configuration (PHP 5.4), and Actions (Refresh, Upload and Deploy, Edit). Below these are sections for Recent Events and Application Version History. The Recent Events table shows three log entries:

Time	Type	Details
2013-10-01 21:22:01 UTC-0300	INFO	Environment update completed successfully.
2013-10-01 21:22:01 UTC-0300	INFO	New application version was deployed to running EC2 instances.
2013-10-01 21:21:53 UTC-0300	INFO	Deploying new version to instance(s).

Fig. 23 – Página principal do ambiente de desenvolvimento. **Fonte:** Elaborado pelo autor.

A Figura 24 mostra a aplicação que foi implementada e implantada pelo Beanstalk sendo acessada pelo link fornecido pelo serviço.



Fig. 24 – Aplicação sendo executada no Elastic Beanstalk. **Fonte:** Elaborado pelo autor.

Na opção ‘*Configuration*’ é possível escolher as configurações de escalabilidade, tipo da instância, configurações de notificação, configurações de *logs* e de rede. A Figura 25 demonstra essas configurações.

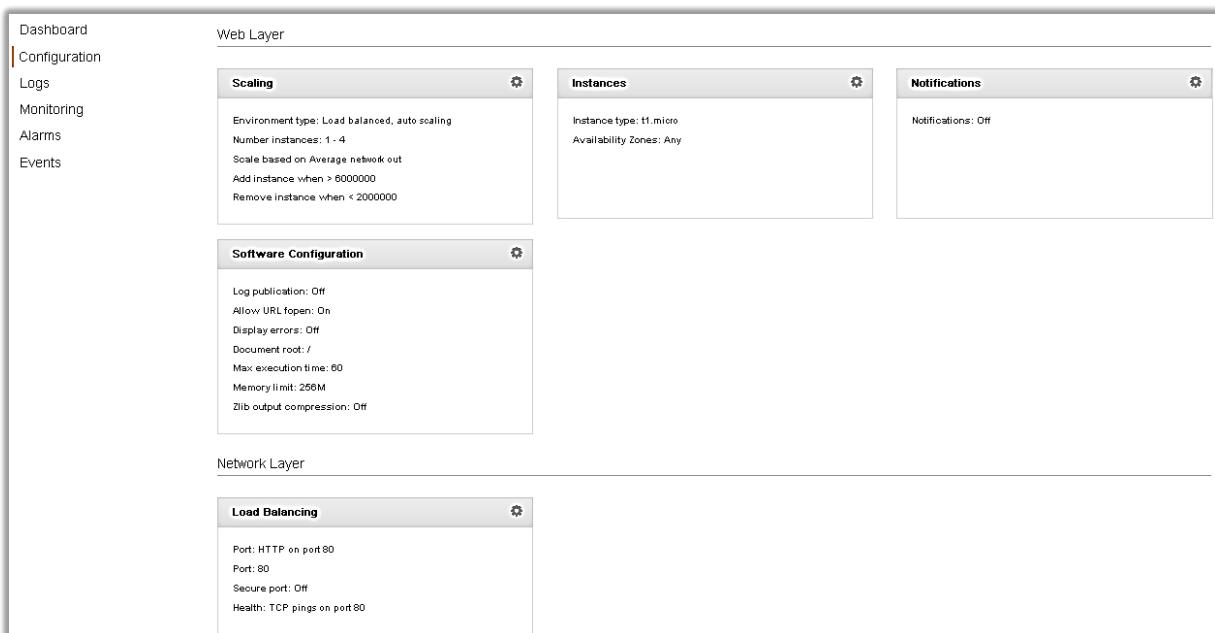


Fig. 25 – Tela de configuração do ambiente de desenvolvimento. **Fonte:** Elaborado pelo autor.

APÊNDICE 5
AMAZON ELASTIC COMPUTING CLOUD (EC2)

Amazon Elastic Compute Cloud (EC2)

O EC2 é o serviço que oferece instâncias para que sejam usadas em uma rede virtual ou por aplicações que sejam um SaaS e portanto irão utilizar o processamento delas. Este tutorial mostra como criar e acessar as instâncias tanto para Linux, quanto para Windows.

1. Acessar o serviço do Elastic Compute Cloud, a tela principal será mostrada informando quantas instâncias, *pair keys*, grupos de segurança, *elastic ips*, *snap shots* e outros serviços associados ao uso de instâncias existem criados. Nessa tela clicar em ‘*Launch Instance*’ como demonstrado na Figura 26.

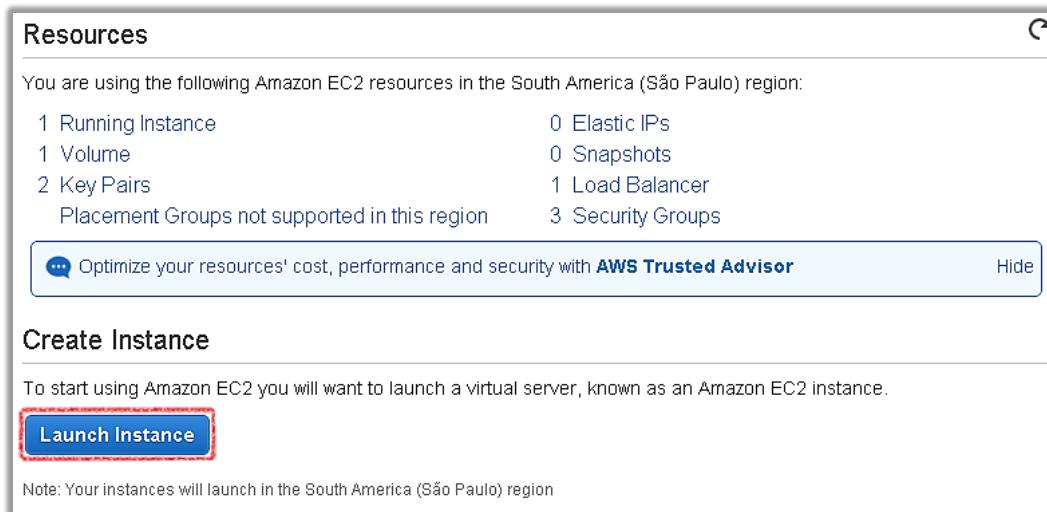


Fig. 26 – Tela inicial do Elastic Cloud Computing. **Fonte:** Elaborado pelo autor.

2. Na configuração da Figura 27 existem três opções para a criação da instância:
 - *Classic Wizard*: O usuário escolhe de acordo com sua necessidade todos os detalhes e configurações da instância.
 - *Quick Launch Wizard*: Será criada uma instância com configurações pré-definidas pela AWS;
 - *AWS Market*: é uma loja *on-line* da Amazon, onde o usuário pode comprar uma aplicação pronta que instantaneamente é implantada junto com uma instância, que é pago, apenas pelo tempo de uso.

Neste passo escolher a opção ‘*Classic Wizard*’ e clicar em ‘*Continue*’.

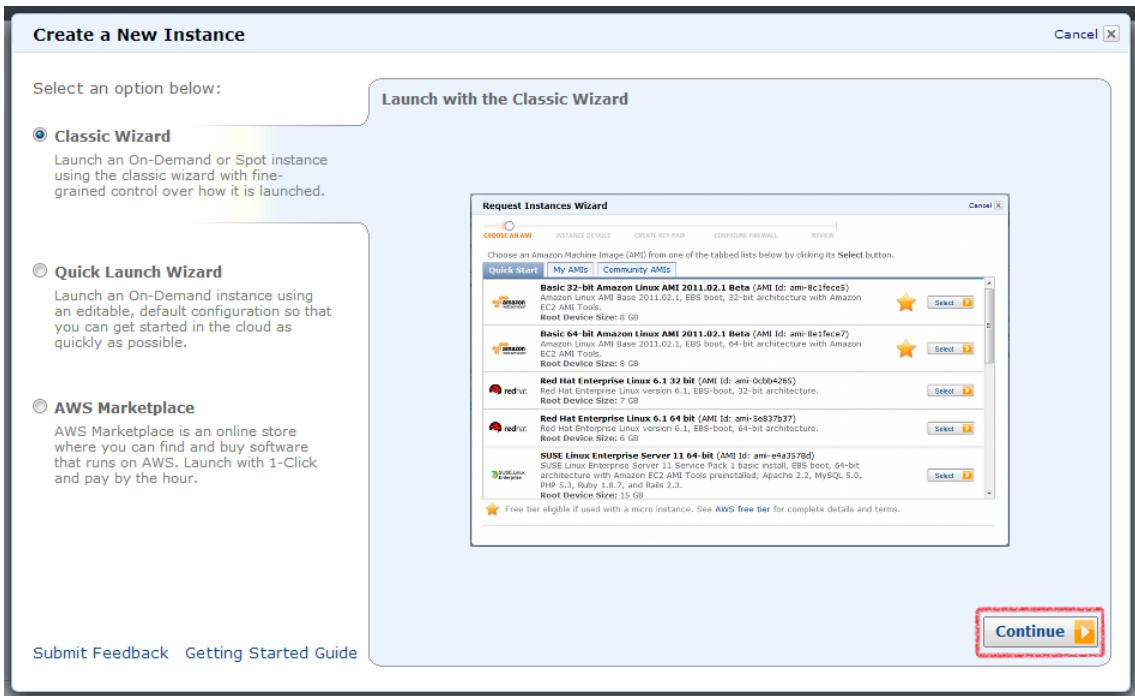


Fig. 27 – Tela para a escolha das instâncias. **Fonte:** Elaborado pelo autor.

3. Na Figura 28 serão exibidos vários tipos de instâncias Linux e Windows, após escolher clique no botão ‘Select’.

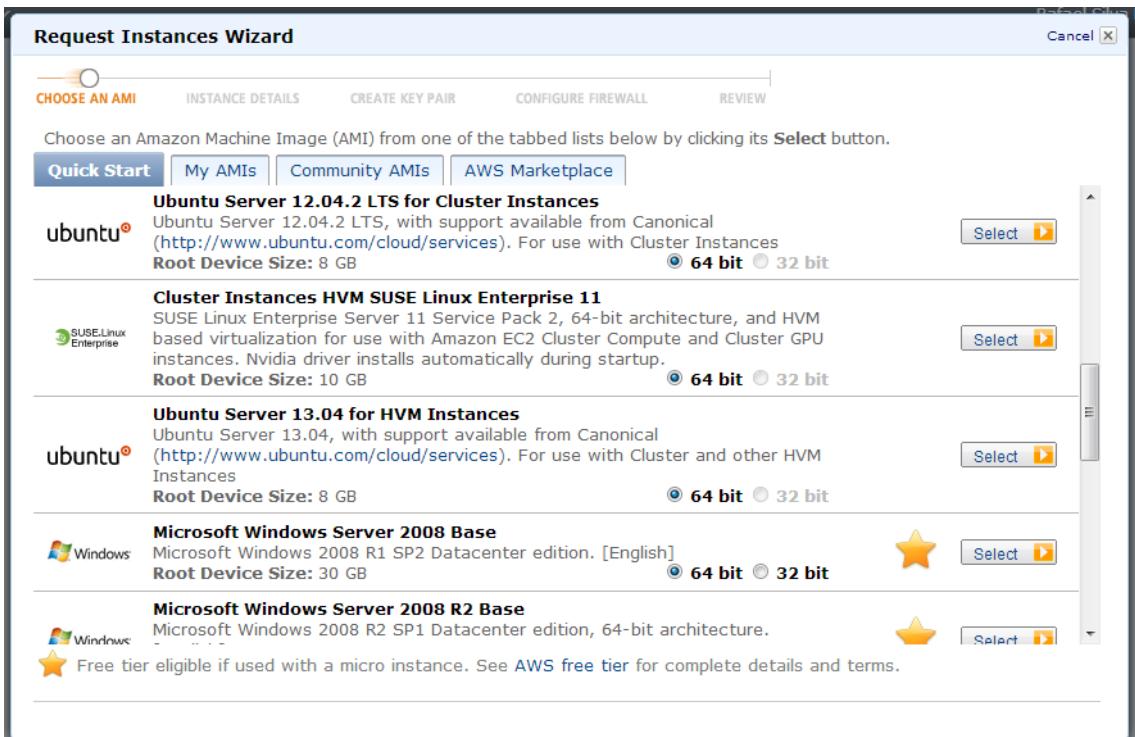


Fig. 28 – Escolha do tipo de instância. **Fonte:** Elaborado pelo autor.

4. Nos passos demostrados na Figura 29 e na Figura 30 serão configurados o tipo e a quantidade de instâncias, assim como a zona de que ela fará parte.

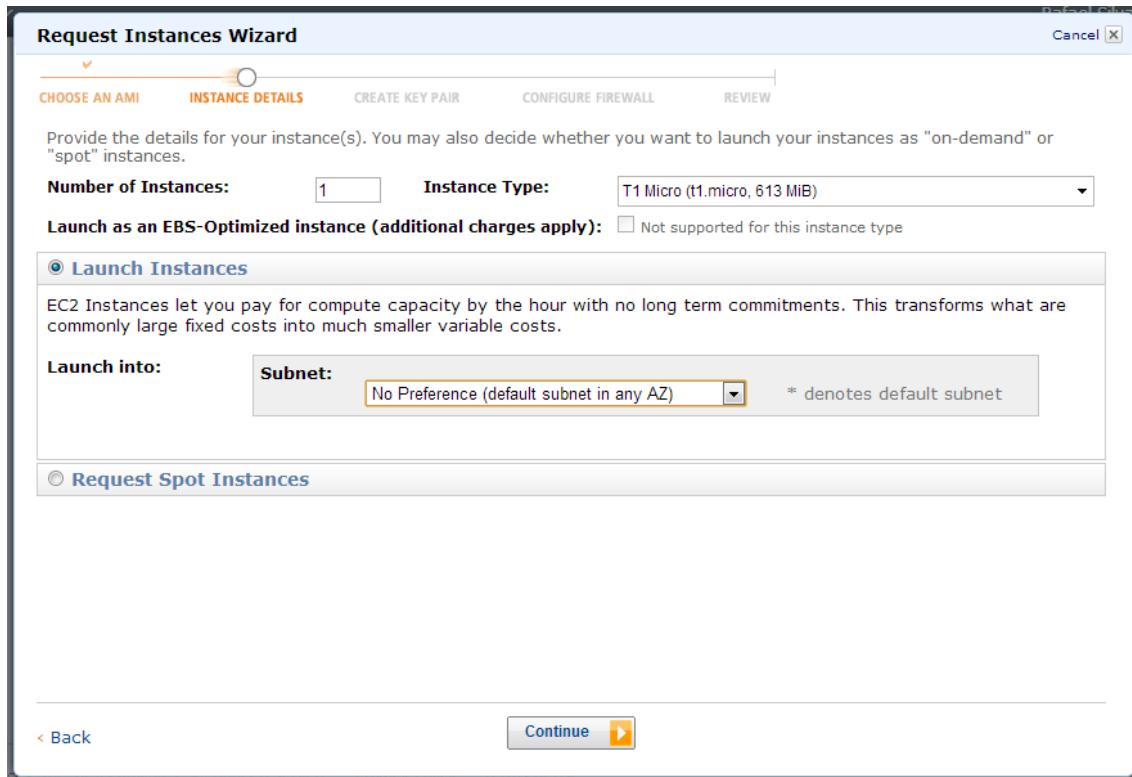


Fig. 29 – Tela para configuração da instância. **Fonte:** Elaborado pelo autor.

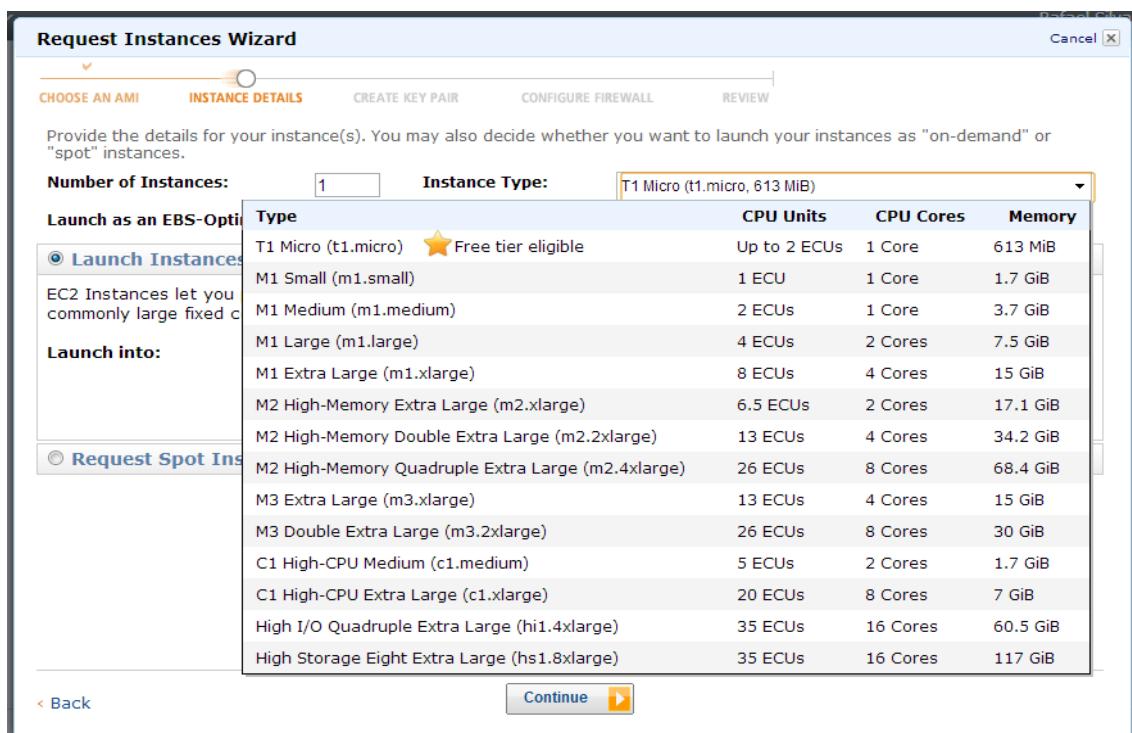


Fig. 30 – Tipos de instância para a escolha. **Fonte:** Elaborado pelo autor.

5. Na Figura 31 serão apresentadas as configurações avançadas, como a escolha de um *kernel* e tipo de memória RAM, ativar ou não o *CloudWatch*, escolha do IAM e criação de um *public IP*.

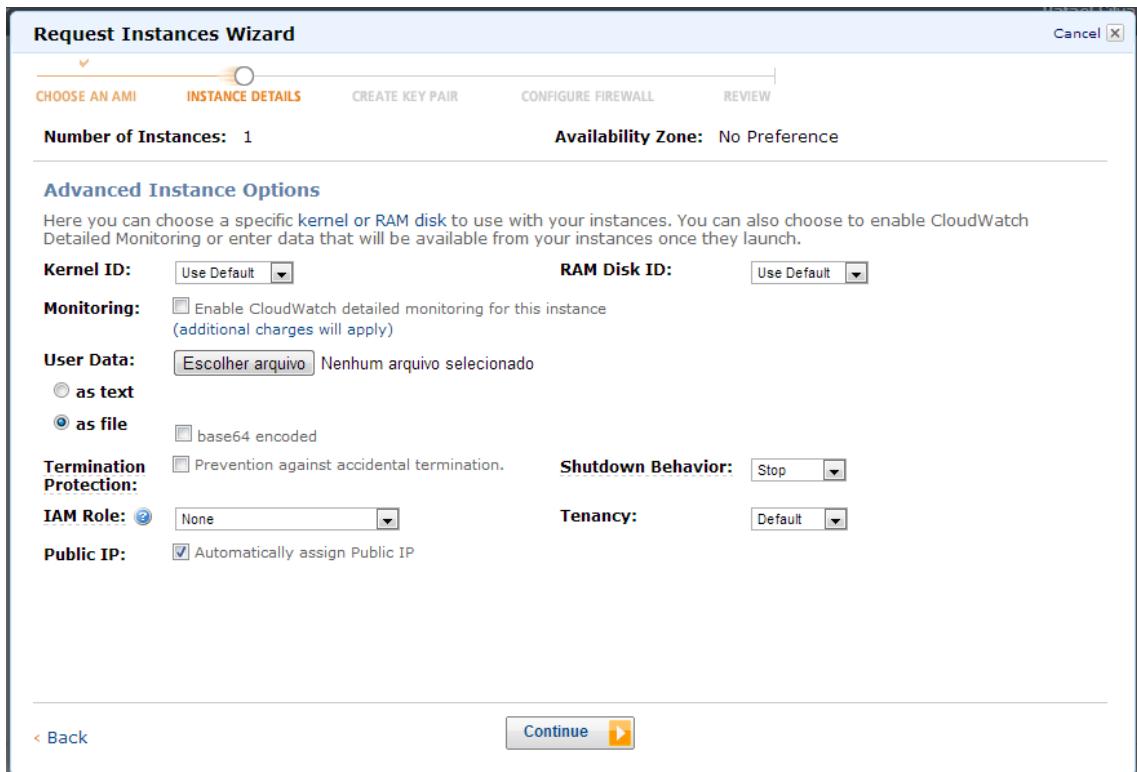


Fig. 31 – Tela de configurações avançadas da instância. **Fonte:** Elaborado pelo autor.

6. Na sequência a Figura 32 demonstra a configuração da memória de armazenamento da instância, com a possibilidade de usar um armazenamento convencional escolhendo o ‘*Root Volume*’ ou utilizar o serviço Elastic Block Store que oferece muitos recursos, porém tem custos adicionais, escolhendo ‘*EBS Volumes*’.

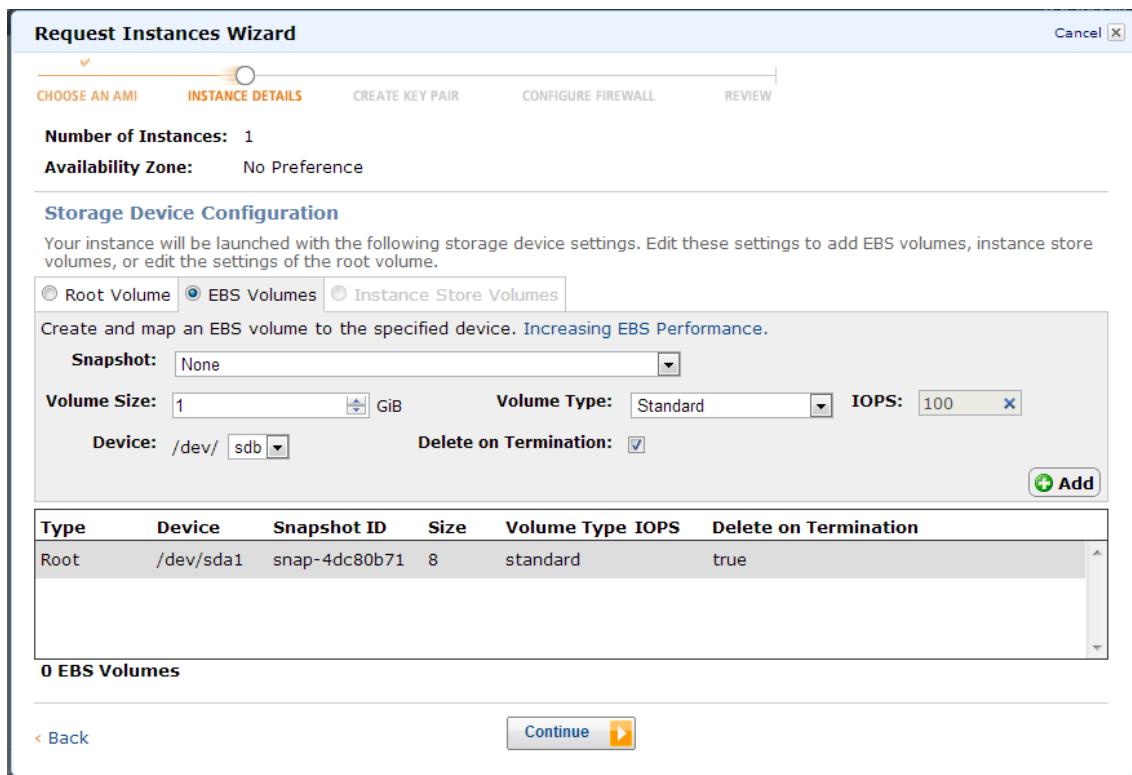


Fig. 32 – Configuração da memória principal da instância. **Fonte:** Elaborado pelo autor.

7. Agora escolher um nome para *key pair*, que é a chave que será usada para acesso à instância como demonstrado na Figura 33.

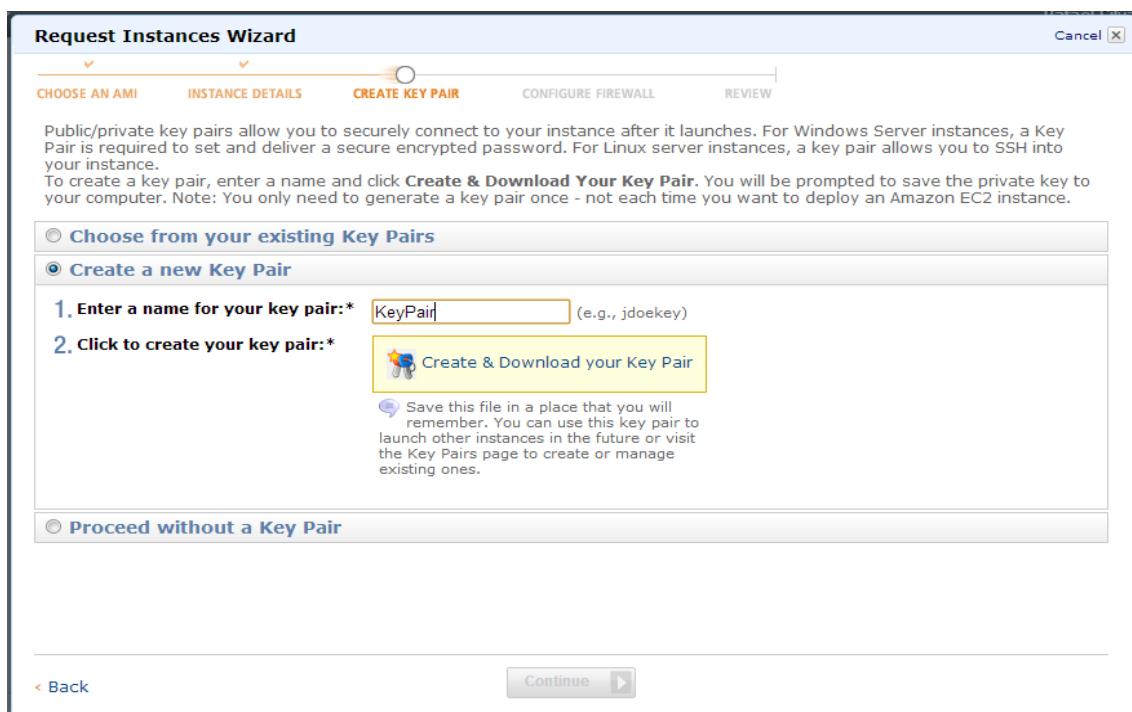


Fig. 33 – Criando uma *key pair*. **Fonte:** Elaborado pelo autor.

8. A seguir na Figura 34 será apresentada a tela de configuração do *firewall* onde serão configuradas as portas de acesso e as regras de segurança para a instância.

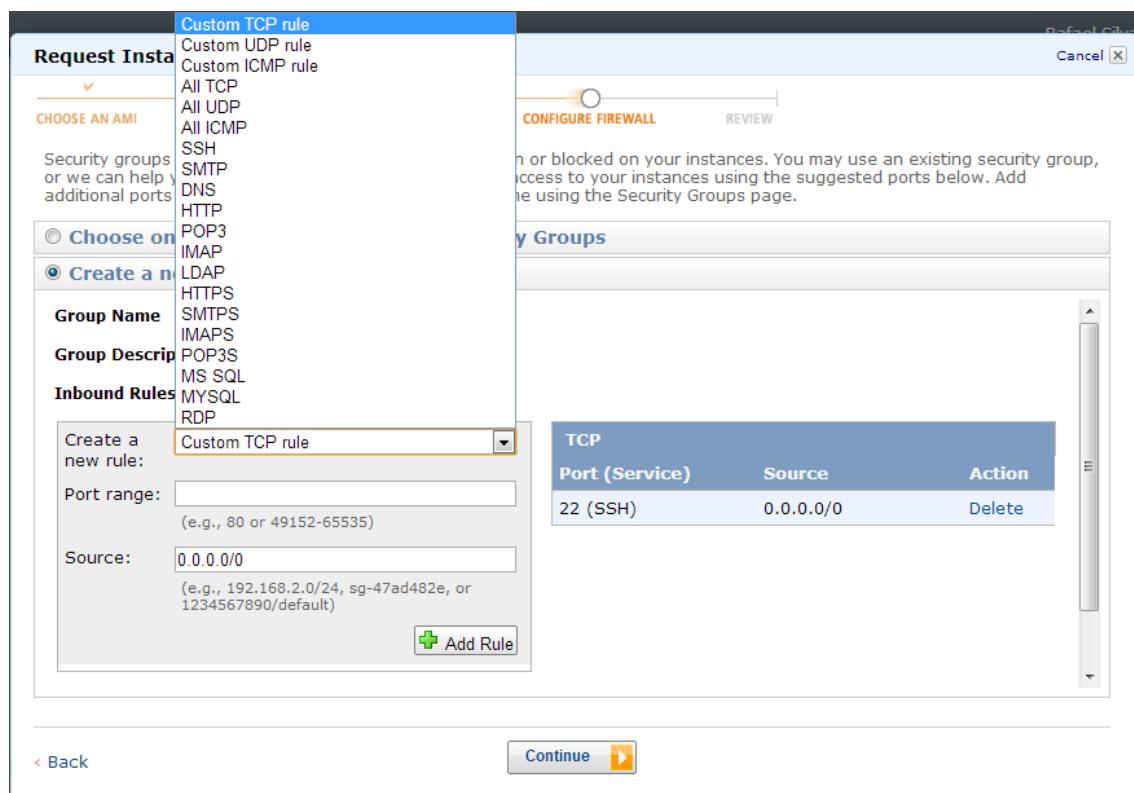


Fig. 34 – Criando regras para o *firewall*. **Fonte:** Elaborado pelo autor.

9. Por fim serão apresentadas as informações de todas as configurações da instância para conferência como ilustrado na Figura 35.

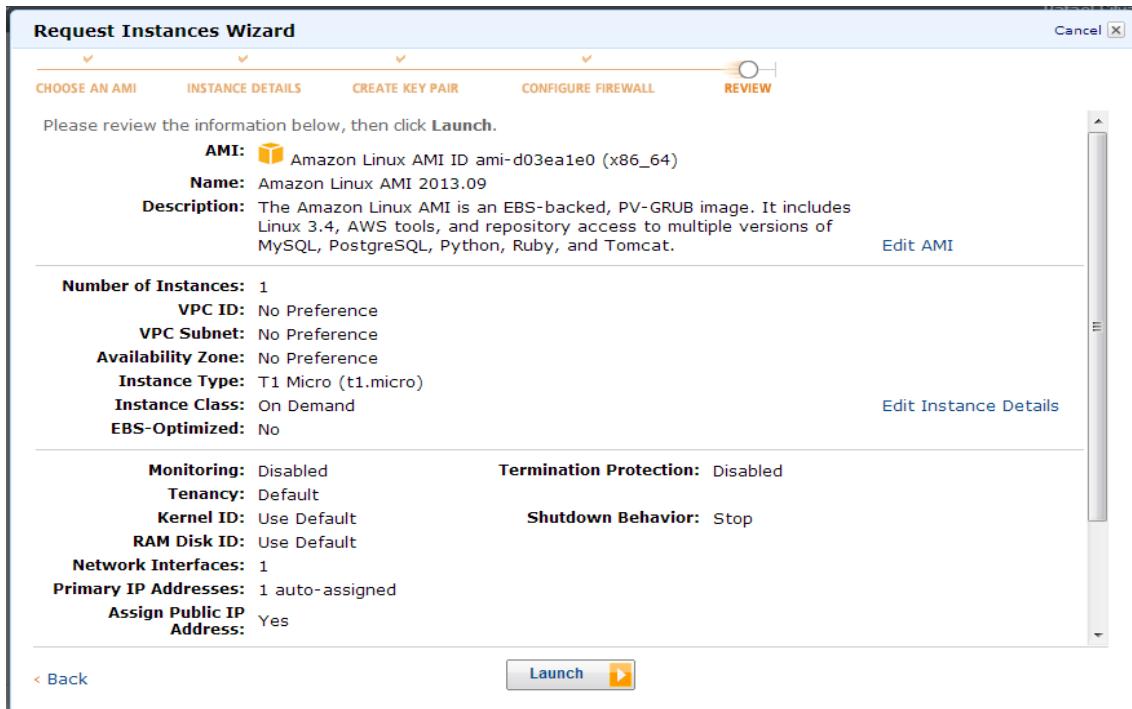


Fig. 35 – Visão geral das configurações. **Fonte:** Elaborado pelo autor.

10. As Figuras 36 e 37 ilustram a instância que foi criada, mas leva alguns minutos para que ela esteja inicializada com suas configurações.

Instances											
	Name	Instance	AMI ID	Root Device	Type	Status	Status Checks	Alarm Status	Monitoring	Security Groups	Key Pair
	empty	i-bca52388	ami-d03ea1e0	ebs	t1.micro	pending	initializing...	none	basic	PhotoUpload Sec.	KeyPair

Fig. 36 – Instância sendo criada no Elastic Cloud Compute. **Fonte:** Elaborado pelo autor.

Instances											
	Name	Instance	AMI ID	Root Device	Type	Status	Status Checks	Alarm Status	Monitoring	Security Groups	Key Pair
	empty	i-bca52388	ami-d03ea1e0	ebs	t1.micro	running	2/2 checks p.	none	basic	PhotoUpload Sec.	KeyPair

Fig. 37 – Instância criada no Elastic Cloud Compute. **Fonte:** Elaborado pelo autor.

11. Para se conectar à instância, após selecioná-la clicar com o botão direito sobre ela e em seguida no menu ‘Connect’ como ilustrado na Figura 38.

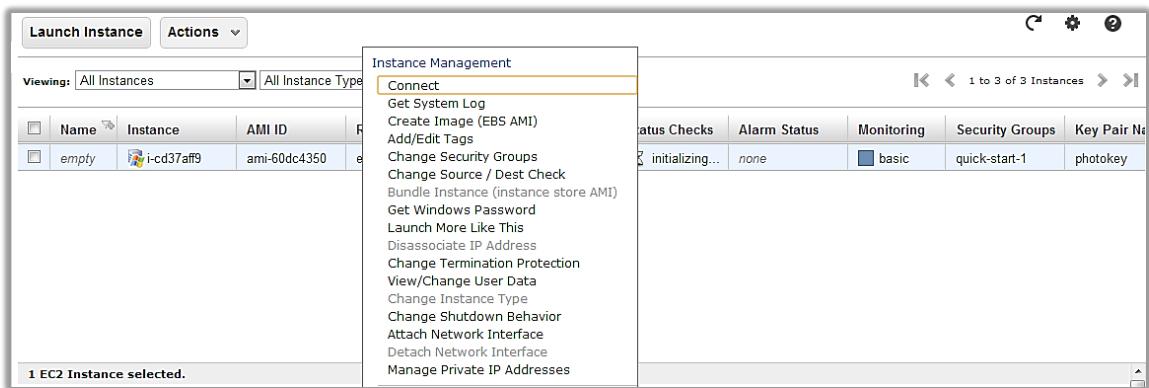


Fig. 38 – Menu de opções da instância. **Fonte:** Elaborado pelo autor.

APÊNDICE 6
CONEXÃO A UMA INSTÂNCIA LINUX

Conexão a uma instância Linux

Essa parte do tutorial assume que o usuário tenha seguido até no apêndice 5 e que a instância escolhida tenha sido com o sistema operacional Linux.

1. Após selecionar a opção ‘*Connect*’, na tela que se abriu selecionar a segunda opção ‘*Connect from your browser using the Java SSH Client*’, lembrando que é necessário ter o Java instalado no computador. No campo ‘*Private key*’ coloque o caminho de onde o arquivo da chave foi salvo e clique em ‘*Launch SSH Client*’. A Figura 39 ilustra essa configuração.

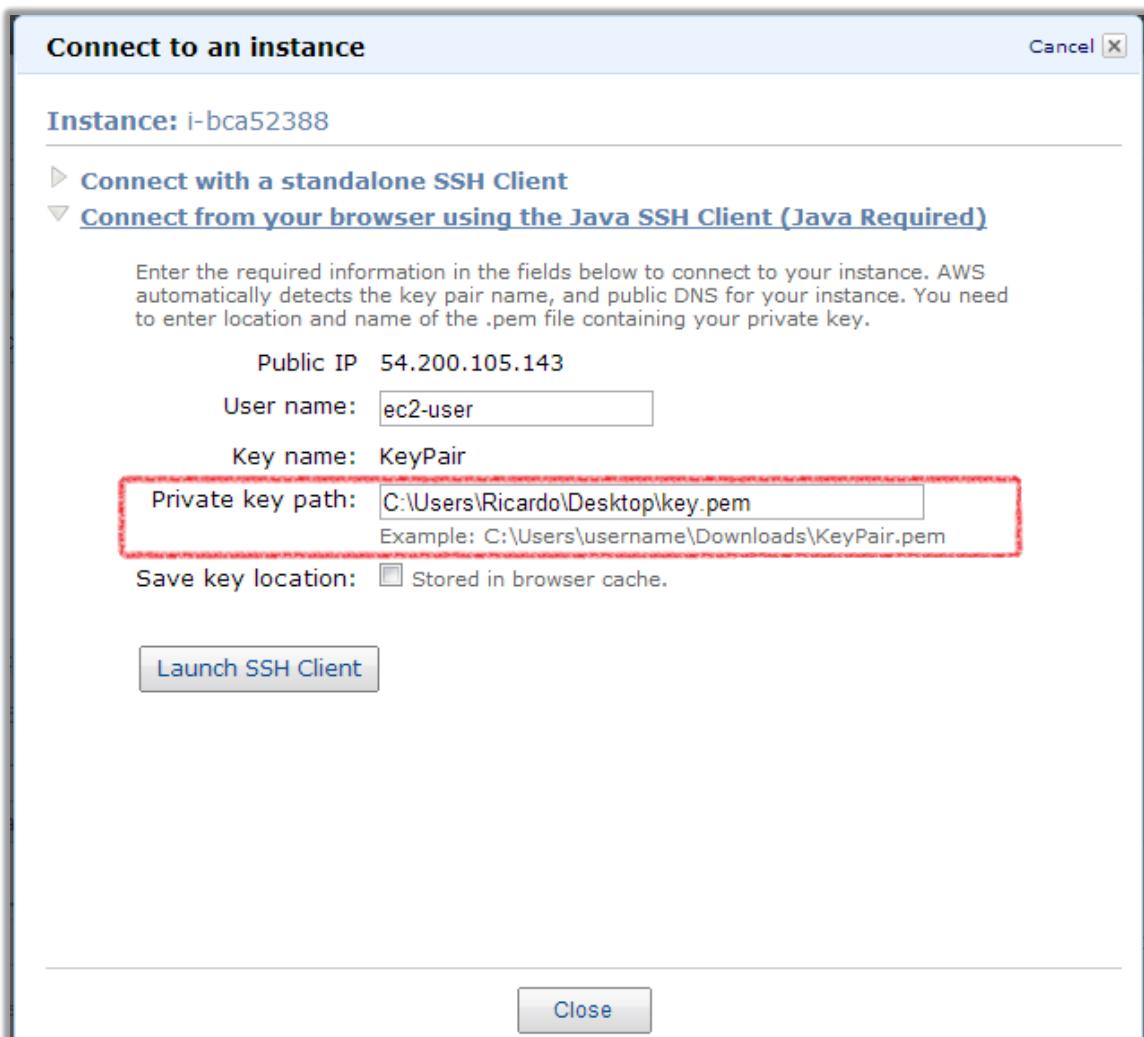


Fig. 39 – Tela principal de conexão a uma instância Linux. **Fonte:** Elaborado pelo autor.

2. Após isso ele instalará localmente o mindTerm, que é o cliente que dará acesso ao console da sua instância Linux. A Figura 40 demostra o mindTerm dando acesso a uma instância Linux.

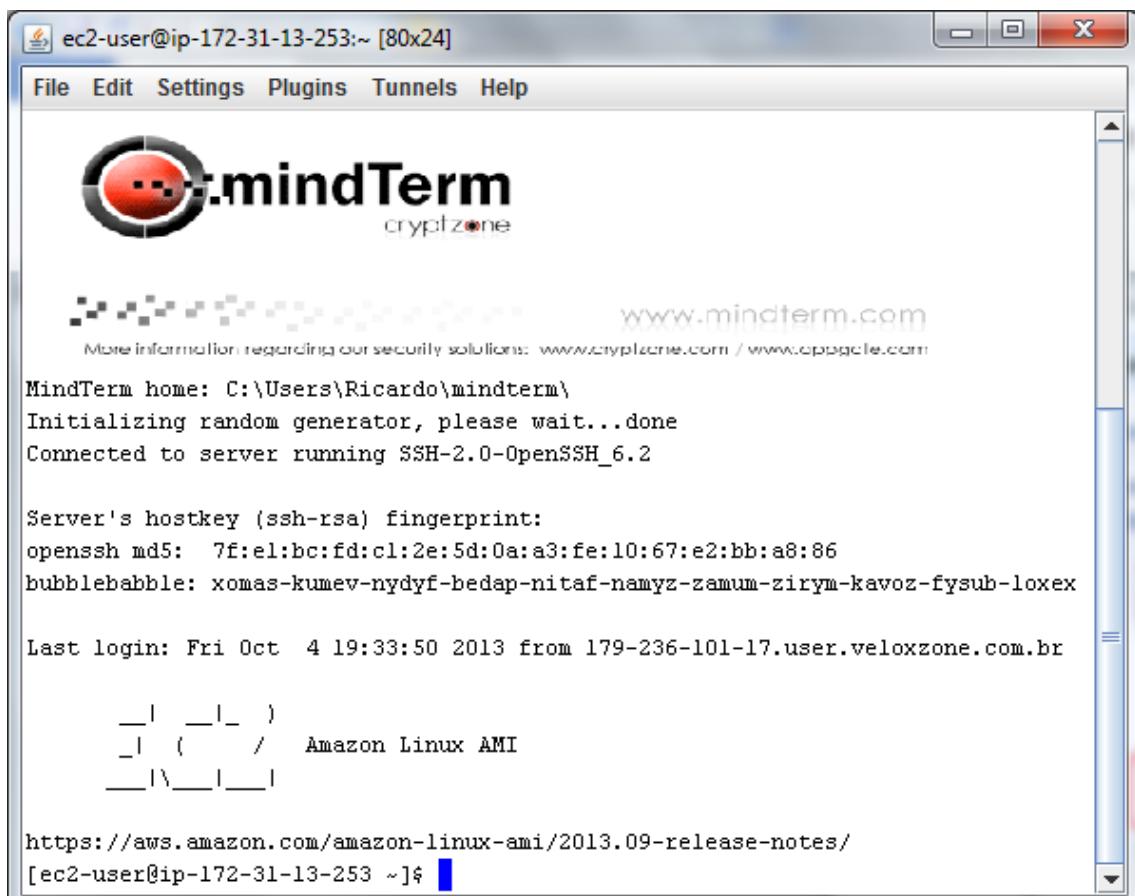


Fig. 40 – mindTerm dando acesso à instância Linux. **Fonte:** Elaborado pelo autor.

Depois de instalado, o mindTerm dará acesso e o controle da instância Linux.

APÊNDICE 7
CONEXÃO À UMA INSTÂNCIA WINDOWS

Conexão a instância Windows

Essa parte do tutorial assume que o usuário tenha seguido até o passo 10 e que a instância escolhida tenha sido com o sistema operacional Windows.

1. Após selecionar a opção ‘Connect’, na tela que se abriu selecione a opção ‘*Log in with your credentials*’ e clique em ‘*Retrieve Password*’ como demonstrado na Figura 41.



Fig. 41 – Tela de conexão a uma instância Windows. **Fonte:** Elaborado pelo autor.

2. Clique no botão ‘Escolher Arquivo’ e selecione sua *pair key* e em seguida clique em ‘*Decrypt Password*’ como ilustrado na Figura 42.

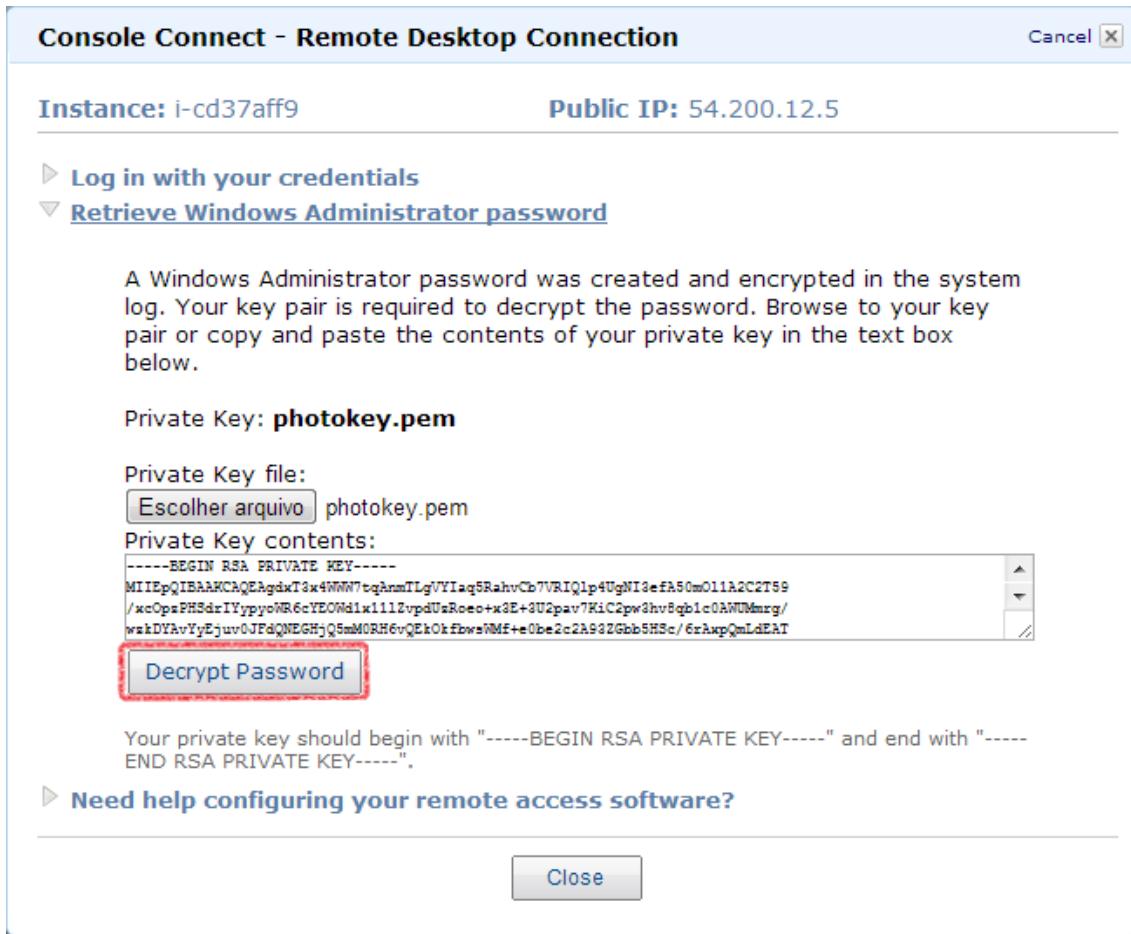


Fig. 42 – Decodificando a key pair para acesso. **Fonte:** Elaborado pelo autor.

3. A Figura 43 irá mostrar o IP da instância, o usuário e a senha de acesso, que foi decodificada de sua *pair key*.

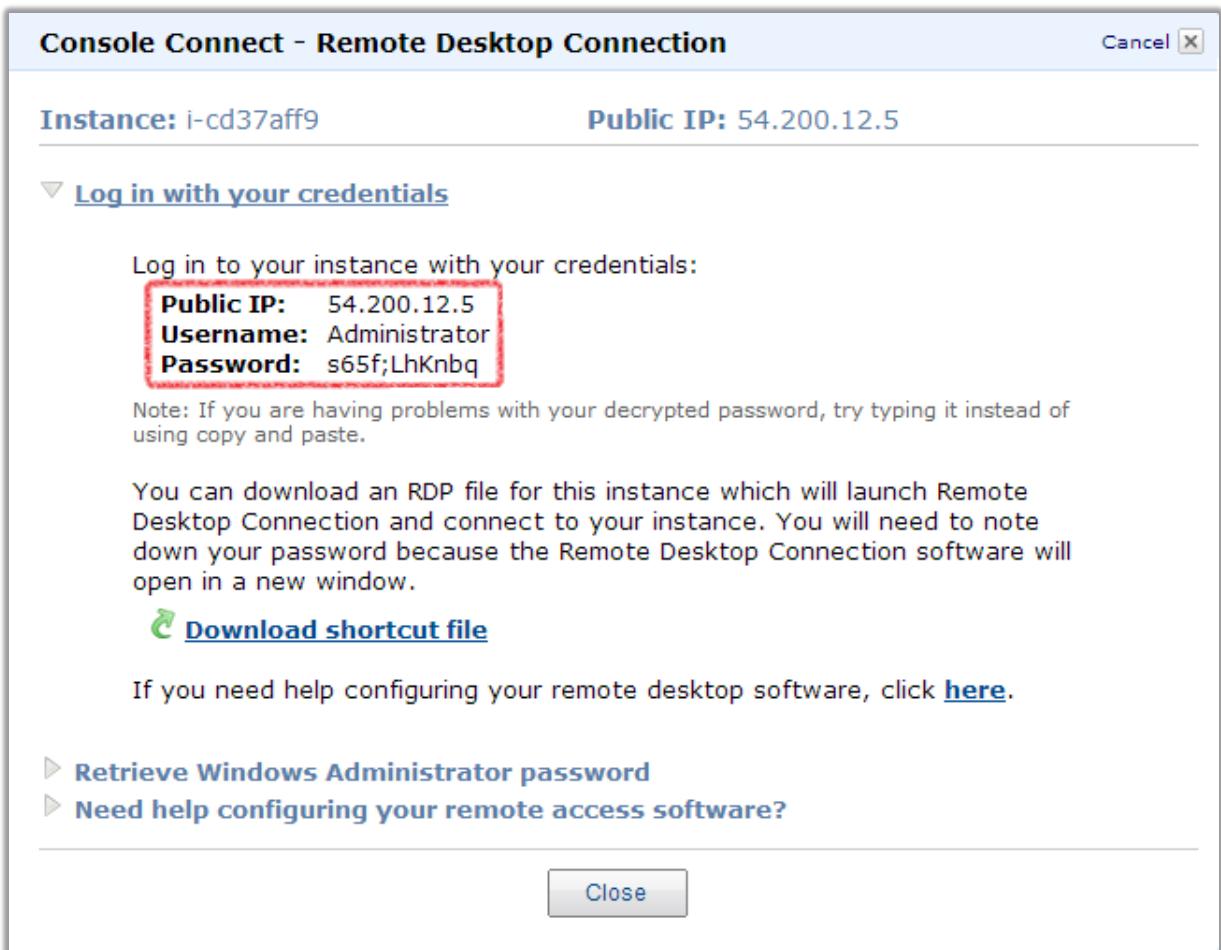


Fig. 43 – Dados para acesso à instância Windows. **Fonte:** Elaborado pelo autor.

4. Com esses dados, abrir o acesso remoto do Windows e digite o IP fornecido anteriormente e clique em ‘Conectar’ como ilustrado na Figura 44.

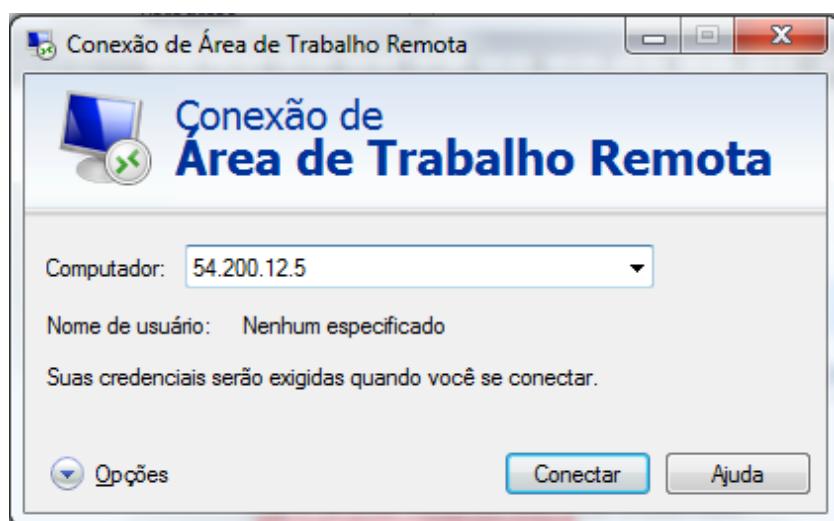


Fig. 44 – Acessando a instância no Elastic Cloud Compute.
Fonte: Elaborado pelo autor.

5. A seguir serão pedidos o usuário e senha como ilustrado na Figura 45. Colocar e clicar em ‘OK’.

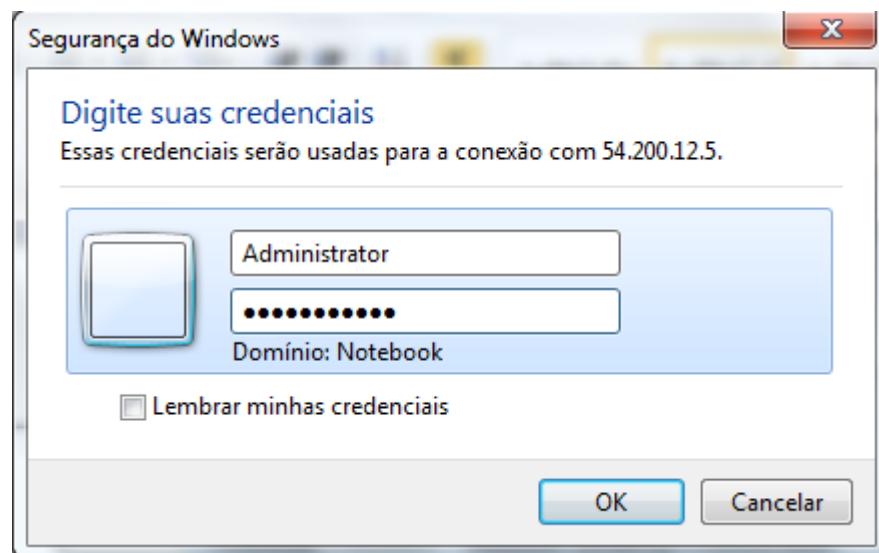


Fig. 45 – Colocando Usuário e senha para acesso. **Fonte:** Elaborado pelo autor.

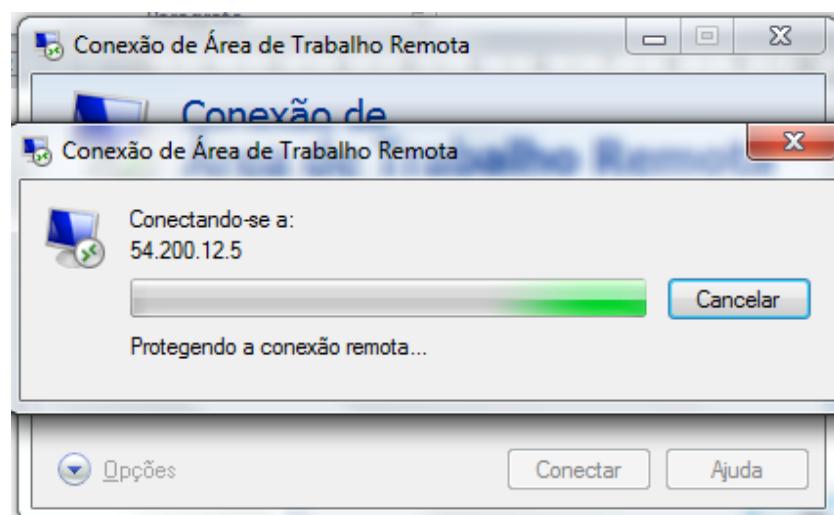


Fig. 46 – Conectando à instância. **Fonte:** Elaborado pelo autor.

Após seguir esses passos, será possível o acesso à instância criada, conforme a a Figura 47.

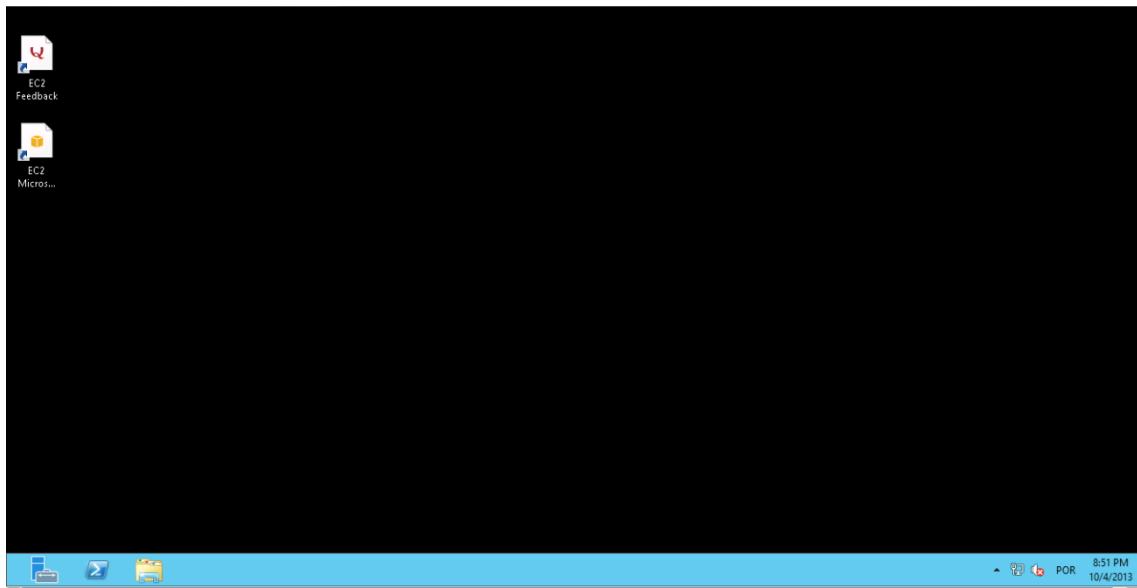


Fig. 47– Instância acessada pelo acesso remoto do Windows. **Fonte:** Elaborado pelo autor.

Com esse acesso é possível utilizar a instância como ambiente de desenvolvimento, servidor ou até mesmo rodar jogos.

**APÊNDICE 8
AMAZON RDS**

Amazon RDS

O Amazon RDS é o serviço que fornece uma instância de banco de dados na nuvem. Este tutorial mostra como criar uma instância.

1. Acessar o RDS no console do AWS e clicar em ‘*Launch a DB instance*’ como ilustrado na Figura 48.

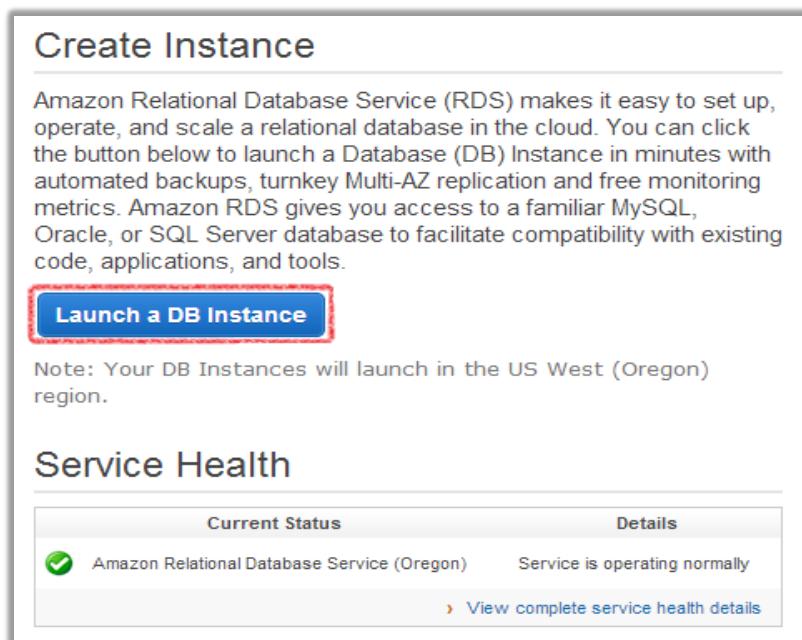


Fig. 48 – Tela inicial do Amazon RDS. **Fonte:** Elaborado pelo autor.

2. Na tela a seguir ilustrado na Figura 49 escolher o tipo da instância (SQLServer, MySql ou Oracle). Após isso clicar em ‘*Select*’.

Engine Selection

To get started, choose the DB Instance details below and click Continue

	mysql MySQL Community Edition	Select ▶
	oracle-se1 Oracle Database Standard Edition One	Select ▶
	oracle-se Oracle Database Standard Edition	Select ▶
	oracle-ee Oracle Database Enterprise Edition	Select ▶
	sqlserver-ex Microsoft SQL Server Express Edition	Select ▶
	<i>Note that SQL Server Express Edition limits the storage of per database to a maximum of 10GB. Refer to this link for more details.</i>	
	sqlserver-web Microsoft SQL Server Web Edition	Select ▶
	<i>Note that in accordance with Microsoft's licensing policies, SQL Server Web Edition can only be used to support public and internet accessible Web pages, Websites, Web applications and Web services. Refer to the AWS Service Terms for more details.</i>	

Fig. 49 – Opções de bancos de dados. **Fonte:** Elaborado pelo autor.

3. Em seguida o provedor irá perguntar se deseja criar uma instância para fins comerciais, se a escolha for sim, será oferecida a utilização do serviço Multi-AZ e Provisioned IOPS. A Figura 50 demonstra essa tela.

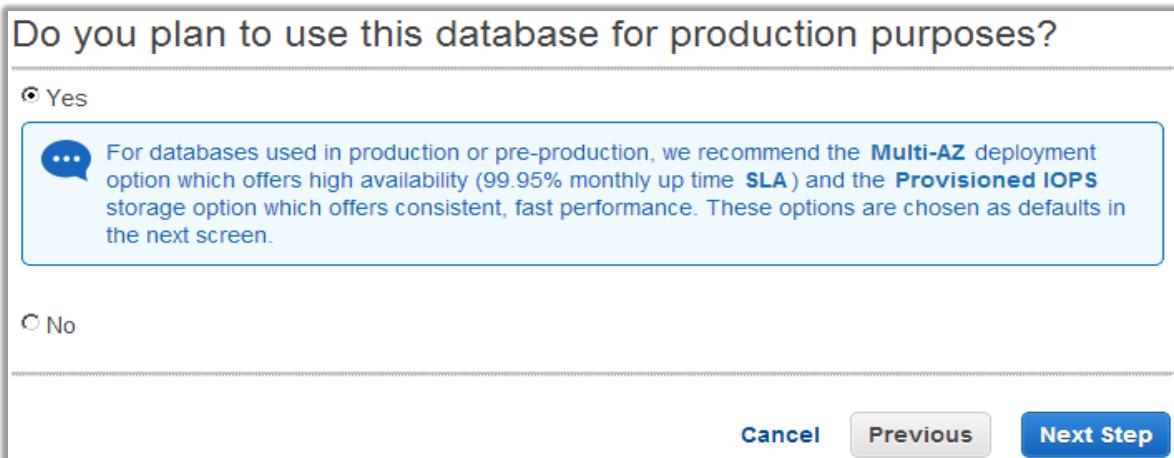


Fig. 50 – Configuração do banco de dados . **Fonte:** Elaborado pelo autor.

4. Na Figura 51 serão pedidas as configurações da instância, como o tipo de licença, a versão da *engine*, o tipo da instância, tamanho, usuário e senha.

DB Instance Details

To get started, choose a DB engine below and click Continue

DB Engine:	mysql
License Model:	General Public License
DB Engine Version:	5.6.13
DB Instance Class:	db.m1.xlarge
Multi-AZ Deployment:	Yes
Auto Minor Version Upgrade:	<input checked="" type="radio"/> Yes <input type="radio"/> No

Provide the details for your RDS Database Instance.

Allocated Storage: [*]	100	GB (Minimum: 100 GB, Maximum: 3072 GB)
Use Provisioned IOPS:	<input checked="" type="checkbox"/> Use m1.large or larger instances for best results.	
Provisioned IOPS:	1000	RDS MySQL supports IOPS / GB ratios between 3 and 10

For a workload with 50% writes and 50% reads running on a cr1.8xlarge instance, you can realize up to 20,000 IOPS. However, by provisioning more than this limit, you may be able to achieve lower latency and higher throughput. Your actual realized IOPS may vary from the amount you provisioned based on your database workload and instance type. Refer to the **Factors That Affect Realized IOPS section to learn more.**

DB Instance Identifier: [*]	mydbinstance	(e.g. mydbinstance)
Master Username: [*]	awsuser	(e.g. awsuser)
Master Password: [*]	mypassword	(e.g. mypassword)

Cancel **Previous** **Next Step**

Fig. 51 – Detalhes da instância. **Fonte:** Elaborado pelo autor.

5. Nas configurações adicionais, serão pedidos dados como nome da instância, grupo de segurança, grupo da VPC e zona de criação como demonstrado na Figura 52.

Additional Config

Provide the optional additional configuration details below.

Database Name: (e.g. mydb)

Note: if no database name is specified then no initial MySQL database will be created on the DB Instance.

Database Port: 3306

Choose a VPC:

DB Subnet Group:

Publicly Accessible: Yes No

Availability Zone:

Option Group:

If you have custom DB Parameter Groups or DB Security Groups you would like to associate with this DB Instance, select them below, otherwise proceed with default settings.

Parameter Group:

VPC Security Group(s):
quick-start-1 (VPC)
default (VPC)

Cancel **Previous** **Next Step**

Fig. 52 – Configurações adicionais do Amazon RDS. Fonte: Elaborado pelo autor.

6. A Figura 53 demonstra a configuração do backup automático.

Management Options

Enabled Automatic Backups: Yes No

The number of days for which automated backups are retained.

Please note that automated backups are currently supported for InnoDB storage engine only . If you are using MyISAM, refer to details [here](#) .

Backup Retention Period: days

The daily time range during which automated backups are created if automated backups are enabled

Backup Window: Select Window No Preference

The weekly time range (in UTC) during which system maintenance can occur.

Maintenance Window: Select Window No Preference

Cancel

Previous

Next Step

Fig. 53 – Configurando o backup automático. **Fonte:** Elaborado pelo autor.

7. Por fim serão exibidas todas as configurações selecionadas para a criação da instância como ilustrado na Figura 54.

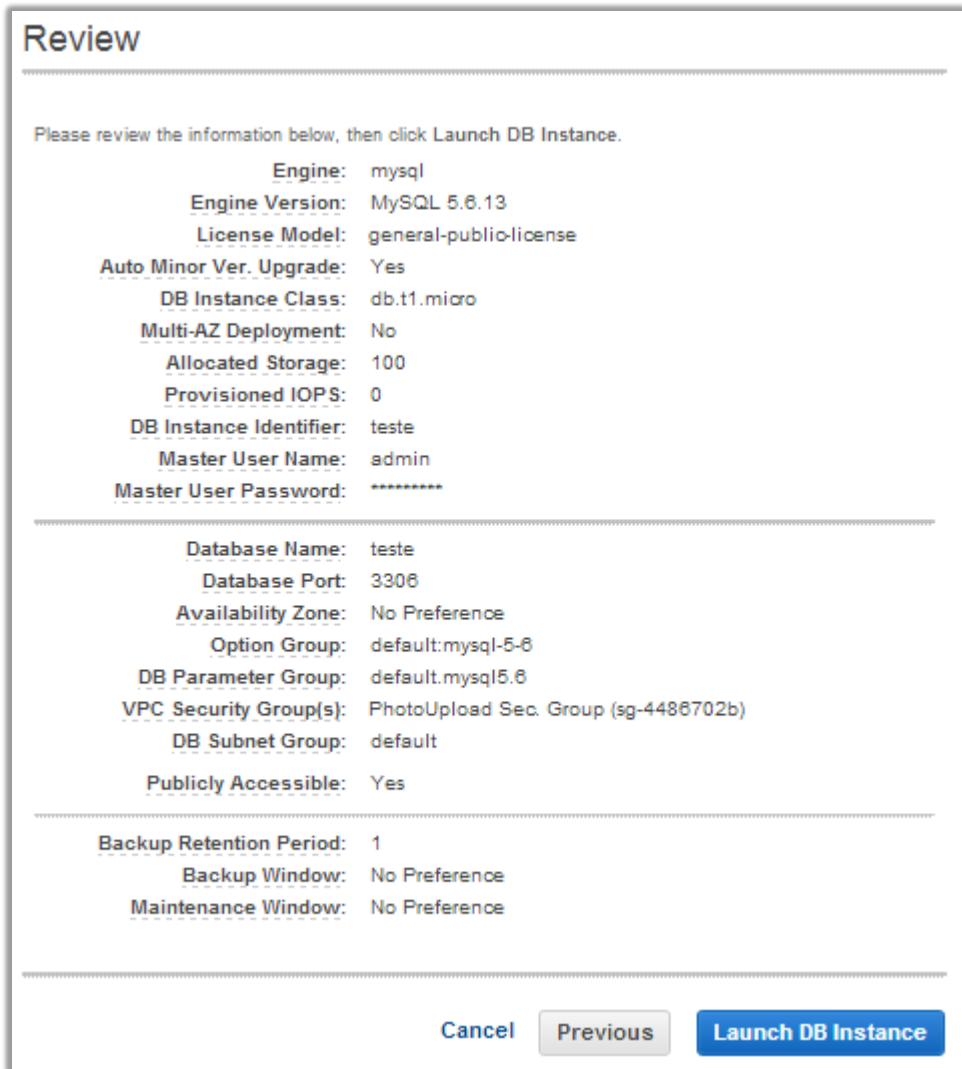


Fig. 54 – Resumo das configurações escolhidas. **Fonte:** Elaborado pelo autor.

Com os passos anteriores concluídos, aguardar alguns minutos até que ela seja criada nos servidores da AWS. A Figura 55 demonstra o RDS sendo criado e a Figura 55 ilustra o RDS Criado.

Launch DB Instance	Show Monitoring	Instance Actions							
Filter:	All Instances	Search DB Instances...	X	Viewing 1 of 1 DB Instances					
	DB Instance Identifier	VPC ID	Multi-AZ	Class	Status	Storage	Security Groups	Engine	Zone
	teste	vpc-8ca861e7	No	db.t1.micro	creating	100 GB	PhotoUpload Sec. Group (active)	mysql	us-west-2a

Fig. 55 – Instância sendo criada no Amazon RDS. **Fonte:** Elaborado pelo autor.

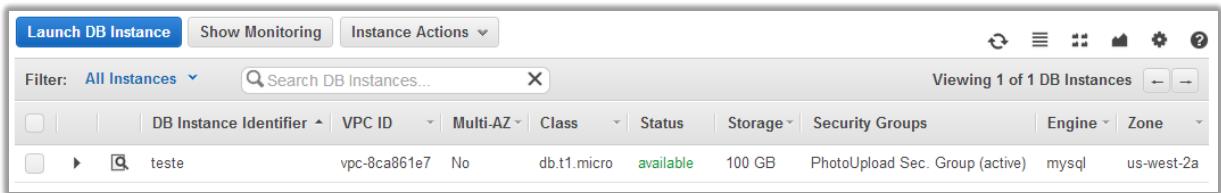


Fig. 56– Instância criada no Amazon RDS. **Fonte:** Elaborado pelo autor.

Clicando sobre a instância é possível ver todas as configurações dela como demonstrado na Figura 57.

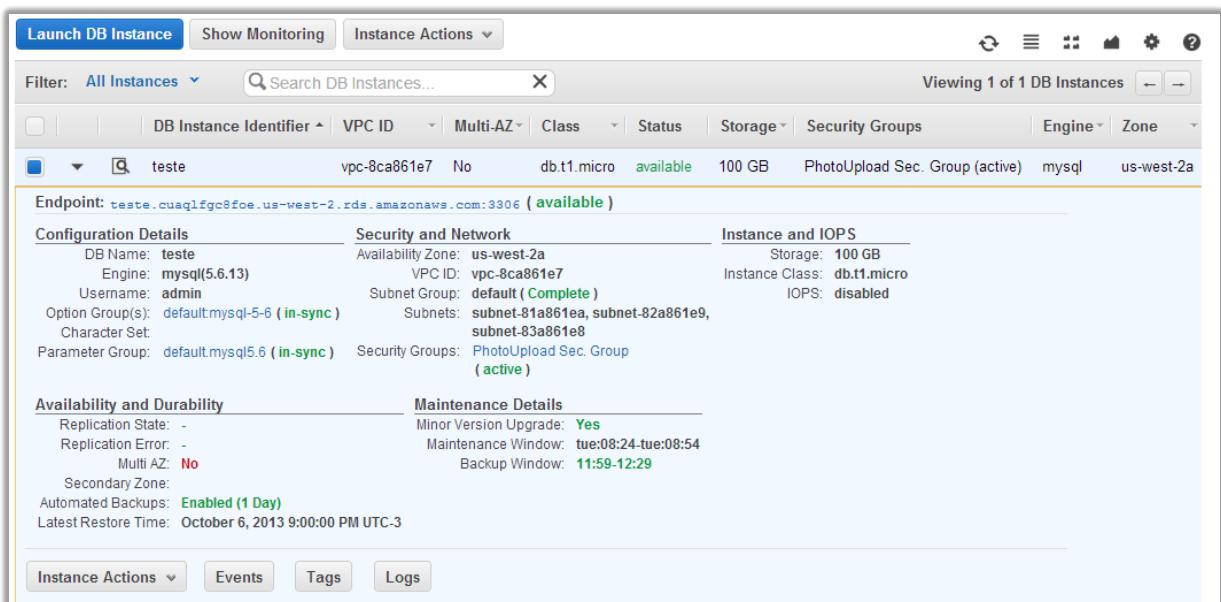


Fig. 57 – Propriedades da instância no Amazon RDS. **Fonte:** Elaborado pelo autor.

APÊNDICE 9

AMAZON S3

Amazon S3

O Amazon S3 é o serviço que oferece o armazenamento de diversos dados com variados formatos, como vídeos, músicas e etc. É nele que o Amazon Elastic Beanstalk armazena a aplicação e os arquivos gerados por ela.

1. Para a criação de um *bucket* no S3, primeiro acessar o serviço e em seguida clicar em ‘Create Bucket’ como demonstrado na Figura 58.

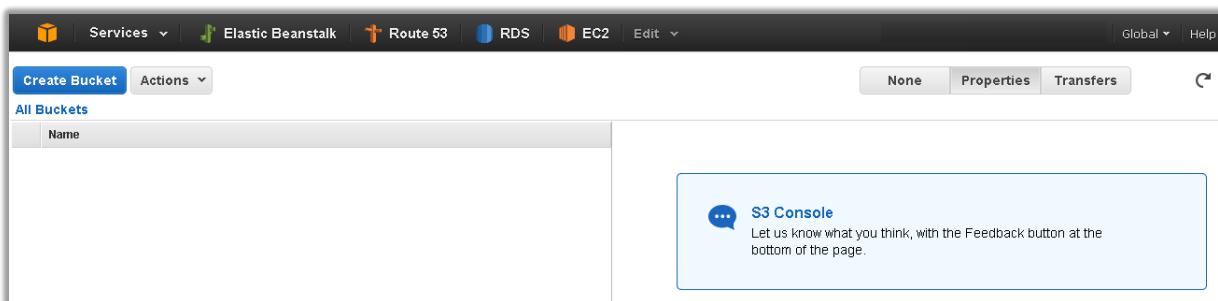


Fig. 58 – Tela inicial do Amazon S3. **Fonte:** Elaborado pelo autor.

2. Na tela seguinte demonstrada na Figura 59, serão pedidos o nome e a região do *bucket*, após colocar essas informações, clicar em ‘Create’.

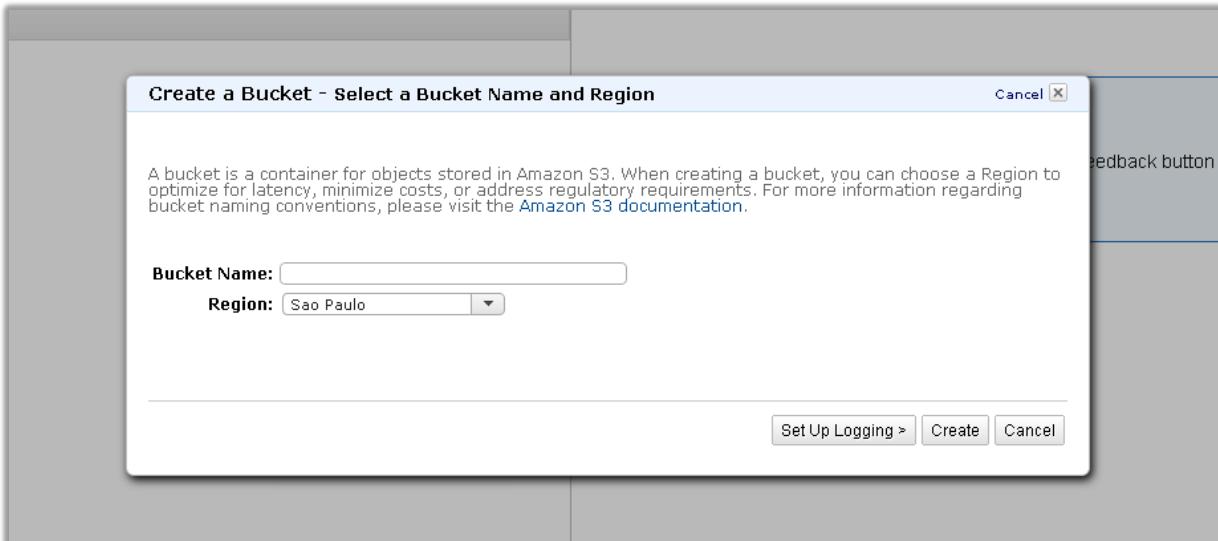


Fig. 59 – Escolhendo nome e região. **Fonte:** Elaborado pelo autor.

3. Por fim o *bucket* foi criado e será apresentado na tela como ilustrado na Figura 60. É possível acessá-lo, criar pastas e manipular arquivos clicando duas vezes em cima do nome dele.

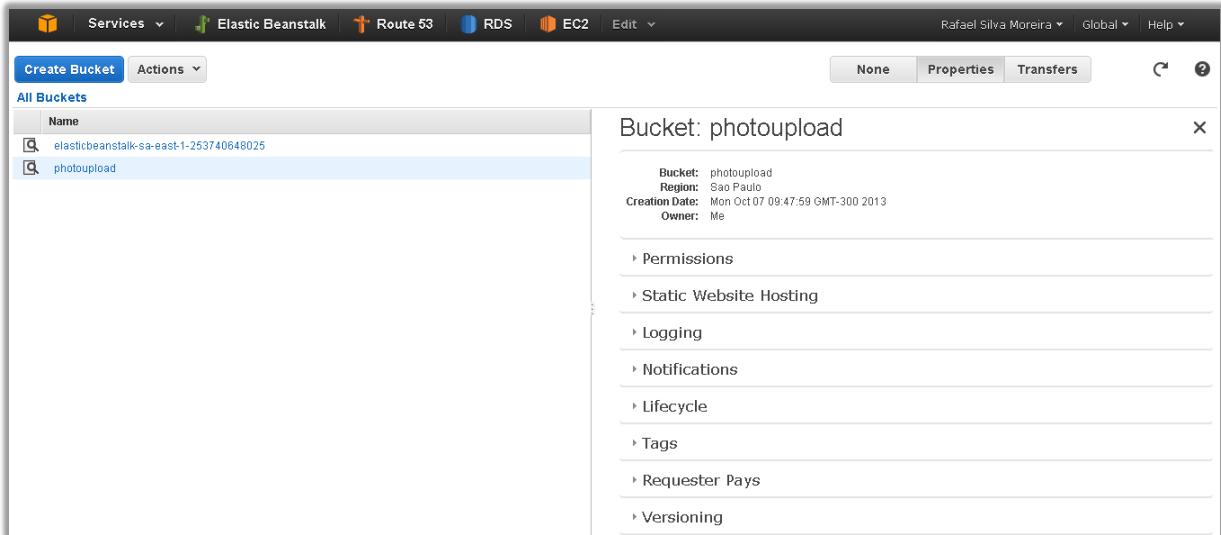


Fig. 60 – Relação de buckets criados no Amazon S3. **Fonte:** Elaborado pelo autor.

Na tela principal do Amazon S3 após selecionar o *bucket* criado, no lado direito do console será mostrado um menu de configurações, permitindo alterar arquivos de *logs*, permissões de acesso, ciclos de vida dentre outras.