UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLOGIA
CURSO DE CIÊNCIA DA COMPUTAÇÃO

THIAGO AUGUSTO LOPES SILVA

UTILIZAÇÃO DE COMPUTAÇÃO EM NUVEM PARA O DESENVOLVIMENTO E IMPLANTAÇÃO DE SOLUÇÕES WEB

São Luís
2016

THIAGO AUGUSTO LOPES SILVA

UTILIZAÇÃO DE COMPUTAÇÃO EM NUVEM PARA O DESENVOLVIMENTO E IMPLANTAÇÃO DE SOLUÇÕES WEB

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Orientador: Prof. Dr. Mário Antonio Meireles Teixeira

São Luís
2016

Silva, Thiago Augusto Lopes.
Utilização de computação em nuvem para o desenvolvimento e implantação de soluções WEB/ Thiago Augusto Lopes Silva.– São Luís, 2016.

76 f

Monografia (Graduação) – Curso de Ciência da Computação, Universidade Federal do Maranhão, 2016.

Orientador: Prof. Mário Antonio Meireles Texeira

1. LAMP. 2.Sistemas web. 3. Hardware. 4. Software. 5. Computação em Nuvem. I.Teixeira, Mário Antonio Meireles. II. Título

THIAGO AUGUSTO LOPES SILVA

UTILIZAÇÃO DE COMPUTAÇÃO EM NUVEM PARA O DESENVOLVIMENTO E IMPLANTAÇÃO DE SOLUÇÕES WEB

Monografia apresentada ao curso de Ciência da Computação da Universidade Federal do Maranhão, como parte dos requisitos necessários para obtenção do grau de Bacharel em Ciência da Computação.

Aprovada em: 20/05/16

BANCA EXAMINADORA

Prof. Dr. Mário Antonio Meireles Teixeira (Orientador)
Doutor em Ciência da Computação
Universidade Federal do Maranhão

Prof. Dr. Samyr Beliche Vale
Doutor em Ciência da Computação
Universidade Federal do Maranhão

Profa. Ms. Maria Auxiliadora Freire
Mestre em Ciência da Engenharia
Universidade Federal do Maranhão

AGRADECIMENTOS

Agradeço, acima de tudo, a Deus, por ter me dado inteligência e me guiado ao longo do desafiante trabalho. Gostaria de agradecer também ao professor doutor Mário Meireles Teixeira, pela paciência e sensatez, pela dedicação e pelo conhecimento repassado a mim, e agradecer todos os professores da UFMA os quais eu fui aluno, em especial a professora Inez Cavalcanti Dantas, por ter me oferecido a incrível experiência de ensinar os idosos a utilizar o computador.

Em segundo, agradeço à minha família e namorada, que me apoiaram até o fim desta jornada, e que em momentos de fraqueza souberam me levantar para continuar a luta diária.

Agradeço aos amigos do curso, pois cresceram junto comigo nesta jornada e me fizeram aprender que o trabalho em equipe é essencial, além de entender as dificuldades do curso e estudar para ultrapassar as mesmas. Aos amigos e colegas não estudantes da universidade, pois me alegraram em momentos difíceis.

Agradeço, por fim, a qualquer pessoa que diretamente ou indiretamente me fizeram continuar a jornada pela universidade e a realização do trabalho.

“O cientista não é o homem que fornece as verdadeiras respostas; é quem faz as verdadeiras perguntas”. (Claude Lévi-Strauss)

RESUMO

O desenvolvimento tradicional de sistemas web, conhecido pelo acrônimo LAMP contém um conjunto de ferramentas que, juntas, desenvolvem soluções para vários propósitos, seja em intranets ou na internet. O principal problema deste modelo de desenvolvimento é a escalabilidade, pois um usuário não consegue alterar os recursos do servidor sem que haja mudança ou quebra do contrato. Por causa disso algumas empresas começaram a proporcionar serviços, os quais os usuários poderão selecionar o hardware e software necessário para que a aplicação funcione, pagando apenas o que usou. A este novo modelo de serviço chama-se de Computação em Nuvem. O objetivo do trabalho é desmistificar o uso deste modelo e criar um *webapp* a partir de serviços em nuvem da empresa Amazon.

Palavras-chave: LAMP. Sistemas web. Hardware. Software. Computação em Nuvem.

ABSTRACT

The traditional development of web systems, known as LAMP contains a set of tools that, together, develops solution to various purposes, in intranets or internet. The main issue of this model of development is the scalability, because the user can't alter the capacity without any changes or breach of contract. Because of that, some companies started to provide scalable services, wich users can select hardware and software needed to make the application works, paying just what have been used. To this new model of service is called Cloud Computing. The goal of the work is to demystify the use of this model, and create an webapp starting from services in cloud of the Amazon business.

Keywords: LAMP, web systems, hardware, software, Cloud Computing.

LISTA DE FIGURAS

Figura 1-XAMPP para as diversas plataformas (Apache Friends, 2016).....17
Figura 2-mecanismo de seleção de licença de software livre (WIKIPÉDIA, 2015).....18
Figura 3 - Logo Open Source Initiative (Google, 2016). 20
Figura 4 - Logo do Ubuntu (Google, 2016). 23
Figura 5 - Logo do apache http server (Google, 2016). 23
Figura 6 - Logo do MySQL (Google, 2016). 30
Figura 7 - Gráfico da interação entre front-end e back-end (3NY TECHNOLOGY, 2016)....31
Figura 8 - Ilustração do acesso aos serviços da Amazon. 43
Figura 9 - Opções de usuário no site da Amazon (AWS, 2016). 48
Figura 10 - Sistema de gerenciamento de custos da Amazon (AWS, 2016).....48
Figura 11 - Lista de serviços AWS (AWS, 2016). 50
Figura 12 - Seções e subseções Amazon EC2 (AWS, 2016). 51
Figura 13 – Primeiro passo: Selecionar uma imagem da máquina Amazon (AWS, 2016). 52
Figura 14 – AMI Ubuntu Server 14.04 (AWS, 2016).....52
Figura 15 – Tipos de instância de EC2 (AWS, 2016). 53
Figura 16 – Estado da instância EC2 (AWS, 2016). 53
Figura 17 – Possíveis estados da instância (AWS, 2016). 54
Figura 18 – Logo Cygwin (Wikipedia, 2016). 54
Figura 19 – Conexão com a Instância EC2 – Terminal do Cygwin.....56
Figura 20 – Aplicativo web acessado através do Public DNS 57
Figura 21 – Logo Filezilla (Google, 2016).....58
Figura 22 – Gerenciador de sites com os dados do EC2. 58
Figura 23 – Formulário de criação do bucket (AWS, 2016). 60
Figura 24 – Arquivos armazenados dentro do bucket (AWS, 2016).....60
Figura 25 – Código PHP para a instanciação do cliente S3 (AWS Documents, 2016).....61
Figura 26 – Código para instanciação de um cliente S3 utilizando credenciais codificadas. ..62
Figura 27 – Procedimento getIterator.....63
Figura 28 – Função foreach.....63
Figura 29 – Função echo. 64
Figura 30 – Código PHP que move o arquivo para uma pasta temporária. 64
Figura 31 – Método para inserção de arquivos no bucket do S3.....65

Figura 32 – Inclusão do pacote de exceções do S3. 65
Figura 33 – Bancos de dados suportados pelo Amazon RDS (AWS, 2016). 67
Figura 34 – Comando “show databases;” 68
Figura 35 – Variáveis de conexão PDO com o Amazon RDS. 70

LISTA DE ABREVIATURAS E SIGLAS

ACL	Access Control List
AMI	Amazon Machine Image
ANSI	American National Standards Institute
API	Application Programming Interface
ASF	Apache Software Foundation
AWS	Amazon Web Services
AZ	Availability Zone
BSD	Berkeley Software Distribution
CPU	Central Processor Unity
CSS	Cascading Style Sheet
DB	Database
DML	Data Manipulation Language
DNS	Domain Name System
EC2	Elastic Compute Cloud
EUA	Estados Unidos da América
FISMA	The Federal Information Security Management Act
FTP	File Transfer Protocol
GaaS	Gaming as a Service
GB	GygaByte
GNU	General
HTML	Hypertext Markup Language
HTTP	Hypertext Transfer Protocol
HTTPS	Hypertext Transfer Protocol Secure
IaaS	Infrastructure as a Service
IAM	Identity and Access Management
IBM	International Business MAchines
ID	Identifier
IDE	Integrated Development Environment
I/O	Input/Output
IP	Internet Protocol
ISO	International Organization for Standardization

ITU	International Telecommunication Union
J2EE	Java 2 Platform Enterprise Edition
LAMP	Linux, Apache, MySQL ou MariaDB, PHP ou Python
LDD	Linguagem de Definição de Dados
LMD	Linguagem de Manipulação de Dados
MIT	Massachusetts Institute of Technology
MySQL	My Structured Query Language
NCSA	National Center for Supercomputing Applications
NoSQL	No Structured Query Language
ODBC	Open Database Connectivity
OS	Operating System
OSS	Open Source Software
PaaS	Platform as a Service
PCRE	Pearl Compatible Regular Expressions
PDO	PHP Data Objects
PHP	Hypertext Preprocessor
PyGTK	Python GIMP Toolkit
RDS	Relational Database Service
RHEL	Red Hat Enterprise Linux
S3	Simple Storage Service
SaaS	Software as a Service
SDK	Software Development Kit
SFTP	SSH File Transfer Protocol
SGBD	Sistema Gerenciador de Banco de Dados
SGBDOO	Sistema Gerenciador de Banco de Dados Orientado à Objetos
SLES	SUSE Linux Enterprise Server
SNS	Simple Notification Service
SQL	Structured Query Language
SQS	Simple Queue Service
SSD	Solid State Drive
SSH	Secure Shell
SSL	Secure Sockets Layer
temp	Temporary
TI	Teconologia da Informação

VPC	Virtual Private Cloud
WAMP	Windows, Apache, Mysql ou MariaDB, PHP ou Python
XML	eXtensible Markup Language

SUMÁRIO

1 INTRODUÇÃO 15
1.1 OBJETIVOS 16
1.2 ORGANIZAÇÃO DO TRABALHO 16
2 DESENVOLVIMENTO WEB TRADICIONAL (LAMP) 17
2.1 SOFTWARE LIVRE 17
2.2 CÓDIGO ABERTO 19
2.3 SISTEMA OPERACIONAL LINUX 21
2.4 APACHE HTTP SERVER 23
2.4.1 Apache Software Foundation (ASF) 24
2.5 BANCO DE DADOS 25
2.5.1 Sistemas Gerenciadores de Banco de Dados (SGBD) 25
2.5.2 Structured Query Language 27
2.5.3 MySQL 29
2.5.4 MariaDB 30
2.6 LINGUAGEM DE PROGRAMAÇÃO PHP E PYTHON 30
2.6.1 Frontend vs Backend 31
2.6.2 Por que PHP? 32
2.6.3 Por que Python? 33
2.7 NOVO MODELO DE SERVIÇO 33
3 COMPUTAÇÃO EM NUVEM 35
3.1 SOFTWARE AS A SERVICE (SAAS) 38
3.2 INFRASTRUCTURE AS A SERVICE (IAAS) 38
3.3 PLATFORM AS A SERVICE (PAAS) 39
3.4 GAMING AS A SERVICE (GAAS) 39
3.5 COMMUNICATION AS A SERVICE (CAAS) 40
3.6 DATABASE AS A SERVICE (DBAAS) 40
4 AMAZON WEB SERVICES (AWS) 41
4.1 AMAZON ELASTIC COMPUTE CLOUD (EC2) 43
4.2 AMAZON SIMPLE STORAGE SERVICE (S3) 44
4.3 AMAZON RELATIONAL DATABASE SERVICE (RDS) 45
4.4 AMAZON VPC 45

5 DESENVOLVIMENTO DA SOLUÇÃO WEB NA AWS 47
5.1 FUNCIONALIDADES DA SOLUÇÃO WEB 49
5.2 INSTÂNCIA EC2 50
5.2.1 Criação 50
5.2.2 Utilizando o Cygwin para conexão SSH 54
5.2.3 Utilizando o FileZilla para acessar os arquivos do servidor 57
5.3 ARMAZENAMENTO DE ARQUIVOS NÃO DINÂMICOS NO AMAZON S3 59
5.3.1 Criando um bucket 59
5.3.2 Instalando o Amazon SDK for PHP 60
5.3.3. Utilizando o SDK para acessar arquivos no S3 61
5.4 UTILIZANDO BANCO DE DADOS ATRAVÉS DO AMAZON RDS 66
5.4.1 Criando uma instância de banco de dados MySQL 66
5.4.2 Acessando a instância MySQL 68
5.4.3 Modificando o código para acessar o Amazon RDS 69
CONCLUSÃO 71
REFERÊNCIAS BIBLIOGRÁFICAS 72

1 INTRODUÇÃO

A web pode ser definida como um conjunto de computadores interligados entre si, compartilhando dados e arquivos em formas de páginas. No início, os arquivos eram conhecidos como páginas estáticas, quase não havia interação entre o usuário e a página web referenciada. Com o passar do tempo, ferramentas foram criadas para auxiliar no desenvolvimento de páginas web de forma dinâmica, com a utilização de bancos que armazenam dados e linguagens que possam acessar esses dados.

Algumas destas ferramentas ganharam maior notoriedade que outras, por serem gratuitas e de fácil aprendizado, além de contar com comunidades atuantes e expressivas, o que resulta num melhor suporte para os desenvolvedores web. A notoriedade fora tanta, que a comunidade passou a chamar a junção das mesmas de "LAMP Stack", pois juntas as mesmas são capazes de criar aplicativos web dinâmicos. Vale ressaltar que o termo "aplicação web" (*web apps*) é designado para sistemas de computação que foram projetados para que a sua utilização fosse feita através do navegador, utilizando o protocolo HTTP ou HTTPS. A máquina do usuário solicita uma página (arquivo) ao servidor que a hospeda e o mesmo formata a resposta de forma que o navegador consiga entender (em HTML, usualmente) e a envia ao usuário.

As empresas guardam muita informação de usuários e serviços que os mesmos utilizaram, e acabam acarretando em um armazenamento de grandes quantidades de dados, criando o que chamamos de *Big Data.* Além disso, para se fazer um aplicativo web deve-se fazer uma estimativa de acesso e utilização de dados ao longo do tempo de vida do software.

Esta estimativa nem sempre corresponde ao uso real da aplicação depois de algum tempo, e os desenvolvedores acabam tendo que negociar com o servidor de hospedagem para modificar os serviços contratados, porque não há escalabilidade no contrato do serviço. Esta modificação muitas vezes demora algum tempo para ser concretizada, pois o contrato deve ser revisto, ou até mesmo quebrado, e quem sofre com estes problemas são os próprios usuários, que precisam utilizar a aplicação, pois a mesma acaba se tornando lenta.

Como uma das soluções para este problema, surgiu um conjunto de serviços altamente escalável, que possa corresponder mais rapidamente às modificações das regras de negócio ou ao número de acessos diários, sem que haja contato direto entre o usuário e a empresa prestadora do serviço. A esta nova modalidade chama-se de computação em nuvem. Esta modalidade permite o desenvolvimento de aplicações cujos servidores podem modificar

sua capacidade computacional, acarretando em diminuição de custos, além de poder ser acessado em qualquer parte do mundo, desde que o dispositivo possa se conectar a internet.

1.1 OBJETIVOS

Este trabalho tem como objetivo geral desmistificar a computação em nuvem por meio do desenvolvimento de uma solução de aplicação web utilizando um conjunto de serviços em nuvem oferecidos pela Amazon, conhecido como *Amazon Web Services* (AWS).

Os objetivos específicos são:

a) Definir a diferença entre computação em nuvem e desenvolvimento tradicional;
b) Analisar os serviços oferecidos pela Amazon Web Services;
c) Desenvolver uma solução web completa utilizando serviços da nuvem AWS, da Amazon, dentre eles o EC2, S3 e RDS.

1.2 ORGANIZAÇÃO DO TRABALHO

O capítulo 2 explica o conjunto de ferramentas utilizado para o desenvolvimento tradicional de aplicações web (LAMP), de software livre e código aberto.

O capítulo 3 apresenta o conceito de computação em nuvem e qual a diferença entre esta modalidade de desenvolvimento e o desenvolvimento web tradicional, além de explicar as diferentes modalidades de serviços em nuvem.

O capítulo 4 apresenta os serviços da nuvem da Amazon, conhecidos como *Amazon Web Services* (AWS).

O capítulo 5 apresenta as especificações da aplicação escolhida nesta monografia e as etapas para o desenvolvimento do mesmo utilizando serviços da AWS (EC2,S3 e RDS).

2 DESENVOLVIMENTO WEB TRADICIONAL (LAMP)

O termo LAMP designa uma junção de softwares livres e de código aberto. Cada letra do termo é referente às primeiras letras de Linux, Apache, MySQL (atualmente MariaDB), e PHP (ou Python), que serão explicados adiante.

Este pacote de softwares se tornou altamente popular por ser de código aberto e livre de custo, o que garante uma fácil adaptação e, quando usados juntos, suportam servidores de aplicações web. Além disso, são de fácil instalação, incluindo até instaladores mais genéricos, como o XAMPP, bastando alguns cliques para que o pacote PHP/MySQL e Apache estejam instalados. Estes instaladores são suportados nos sistemas operacionais mais utilizados, como se pode ver na figura abaixo.

Figura 1-XAMPP para as diversas plataformas (Apache Friends, 2016).

Desde a sua criação, o modelo LAMP tem sido adaptado para outros componentes, sendo estes, também, livres e de código aberto. O WAMP é um exemplo desses componentes, que é a instanciação do "AMP" para a família de sistemas operacionais Windows (MARGARET ROUSE, 2008).

2.1 SOFTWARE LIVRE

Podemos dizer que um programa é classificado como software livre (*free software*) quando o mesmo permite adaptações ou modificações sem que haja a necessidade de contato, pagamento ou pedir permissão do proprietário. O termo foi formalizado pela primeira vez por Richard Stallman, no começo do projeto GNU.

Segundo a comunidade de software livre do Brasil (2009), para que o software seja considerado livre, o mesmo deve obedecer as seguintes características:

- O usuário poderá executar o programa para qualquer propósito;
- O usuário poderá estudar e adaptar o programa para as suas necessidades;
- O usuário poderá redistribuir o código de modo a ajudar o próximo;
- O usuário poderá aperfeiçoar o software, e liberar estes aperfeiçoamentos para que toda a comunidade possa se beneficiar do mesmo.

Percebe-se, através destas características, que os usuários deverão ter acesso indiscriminado ao código do programa e suas modificações (quando publicados para a comunidade). Quando uma modificação é publicada, o usuário não deve ser obrigado a avisar alguém especial. Além disso, podem-se redistribuir cópias exatas ou modificações da mesma, de graça ou cobrando taxa pela distribuição, para qualquer indivíduo ou entidade.

Vale a pena ressaltar que o software livre é diferente de não comercial. Um programa não comercial não pode ser distribuído para fins comerciais, já o outro deve estar disponível para uso, desenvolvimento e distribuição para fins inclusive comerciais. (GNU, 2013)

Apesar destas quatro características, softwares livres diferentes podem conter diferentes tipos de licença, obedecendo a determinados propósitos, de acordo com a figura 2.

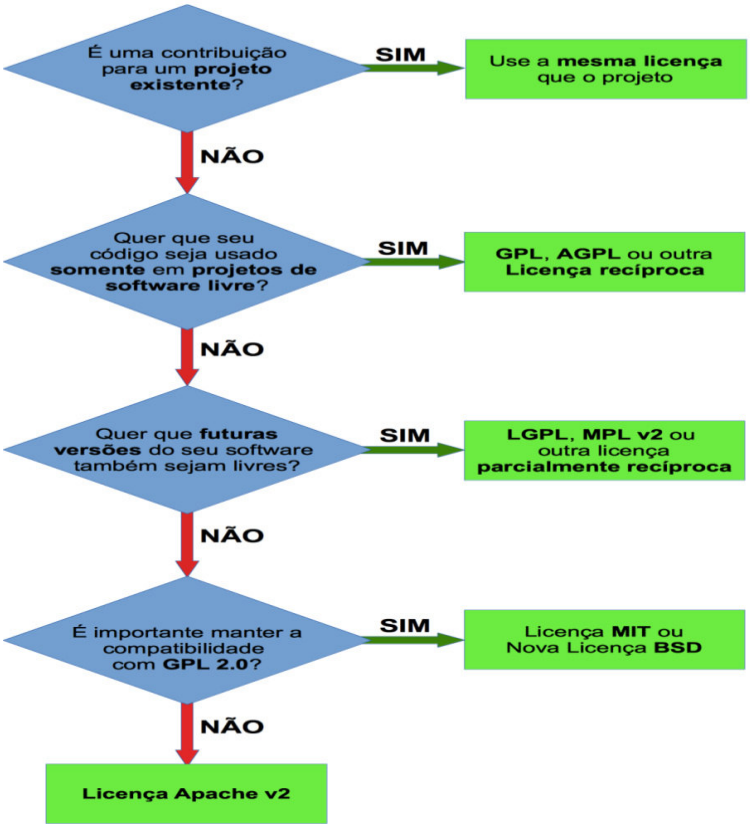


Figura 2 - Mecanismo de seleção de licença de software livre (WIKIPÉDIA, 2015).

Segundo Vanessa Sabino e Fabio Kon (2009), destacam-se três tipos de licenças, que são:

- Licenças permissivas: também são conhecidas como "licenças acadêmicas", pois a origem do BSD foi originária da Universidade da Califórnia, e também do MIT (Instituto de Tecnologia de Massachusetts). Como o nome diz, este tipo de licença permite que os usuários possam até mesmo modificar o código e fechá-lo para exploração comercial. Um exemplo de licença que usa essas características é a licença Apache;
- Licenças recíprocas totais: delimita que todo software livre deve continuar livre, mesmo que sofra alterações. Neste caso, o software obedece a filosofia do *copyleft*;
- Licenças recíprocas parciais: "meio termo" entre as licenças permissivas e recíprocas totais. Um software, a princípio, deve ser compartilhado sem restrições, caso haja modificações. Mas quando esta modificação se torna componente para outro software, o projeto não precisa ser disponibilizado com a mesma licença;

Assim, o usuário pode adicionar certas restrições, mesmo que o software seja dito "livre".

2.2 CÓDIGO ABERTO

Enquanto o Software Livre garante que o programa possa ser distribuído de acordo com a sua licença, para fins comerciais ou não, o código aberto (*open source*) é uma metodologia de desenvolvimento que garante que todos possam acessar o código fonte do programa, além de estudar, modificar e distribuir o software de graça para qualquer um e para qualquer finalidade. A grande diferença entre ambos é que a metodologia de código aberto não aceita determinadas restrições que a de software livre defende, por considerá-las restritivas demais. Diz-se que "*Open Source* é um modelo de desenvolvimento, e o software livre é um movimento social" (OPEN SOURCE INITATIVE, 2014). Vale ressaltar que quase todo software livre é Open Source, e vice-versa.

Richard Stallman, atualmente conhecido como o pai do software livre, lançou tal movimento em 1983. O termo se torna ambíguo para quem o lê e tenta entendê-lo de forma literal. Assim, um grupo de indivíduos defendeu, em 1988, que o termo deveria ser trocado

por software open-source (OSS). Entendiam que, além da ambiguidade, o termo seria mais bem aceito pelo mundo corporativo.

O *Open Source Initiative*, cuja logo está ilustrada na figura 3, lançou um documento que define o código aberto, além de determinar se uma licença de software pode ou não ser marcada com certificação de código aberto, chamado de *Open Source Definition*, e baseado na *Debian Free Software Guidelines*.

O termo *Open Source* foi tão amplamente aceito pelo mundo corporativo, que está sendo aplicado em outras áreas, tais como o *Open Source Ecology*, que é um movimento para tentar descentralizar a tecnologia, a fim de que a mesma possa ser utilizada por qualquer pessoa.

Figura 3 - Logo Open Source Initiative (Google, 2016).

Há uma filosofia de desenvolvimento para softwares de código livre que diz que o mesmo deve obedecer a um Bazar. Enquanto o desenvolvimento tradicional obedece ao modelo de Catedral. Estas duas filosofias foram publicadas em 1997, em um ensaio chamado "A Catedral e o Bazar". Eric S Raymond compara, através deste ensaio, os dois modelos de desenvolvimentos: o tradicional, que conta com engenheiros de software, desenvolvedores e interessados no projeto (arquitetos), além de responsáveis pelo gerenciamento do projeto e pessoas responsáveis pela execução, com objetivos claros e bem definidos; o bazar, defendido pelo Raymond, considera que os papeis não são claramente definidos, visto que qualquer um pode modificar o software à sua maneira.

Segundo Eric S Raymond (1997), percebe-se que no modelo bazar os objetivos não são claramente definidos, pelo fato de que todos podem acessar, estudar, modificar e publicar as modificações para a comunidade. Portanto, sugeriu-se que tal modelo deveria seguir determinados padrões, que são:

- O código depende de usuários para que o mesmo possa evoluir. Portanto, cada usuário deverá ser tratado como um co-desenvolvedor. Quanto mais desenvolvedores puderem acessar o arquivo para modificá-lo, mais bugs poderão ocorrer, pois cada um tem habilidades diferentes, ambientes diferentes, sistemas operacionais diferentes;
- Quanto mais cedo se publicar um código aberto, mais rapidamente se encontrará co-desenvolvedores. Portanto, deve-se publicar logo quando terminado;
- O mesmo deve acontecer com integrações, para que se possa evitar a fixação sobrecarregada de erros no final do ciclo de vida do projeto.
- Como o software é trabalhado de forma cooperativa, o mesmo deve ter pelo menos duas versões: uma mais estável, para que possa ser utilizado por qualquer pessoa; e a versão *buggy*, que é conhecida como versão de desenvolvimento, para usuários que querem desfrutar de novas *features*, mesmo cientes de que estão prestes a utilizar um código ainda em desenvolvimento. Quando se utiliza a versão *buggy*, o usuário se torna um co-desenvolvedor, e pode relatar falhas, erros, e até mesmo fornecer correções para os mesmos.
- O software deve ser desenvolvido por módulos, pois assim pode ocorrer o desenvolvimento de diferentes módulos em paralelo.
- É necessário que o software possa suportar mudanças de necessidade dos usuários/consumidores, pois o mercado muda rapidamente e, portanto, mudam-se os requisitos. Portanto, deve-se estruturar a tomada de decisão estratégica, mesmo que informalmente.

2.3 SISTEMA OPERACIONAL LINUX

O sistema operacional Linux é o nome dado ao software que utiliza o Kernel Linux. Este sistema é responsável pela interface de comunicação entre o hardware, software e o usuário (BRITO, 2015). Vale ressaltar que Linux é o núcleo, também conhecido como *Kernel*, que faz a ligação entre o hardware e o sistema operacional, além de administrar os softwares instalados na máquina. A fundação GNU foi responsável por criar programas que possam interagir com tal *kernel*, portanto muitos referenciam ao sistema operacional em questão como GNU/Linux (LINS, 2009).

Em 1985, a Free Software Foundation tinha a pretensão de buscar um kernel que fosse compatível com os ideais de software livre, e posteriormente o Linux encaixou-se com tais ideais. O finlandês Linus Torvalds iniciou o aprimoramento do kernel MINIX pessoalmente em 1989, vindo a ser relatado para outros indivíduos mais tarde, e informado que o código estaria sendo aberto para que pudessem ajuda-lo a aprimorar o kernel. O kernel foi anunciado no dia 5 de outubro de 1991, e foi o encaixe perfeito para a fundação de software livre (SILVA, 2009).

Pelo fato de ser um software livre e código aberto. Vários co-desenvolvedores estão trabalhando para melhorar o sistema, colocar novas *features* e novas versões do mesmo. Para tanto, foi criado o Linux Foundation, um consórcio sem fins lucrativos para que o sistema operacional em questão possa crescer. A fundação patrocina o trabalho de desenvolvimento e aperfeiçoamento do Linux, e também é apoiada por empresas líderes em Open Source e Linux. Além disso, a mesma tenta promover o sistema para consumidores e co-desenvolvedores em potencial, ao dizer que o Linux fornece um conjunto de serviços tão abrangente e eficaz quanto os sistemas operacionais de plataforma fechada.

Um exemplo de sistema que utiliza o núcleo Linux é o Ubuntu. O nome veio à partir de uma filosofia africana para a humanidade com outros, ou seja, trata-se da maneira a qual o ser humano age em sociedade e sobre a essência do mesmo. Trocando em miúdos, o Ubuntu significa compaixão, desejo de paz e felicidade para todos os seres humanos (SIGNIFICADO, 2016).

Foi escolhido este nome para o sistema pela ideologia do projeto, que realça a maneira o qual é desenvolvido e modificado através de colaboradores em todo o mundo. Além disso, o produto é um software livre de código aberto, o que realça o compartilhamento de seu código com qualquer pessoa gratuitamente.

Atualmente o Ubuntu é o sistema baseado em Linux mais popular do mundo, e existe há mais de dez anos. Muitos são os motivos para a popularidade do mesmo, dentre eles podemos destacar a gratuidade, simples, de fácil instalação e acessível à todos, além de estar em constante desenvolvimento, tanto no design quanto em funcionalidades (PORQUÊ, 2016). A logo do Ubuntu está ilustrado na imagem 4.

Figura 4 - Logo do Ubuntu (Google, 2016).

O LAMP é o conjunto de softwares livres e de código aberto que se juntam para formar um servidor web, de modo a hospedar websites dinâmicos e aplicações web, o que torna o GNU/Linux ferramenta chave para o mesmo. Pode-se destacar também que há um conjunto de ferramentas e frameworks suficientes para o desenvolvimento de aplicações web e de outras naturezas. Um bom exemplo é o Aptana Studio, que é desenvolvido em Java e suporta as linguagens HTML, JavaScript, CSS, ScriptDoc, PHP, Ruby on Rails, Adobe Air, entre outros. Este IDE é baseado no Eclipse, mas não é necessário a instalação do mesmo, pois há a versão "standalone". Outro IDE muito conhecido é o Bluefish, que é focado na criação e edição de websites dinâmicos. O mesmo está disponível em diversos outros sistemas operacionais (FERREIRA, 2014).

2.4 APACHE HTTP SERVER

Apache HTTP Server é o servidor web livre mais bem sucedido, sendo a principal tecnologia da Apache Software Foundation atualmente, conhecida por vários projetos que envolvem tecnologias de transmissão via web, sistemas distribuídos e processamento de dados. Ele é compatível com o HTTP 1.1 e mantém um conjunto de módulos, além de permitir que o usuário possa escrever os seus próprios módulos utilizando o conjunto de rotinas e padrões fornecidos pelo próprio software. A Figura 5 ilustra a logo da empresa.

Figura 5 - Logo do apache http server (Google, 2016).

O nome “Apache” foi dado pelo Apache Software Foundation, referenciando uma tribo de americanos nativos que tinham grande resistência e estratégias superiores de combate. Esta tribo se tornou símbolo da resistência da filosofia de software livre para os interesses privados. Além disso, o servidor Apache tem uma excelente estabilidade e um conjunto vasto de ferramentas, que podem ser utilizadas em qualquer situação na web, o que faz alusão às estratégias superiores de combate do povo apache. Outra corrente diz que o nome viria da expressão “a patchy server”, que significa “servidor remendado”, já que o mesmo foi criado sobre o código do servidor da NCSA existente (THE APACHE SOFTWARE FOUNDATION, 2016).

2.4.1 Apache Software Foundation (ASF)

A ASF é uma organização descentralizada de desenvolvedores sem fins lucrativos, formada para criar e melhorar os projetos de código aberto, como o Apache HTTP Server. Estes projetos de software livre são distribuídos sob a licença Apache. Para ser membro da fundação, o usuário deve ter participado ativamente de projetos Apache, e ser aceito pela maioria dos usuários membros da fundação.

Os projetos são desenvolvidos com a ajuda de voluntários individuais, sendo estes membros ou não. Além disso, pessoas são escolhidas para gerenciar os projetos, dentre todos os participantes do projeto em questão, e os candidatos são os que mais atuam no mesmo.

A fundação tem o objetivo de proteger os co-desenvolvedores e participantes do projeto, e prevenir que outras organizações utilizem o nome da associação sem a permissão. Os membros desta elegem periodicamente um Conselho Administrativo, que são responsáveis por gerenciar assuntos organizacionais da mesma. O conselho aponta o número de funcionários para supervisionar o dia-a-dia da ASF (HOW, 2016).

Vários projetos são desenvolvidos por esta fundação, dentre os tais podemos destacar:

- OpenOffice.org: como o nome diz, é um conjunto de softwares abertos para escritório.
- Cocoon: framework de publicação de XML.
- Apache Geronimo: um servidor de J2EE.
- Apache Hadoop: framework para sistemas distribuídos.
- Lenya: gerenciamento de conteúdos.

- Maven: software de gerenciamento de projetos.
- Web services: conjunto de projetos relacionados à área de serviços web.
- Apache XML: como o nome já diz, é uma ferramenta para XML.
- Struts: framework utilizado para o desenvolvimento de aplicações web em Java.

2.5 BANCO DE DADOS

Bancos de dados são coleções de dados que se organizam e se relacionam de forma que se criem informações, proporcionando eficiência na pesquisa e estudo (KOTARO *et al*, 2005). Atualmente são a chave para qualquer sistema de informação. Basicamente todos os softwares desenvolvidos hoje utilizam algum tipo de banco de dados.

Visualmente, os bancos são um conjunto de tabelas com um ou mais atributos. Estes atributos são necessários para distinguir os dados da tabela em questão e serão importantes para que os dados sejam gerenciados de maneira correta através de comandos. Estes comandos serão vistos na seção 2.5.2.

As organizações acumulam cada vez mais informação, e por causa disso a manipulação manual de dados – através de escrita em documentos – se tornou praticamente impossível. Outro fator importante é o desgaste mental e físico do operador ao analisar uma grande pilha de documentos, o que ocasiona maior tendência à erros na busca de uma informação. Por fim, os papéis são passíveis de desgaste com o passar do tempo, e necessitam ser reescritos, caso contrário o documento estará perdido. Portanto, toda empresa necessita ter uma base de dados que possa armazenar dados em geral, como modelo de negócio, processos, usuários, entre outros.

2.5.1 Sistemas Gerenciadores de Banco de Dados (SGBD)

Os Sistemas Gerenciadores de Banco de Dados (SGDB), como o próprio nome diz, são um conjunto de softwares que gerenciam um banco de dados. É ele que controla o acesso, manipulação e organização dos dados. Um SGBD é, basicamente, um conjunto de programas que permitem guardar, extrair e modificar informações guardadas no banco.

Nota-se que os softwares utilizam a informação como alicerce do mesmo, desde programas em computadores pessoais até sistemas associados à mainframes. Assim, um

SGBD é a interface entre os programas de aplicação e os dados físicos armazenados. Os componentes de um SGBD são:

- Linguagem de definição de dados (LDD) – como o próprio nome diz, este componente define os elementos de dados, além de especificar conteúdo e estruturas a base de dados. Basicamente é uma linguagem de computador que define as estruturas de dados, assim como esquemas. Quando compilados, os LDDs são armazenados em dicionários de dados. Declarações CREATE, DROP, ALTER, além de esquemas XML e definições de colunas, estão presentes neste componente.
- Linguagem de manipulação de dados (LMD ou DML) – diferente da LDD, a LMD não trata da estrutura de armazenamento, mas do dado propriamente dito. Pode-se destacar, portanto, os comandos de selecionar, inserir, atualizar e remover dados. As declarações de LMD são do tipo imperativas, que descrevem o que o SGBD deverá realizar, e não como ele irá realizar. Pelo fato de depender da estrutura os quais os dados estão armazenados, há vários tipos diferentes de LMD entre distribuidores de bancos de dados. Por causa disso, a American National Standards Institute (ANSI) estabeleceu um padrão para o SQL, mas os distribuidores ainda fornecem suas extensões como padrão, enquanto não é implementado o padrão por completo.
- Dicionário de dados – é basicamente um conjunto de tabelas de dados que mantém, dentre outros:
    - Definição sobre os elementos de dados;
    - Papéis, privilégios e perfis de usuários;
    - Descrição de objetos;
    - Integridade e suas restrições;
    - Stored procedures (comandos SQL para otimização do banco) e gatilhos;
    - Alocação de espaço;

Existem vários modelos de SGBDs, que definem como os dados serão mantidos no banco de dados. Entre os principais modelos, podemos citar:

- Hierárquico – utiliza a estrutura de dados de árvores para organizar as tabelas do banco de dados (GIOVANANGELO, 2011).

- Em rede – diferente do modelo hierárquico, o modelo de rede utiliza grafos para organizar as tabelas do banco.
- Relacional – propõe o modelo de entidade e relacionamento para a organização das tabelas, em duas dimensões.
- Orientado a objetos (BDOOs) – são indicados para o tratamento de dados complexos e dinâmicos, como textos, gráficos, imagens, programas, entre outros. Os SGBDOOs combinam conceitos de objeto com a capacidade dos bancos de dados, e por isso são ditos como repositórios poderosos para a aplicação. Vale a pena ressaltar, também, que o processo de engenharia de software atual é voltado para o desenvolvimento de sistemas orientados a objetos, e a combinação do banco de dados orientado a objetos com este pode resultar em menos trabalho nesta parte, visto que o diagrama de classes pode resultar no banco de dados em si. Não é muito usado pois as empresas já atuantes e com boa parte dos sistemas de informação já feitos não arriscariam a mudança, pois gera custo e tempo para ser modificado. Além disso, a maioria dos programadores e gerentes de TI conhecem e programam apenas o modelo relacional. A principal característica do SGBDOO é armazenar e modelar estruturas complexas juntamente com o seu comportamento (GREIN et al, 2016).

Para o desenvolvimento tradicional de sites que contenham banco de dados, o desenvolvedor poderá optar por dois SGBDs diferentes: MySQL e MariaDB. Estes sistemas gerenciadores serão vistos ao decorrer do trabalho.

2.5.2 Structured Query Language

Structured Query Language é uma linguagem de consulta padrão que interage com os principais bancos de dados, através dos SGBDs. Esta linguagem foi criada em 1974, no laboratório da IBM. A intenção era se criar uma interface para o SGBD SYSTEM R.

A linguagem é declarativa, pois o usuário somente precisa declarar o que deve ser feito – o objetivo da consulta – para que a mesma seja executada. É através desta linguagem que o usuário poderá armazenar, alterar e remover tabelas e dados dentro de um banco. Há outras linguagens no mercado, mas o SQL é a linguagem mais utilizada. Mesmo assim,

existem várias extensões do SQL para diversos bancos de dados, e por causa disso criou-se o padrão SQL pelo ANSI, em 1986, e em 1987 pelo ISSO (SIGNIFICADO, 2016).

Através de comandos SQL, o usuário pode gerenciar os dados de um banco. Os principais comandos são:

- Insert – insere dados em uma tabela. Cada inserção costuma resultar em uma linha na tabela que foram inseridos os dados.
- Update – Altera os dados de uma linha ou tuplas da tabela.
- Delete – Remove uma linha ou dados de uma tabela.
- Select – Seleciona dados dentro de uma ou mais tabelas.

Segundo Diego Macêdo (2011), o SQL é uma linguagem simples e tem um número bem limitado de comandos, além de ser bem estruturado. Portanto, pode-se dizer que é bem parecido com o inglês. Os comandos têm uma estrutura bem similar, como o comando abaixo:

SELECT (atributos da tabela) FROM (nome da tabela) WHERE (condição);

Onde,

- Atributos da tabela – característica(s) da tabela. Também podem ser chamados de campos da tabela. Visualmente são os nomes dados às colunas da mesma. Pode-se colocar um ou mais atributos no mesmo comando, e até mesmo todos os atributos da tabela utilizando o caractere "*";
- Nome da tabela – necessário para se identificar a tabela para extração de dados.
- Condição – uma vez que a tabela e os dados da mesma foram escolhidos, podem-se colocar as condições para a extração dos dados. Coloca-se operadores de comparação entre duas expressões.

Além de operadores de comparação, pode-se juntar as expressões com operadores lógicos para otimizar a seleção dos dados nas tabelas. Estes operadores estão listados abaixo:

- AND – retorna o valor VERDADEIRO quando ambas as expressões forem verdadeiras.
- OR – retorna o valor VERDADEIRO se pelo menos uma das condições for verdadeira.
- NOT – retorna o valor VERDADEIRO se a condição for falsa.

Outras condições utilizadas são BETWEEN, IN, LIKE e IS. A condição BETWEEN é utilizada para encontrar dados que tenham um valor entre dois diferentes. IN é utilizado para encontrar dados que tenham valores contidos no explicitado. Já o LIKE é usado para encontrar dados que tenham valores parecidos com o escrito. Por fim, a condição IS indica um valor ou alguma condição do dado a ser analisado (MACÊDO, 2011).

Há uma visão ainda em construção conhecida como NoSQL. Com o número imenso de dados – que chamamos de Big Data – e manter a escalabilidade dos softwares de informação pode ser custoso demais. Os bancos de dados relacionais tentam escalar os dados, portanto, quanto maior o banco, mais difícil escalá-lo. O nome NoSQL foi utilizado pela primeira vez em 1998, e voltou nos tempos atuais pelo problema de escalabilidade devido ao grande número de dados armazenados.

Dentre os projetos NoSQL, pode-se destacar o RavenDB, MongoDB, CouchDB, Voldemort, DEX (grafos), entre vários outros projetos open source (NASCIMENTO, 2010).

2.5.3 MySQL

Dentre os SGBDs utilizados para o desenvolvimento tradicional de sistemas web, o MySQL é o mais popular. O seu nome significa “meu SQL”, portanto pode-se inferir que a linguagem utilizada como interface é o SQL. É uma base de dados open source e, portanto, gratuita. Além disso, pode ser utilizado em diversos sistemas operacionais. O software é um dos componentes centrais para qualquer aplicação web tradicional.

Segundo o site oficial do MySQL (2016), o mesmo deve ser utilizado para que se possa economizar tempo e dinheiro enchendo as bases de dados para seus sites, sistemas críticos de negócio, entre outros. Por causa disso, várias grandes empresas e organizações o utilizam. Pode-se citar o Facebook, Google, Adobe, Alcatel Lucent e Zappos.

O projeto foi desenvolvido e publicado pela empresa sueca MySQL AB em maio de 1995, e logo depois a mesma foi comprada pela Sun Microsystems, e mais tarde pela Oracle Corporation. Com esta compra, a empresa restringiu o uso em certos aspectos, apesar de ter mantido uma versão para a comunidade. Por causa disso, um grupo de desenvolvedores colaboraram entre si para o desenvolvimento de forma aberta do MariaDB partindo da versão 5.1 do MySQL, que será visto mais adiante (PISA, 2016). A figura 6 mostra a logo do MySQL.

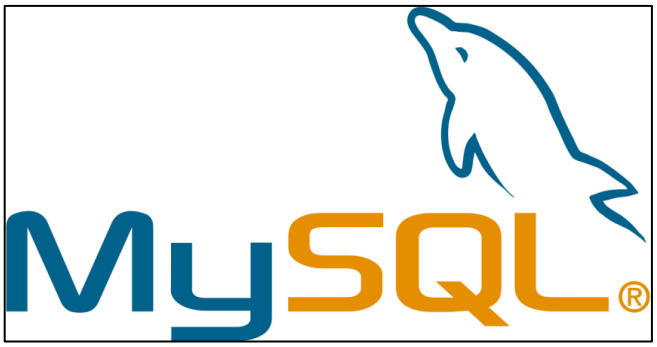


Figura 6 - Logo do MySQL (Google, 2016).

2.5.4 MariaDB

Michael ‘Monty’ Widenius, criador do MySQL, ao presenciar a venda do SGBD para a Oracle, temia que a mesma fechasse o código fonte do sistema, e o comercializasse. Foram vários recursos até que a Oracle concordasse em manter os termos de licença do software, e ao mesmo tempo a fusão.

A partir da fusão, Michael começou a desenvolver um software gerenciador de banco de dados que tivesse uma alta fidelidade ao MySQL, o que acarretou no surgimento do MariaDB. Assim, o MariaDB pode ser considerado o irmão do MySQL. Todos os comandos, componentes, bibliotecas e interfaces utilizados no MySQL também são utilizados no MariaDB.

A grande diferença entre o MariaDB e o MySQL é a filosofia do software livre. Enquanto a Oracle, apesar de ter um acordo com a Sun para a continuação da licença do SGBD, o mesmo pode acabar com a licença deste. Além disso, percebe-se uma leve queda na carga do hardware, além da velocidade e total compatibilidade com o MySQL. A simples mudança de banco de dados já faz com que o servidor possa ganhar velocidade e diminuir o consumo de recursos do mesmo. Por causa disso, algumas grandes empresas já estão migrando para o MariaDB, como o Wikipedia.

2.6 LINGUAGEM DE PROGRAMAÇÃO PHP E PYTHON

No conjunto de softwares utilizados pelo desenvolvimento tradicional, temos as linguagens de programação PHP e PYTHON. Elas são responsáveis por ser o “*backend*” do web app, ou seja, comunicam-se com o SGBD responsável para coletar os dados e apresentá-los para o usuário de forma que ele não possa participar e nem ver o código responsável por

tal. As linguagens consideradas *backend* são responsáveis pela dinamicidade do conteúdo no web app.

2.6.1 Frontend vs Backend

Para se entender melhor qual o propósito das linguagens utilizadas no método tradicional de desenvolvimento, é necessário entender a diferença entre estas para outras linguagens utilizadas durante o processo (HTML, CSS, entre outros).

Também conhecidas como *client-side*, ou lado cliente, as linguagens *frontend* são responsáveis por organizar visualmente a interface entre o usuário e o *backend*, coletando dados do usuário e transformando-o em uma entrada que o *backend* possa entender e processar. Em resumo, pode-se dizer que as linguagens *client-side* são responsáveis pela comunicação do software com o cliente (neste caso o usuário). O desenvolvimento deste lado normalmente é feito por designers, pois são responsáveis em deixar a interface mais amigável ao cliente.

Já o *backend*, conhecido como *server-side*, apresenta linguagens que apenas o servidor entenderá, ou seja, o processamento dos dados é feito pelo servidor, e o mesmo retorna informações para o *client-side* (caso seja solicitado e aprovado). Outras operações, como se comunicar com o SGBD correspondente, também fazem parte do *backend*. Basicamente todo código realizado pelo servidor faz parte do *server-side*, e todo processamento realizado pelo browser do usuário faz parte do *client-side*. A figura 7 ilustra a diferença.

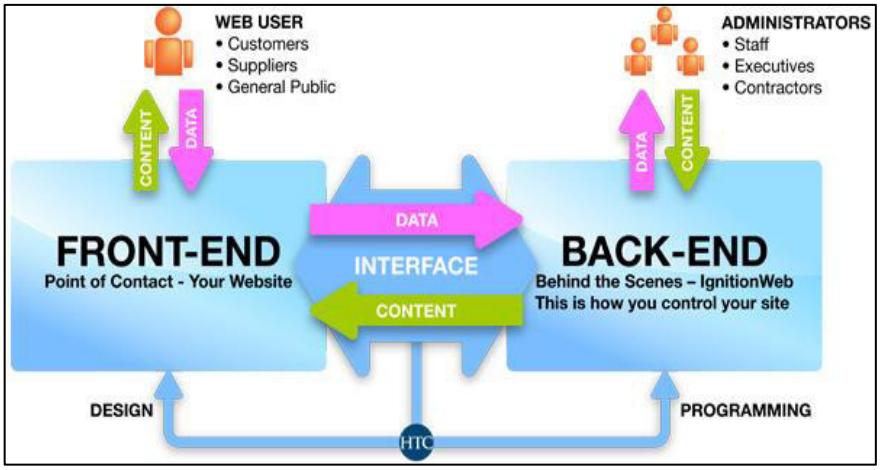


Figura 7 - Gráfico da interação entre front-end e back-end (3NY TECHNOLOGY, 2016).

Alguns exemplos de linguagens *client-side* são o HTML, CSS e JavaScript. Já as linguagens server-side são PHP, Python, Ruby, Java, ASP, entre outros.

2.6.2 Por que PHP?

Quando se começa a trabalhar com a linguagem, logo se percebe a semelhança com as linguagens de programação mais conhecidas no mercado (C e Java). Pode-se citar vários outros fatores para se usar o PHP:

- Fácil instalação, utilização e aprendizagem.
- Linguagem simples e flexível.
- Amplamente utilizado e com várias bibliotecas de terceiros.
- Vários tutoriais e manuais, dentre vídeo aulas e documentos, e scripts na internet.
- Software Livre de código aberto.
- Apesar de o objetivo principal ser uma linguagem server-side para desenvolvimento web, o PHP também pode ser utilizado como um script de linha de comando, e até mesmo criar aplicações desktop.
- Suporte para praticamente todos os servidores web e sistemas operacionais existentes atuais. Além disso, possui muitas maneiras de interação com os bancos, usando até mesmo camadas de abstração, como o PDO, ou até mesmo usando ODBC.
- Pode utilizar programação estruturada ou orientada a objeto, ou até mesmo ambas.
- Processamento de textos poderoso, incluindo compatibilidade de expressões regulares em Pearl (PCRE).
- Vários frameworks.

Como se pode ver, são vários os fatores que acordam para a utilização da ferramenta. Ela dá suporte para todos os tipos de desenvolvedores, inclusive os iniciantes. Além disso, podemos citar a ampla utilização da linguagem por diversas organizações, o que dá mais credibilidade à mesma. Uma empresa que contrata desenvolvedores web costumeiramente pede conhecimento na linguagem PHP.

2.6.3 Por que Python?

Segundo a organização Python Brasil, são vários os motivos para se utilizar a linguagem em questão, os quais se destacam:

- Dúzias de frameworks para o desenvolvimento de aplicações web;
- Ideal para o desenvolvimento de softwares para gestão empresarial, com frameworks e bibliotecas que agilizam o mesmo. Pode-se citar como exemplo o Gazpacho, SQLObject, Kiwi, PyGTK, entre outros;
- Pode ser utilizado como linguagem para dispositivos móveis;
- Pode ser utilizado para scripting para softwares de multimídia e entretenimento em geral, até mesmo para jogos;
- Variedade de bibliotecas voltadas para a ciência;
- Pode ser utilizado para o ensino;
- Software livre e de código aberto;
- Orientado a objetos e de alto nível.
- Pode ser utilizado pela maioria dos sistemas operacionais;
- Boa legibilidade, pois assegura que o código deva ser identado;
- Simples.

Em resumo, o Python é uma linguagem que tem o propósito geral, além de ter funções e maneiras de codificação simples. Além disso, é altamente popular, e com uma comunidade grande e ativa, o que impacta na documentação do mesmo.

2.7 NOVO MODELO DE SERVIÇO

Mesmo com várias ferramentas e servidores ao redor do mundo que oferecem suporte às mesmas, o desenvolvimento tradicional apresenta problemas em um aspecto que não é somente físico, mas também empresarial, que afeta diretamente na implantação e gerenciamento dos servidores: a pouca escalabilidade.

Quando uma aplicação web é criada, não se sabe ao certo quantos serão os acessos ao site ou aplicação, nem com qual frequência a mesma irá guardar os dados no banco de dados. Por este motivo, alguns aplicativos web precisam melhorar a sua capacidade de

processamento, ou aumentar o disco rígido para que se possa guardar mais informações, o que gera problemas.

Para que se possa mudar qualquer configuração física do servidor, normalmente é necessária a quebra de contrato, o que pode acarretar em multas e até mesmo no bloqueio do site em relação à empresa prestadora do serviço. Quando um site, por exemplo, é acessado por um número maior de pessoas do que o servidor pode aguentar, algumas prestadoras acabam retirando-o do ar para aliviar o servidor, e entram em contato para explicar o ocorrido. O usuário precisa mudar de plano, ou até mesmo mudar de hospedagem.

Por esses e outros motivos, surgiu a computação em nuvem, que será vista a seguir.

3 COMPUTAÇÃO EM NUVEM

Um dos grandes fatores que levam as organizações quebrarem contrato com empresas de hospedagem web é a falta de escalabilidade, pois não há como mensurar precisamente o número de acessos ao aplicativo. O que se pode fazer é uma estimativa da relação acesso vezes o número de bytes utilizados para a interação entre o servidor e cliente. Quando estes valores estiverem altos demais, é necessário melhorar o hardware do servidor. Este é o grande problema da contratação de servidores na forma de desenvolvimento tradicional.

Quando se contrata um servidor nesse modelo, é necessário ter em mente um hardware inicial melhor do que o necessário no momento da contratação, pois deverá haver uma "margem", o que proporciona um gasto maior de recursos que, no primeiro momento, estarão ociosos. Percebe-se que o contrato do serviço é fixo, imutável. Tal configuração pode tornar até inviável o desenvolvimento de aplicações que necessitam constantemente de mais recursos. Como uma das formas de contornar o problema, surgiu a computação em nuvem.

Segundo a Amazon, a computação em nuvem é um conjunto de serviços contratados sob demanda. Assim, o usuário paga apenas o que utiliza de recursos de Infraestrutura, Software e Hardware, o que promove uma alta escalabilidade e redução de custos por parte do cliente no investimento inicial do serviço. Outro bom motivo, segundo a empresa, é a economia de tempo nas atividades de manutenção e gerenciamento do hardware, uma vez que a empresa contratada é responsável pelo mesmo.

Pode-se destacar, também, a possibilidade de acessar arquivos e executar funcionalidades pela internet, sem que haja a necessidade de instalação de aplicativos para executá-los. Basta apenas uma internet com boa capacidade e um navegador. Trocando em miúdos, é necessário apenas estar conectado à internet – e ter um contrato com a prestadora – para utilizar os serviços oferecidos pela nuvem, pagando apenas o que for utilizado.

Por precisar apenas da conexão de internet, os serviços podem ser acessados de qualquer dispositivo que tenha um navegador, desde celulares até computadores de mesa.

Por meio deste novo modelo de serviços pode-se acessar desde editores de texto, como o Office365, até jogos online, como o NVIDIA GRID, com o modelo de Game as a Service (GaaS). São inúmeros os serviços que podem ser propostos através da computação em nuvem, em todas as áreas. Na verdade, boa parte das aplicações que podem ser feitas de forma tradicional, mesmo que voltadas para desktops, podem ser escritas de forma a serem utilizadas em nuvem, o que não significa dizer que todos os aplicativos devem seguir tal modelo.

O blog de tecnologia Tecmundo cita alguns exemplos de serviços utilizados na computação em nuvem:

- Dropbox – serviço de sincronização de arquivos em nuvem, reserva-se um espaço em disco do dispositivo para colocar os arquivos que se queira compartilhar/sincronizar.
- Vários aplicativos do Google são ditos "em nuvem", pois possibilitam o acesso dos mesmos em qualquer parte do mundo sem precisar de aplicativos adicionais. Dentre estes se destacam o Google Talk, Google Maps, Google Docs, e até mesmo o Chrome OS, que está sendo desenvolvido para utilizar boa parte de seus aplicativos em nuvem.
- Da Microsoft, destacam-se o SkyDrive, Live (Word, Excel e Powerpoint, One Note). A Microsoft aposta em serviços de edição de texto, e até armazenamento.
- A principal ferramenta da Apple é o iCloud, anunciado junto com a versão 5.0 do iOS. Esta ferramenta integra dados do computador com os dispositivos móveis da empresa, sincronizando e-mails, opções de navegador e várias outras coisas.

Mesmo com tantas vantagens ao se utilizar esta nova modalidade de serviço, ainda existem desvantagens, que podem inviabilizar a utilização do mesmo em determinados casos. Dentre as desvantagens, duas se destacam: a insegurança e a conexão de internet instável ou de baixa velocidade.

Um dos principais problemas de se utilizar o serviço em nuvem é a insegurança nos serviços. Algumas empresas não utilizam este serviço porque não querem colocar os seus dados e informações sigilosas em nuvem, mesmo que o serviço seja "seguro", afinal os arquivos estão disponíveis para qualquer dispositivo que tenha conexão à internet. Isso, entretanto, é discutível, uma vez que o fato de uma empresa ter um servidor próprio, hospedando seus dados dentro da sua rede local, não garante por si só segurança, pois o servidor sempre será passível de ataques de hackers.

As empresas também negligenciam a capacitação de profissionais, pois há custos para a mesma, além de acarretar em pequenas mudanças no processo de desenvolvimento, afetando alguns pontos das regras de negócio. Boa parte das empresas brasileiras, e principalmente as maranhenses, têm medo de inovar, pois acham que as inovações poderão

acarretar em problemas estruturais nas mesmas. Além disso, não há divulgação ou cursos de desenvolvimento que envolva serviços em nuvem no Maranhão, e a solução do usuário para este impasse se torna pesquisar na internet as peças para a criação, ou até mesmo adquirir um curso de desenvolvimento na internet. No Brasil, a computação em nuvem ainda está engatinhando. Há pouco a Embratel criou um conjunto de serviços de computação em nuvem chamado de CloudServer.

Outra desvantagem, já no lado do usuário, é a questão da velocidade e as oscilações que a conexão mal instalada ou com contrato de baixa velocidade pode oferecer. Como se sabe, os serviços disponibilizados estão em nuvem, e todos eles necessitam de um navegador ou aplicativo específico para que possa ser acessado, pois é o servidor que processa os dados e disponibiliza para o usuário. Caso a internet esteja com problemas, a comunicação entre o usuário e o serviço fica limitada, e dependendo do serviço e da velocidade, não poderá ser estabelecida. No Brasil, inclusive no Maranhão, este problema ainda é visível, pois as empresas de telefonia quase nunca conseguem entregar uma internet de qualidade, dentro dos parâmetros estabelecidos do contrato de serviço com o cliente.

A utilização de serviços de computação em nuvem está se tornando cada vez mais comum no mundo do desenvolvimento, pois permite o gerenciamento inteligente dos recursos da(s) máquina(s) pagando apenas pelo que foi consumido. Para tanto, é necessário que se aprenda a utilizar os diversos serviços oferecidos e encontrar aqueles que melhor se adequem à elaboração da solução web a ser implementada.

Vale ressaltar que nem todas as soluções poderão ser melhor aproveitadas utilizando apenas computação em nuvem, pois o que indicará o uso do mesmo é o estudo de viabilidade econômica e técnica. Em geral, as soluções que utilizam este novo conjunto de serviços se tornam menos onerosas economicamente, mas não é uma regra. Além deste, há outros fatores que influenciam diretamente na utilização deste conjunto como, por exemplo, o pouco interesse na migração dos aplicativos para os serviços oferecidos, e até mesmo a ideia (errônea) de que as soluções são mais difíceis de serem instaladas em nuvem.

Como visto anteriormente, são inúmeros os serviços oferecidos pela computação em nuvem. Estes serviços estão divididos em categorias, de acordo com a necessidade do usuário, no modelo *pay-per-use* (pague pelo uso), que serão vistos a seguir.

A utilização de serviços de computação em nuvem da Amazon (AWS) se mostrou bastante competente e viável de acordo com a necessidade de desenvolvimento da solução proposta. Além disso, a mesma oferece outros recursos que podem ser explorados de acordo com o projeto de desenvolvimento.

A desinformação deste novo conjunto de serviços é tamanha que várias pessoas o utilizam e sequer sabem que este serviço oferecido é de computação em nuvem. As pessoas ainda acreditam que a computação em nuvem é apenas algo que possa ser guardado e acessado pela internet.

A computação em nuvem, como visto, vai além disso. Deve-se, portanto, propagar o conhecimento desta nova modalidade para que as mesmas possam entender as vantagens e desvantagens de utilizar os serviços oferecidos pelas prestadoras.

3.1 SOFTWARE AS A SERVICE (SAAS)

De modo tradicional, para se utilizar um software de uma determinada empresa como, por exemplo, o pacote da Adobe, é necessário pagar uma licença para a utilização do mesmo, sendo a mesma semestral, anual, bianual, etc. Em resumo, você paga pelo produto, independentemente de utilizá-lo corriqueiramente ou não. Já na computação em nuvem, o usuário só precisará pagar o serviço quando utilizar o mesmo, ou seja, você não paga para adquirir o aplicativo, mas para utilizar o serviço daquele aplicativo de acordo com a sua necessidade. Este é um dos serviços mais utilizados pelos usuários, mesmo que leigos. O Skype e Office365, além de alguns serviços da gigante Google se encaixam neste tipo de serviço. Assim, o usuário não precisará pagar por uma licença de software, mas apenas pelo uso do mesmo, que na maior parte das vezes, dá-se pelo tempo de uso, acarretando na diminuição de gastos.

3.2 INFRASTRUCTURE AS A SERVICE (IAAS)

Este modelo de serviço prevê a utilização de infraestruturas de TI como serviço, como servidores virtuais. Neste caso, ao invés de comprar servidores físicos e precisar de tempo e recurso para solucionar quaisquer problemas físicos do mesmo, o usuário pode optar por alugar um “servidor” por meio da nuvem, pagando apenas o que consumir do mesmo, em termos de processamento, memória, racks e outras “caixas”, entre outros.

Supondo que o usuário precise de vários servidores durante um período de tempo, este pode solicitar à empresa prestadora um número X de servidores, e cancelar a utilização destes no momento em que quiser, pagando apenas o que consumiu. A Amazon EC2 e IBM são grandes exemplos de prestadoras de serviço de tal tipo.

3.3 PLATFORM AS A SERVICE (PAAS)

Este modelo é utilizado quando se precisa de uma tecnologia mais robusta e necessária para executar determinado aplicativo ou recursos diferentes. Este é o modelo de utilização de aplicações mais flexível, pois combina o SaaS e o IaaS em um só. Pode-se dizer que o modelo é um meio termo entre os supracitados. O usuário pode desenvolver suas aplicações baseadas em diferentes tecnologias e utilizar a infraestrutura necessária para o desenvolvimento do mesmo.

Supondo que se queira utilizar um determinado serviço de software (SaaS) personalizado, e não pode ser feito por causa de limitações do fornecedor, e esta personalização é fundamental para o negócio da empresa, e os computadores desta não têm recursos suficientes para desenvolver a solução. A utilização da plataforma como serviço é recomendada em casos como esse, utilizando o modelo pay-per-use para desenvolver a aplicação com plataforma que desejar, incluindo Sistema Operacional, softwares e hardwares. Tem-se como exemplo o Google Cloud, que oferece um conjunto de serviços e infraestrutura para o usuário contratar e pagar apenas o que utilizar.

3.4 GAMING AS A SERVICE (GAAS)

Esta área de utilização é relativamente nova, se comparado com as outras. Um usuário pode contratar um servidor de jogos para jogar basicamente qualquer jogo que o fornecedor tenha, pagando apenas o tempo e recursos que utilizar. Os jogos são transmitidos via streaming, e o usuário, ao apertar os botões do dispositivo, irá enviar a mensagem para os servidores, que farão o movimento correspondente.

Neste modelo, a utilização do processamento de dados por parte do servidor é mais visível, pois se pode jogar em qualquer plataforma, com qualquer configuração de hardware, necessitando apenas de um browser, como smartphones, desktops, notebooks, netbooks, entre outros dispositivos. Em resumo, é o computador do fornecedor que está executando o jogo, e o resultado disso é transmitido para o usuário.

O Gaming as a Service pode ser classificado como um tipo de Software as a Service, uma vez que o jogo eletrônico é um software de entretenimento e lazer, com a diferença de utilização de recursos e maneira de interação com o game em tempo real. Este é similar ao streaming de vídeo, como o Netflix.

O grande problema do Gaming as a Service é a necessidade constante de boas conexões com a internet. O tempo de resposta entre os dispositivos deve ser a mínima possível, para que o usuário consiga jogar de forma satisfatória. Até mesmo pequenas oscilações ocasionam no aumento de latência, o que retarda o tempo de resposta, tornando frustrante a experiência que usuário tem com este modelo.

3.5 COMMUNICATION AS A SERVICE (CAAS)

Como o próprio nome diz, o CaaS é uma modalidade de computação em nuvem responsável pelo fornecimento de serviços de comunicação, como o VoIP ou mensagens instantâneas, além de poder gerenciar vídeo e outros serviços. O usuário poderá implementar os serviços e recursos a partir de uma base de serviços utilizados, seguindo o modelo *pay-as-you-go*.

3.6 DATABASE AS A SERVICE (DBAAS)

O banco de dados como seviço é amplamente utilizado pelos desenvolvedores de softwares em nuvem, pois o mesmo é responsável por prover banco de dados que pode ser acessado por basicamente qualquer aplicação que utilize a internet. Um aplicativo web, por exemplo, pode acessar o banco de dados MySQL em qualquer lugar, além de poder integrar diversas tecnologias e linguagens. A nuvem da Microsoft Azure, por exemplo, consegue integrar aplicativos em diversas linguagens, até mesmo o PHP. Outro exemplo é o RDS da Amazon.

4 AMAZON WEB SERVICES (AWS)

A Amazon Web Services é um conjunto de serviços de computação em nuvem oferecidos pela Amazon ao usuário. Começou a disponibilizar serviços de infraestrutura de TI, em 2006, para empresas. Atualmente este conjunto de serviços beneficia milhares de empresas ao redor do mundo, com vários datacenters localizados em pontos estratégicos do globo, para que o usuário possa utilizar um datacenter perto de sua localidade. Existem datacenters localizados no Brasil, Europa, EUA, Austrália, Japão e Cingapura. Estes datacenters estão instalados em regiões chamadas de *Availability Zones* (AZs), ou zonas de disponibilidade. O Brasil, por exemplo, é afetado diretamente pela zona de disponibilidade de São Paulo. A AWS opera com 33 zonas de disponibilidades ao redor do globo, em 12 regiões geográficas.

Mesmo que um usuário esteja perto de uma zona de disponibilidade, o mesmo pode selecionar outra zona, caso precise da mesma. Pode-se selecionar, por exemplo, a zona de Virgínia, mesmo que o cliente esteja no Brasil, e a zona mais próxima seja a de São Paulo. Ou seja, até mesmo a escolha do servidor depende da necessidade do cliente.

Segundo o site da AWS, os clientes que utilizam estes serviços têm as seguintes vantagens:

- Baixo custo – o usuário não precisa ter despesas iniciais apenas ao criar a conta, pois só precisa pagar quando utilizar o serviço. Além disso, a Amazon Web Services proporciona um modelo gratuito de conta, que será visto mais a frente.
- Agilidade e elasticidade instantânea – o usuário não precisa se preocupar em esperar dias, ou até meses, para que ocorra uma mudança de hardware. O mesmo pode contratar um servidor com determinadas características, e expandi-lo de acordo com o crescimento da demanda. O usuário pode escolher entre ter um único servidor virtual ou não, além de escolher o tempo que estes servidores irão trabalhar, e paga apenas pelo que utilizar.
- Aberto e flexível – os serviços prestados pela fornecedora independem de sistema operacional ou linguagem, o que significa que o usuário pode escolher qual sistema e linguagem for melhor para o desenvolvimento de aplicações. Assim, o cliente não precisa concentrar esforços na infraestrutura, e sim na solução em si, impactando nos custos da empresa.

- Seguro – a plataforma oferecida pela Amazon possui certificações e auditorias reconhecidas pelo setor, como o ISO 27001, FISMA, Moderate, entre vários outros. A mesma possui várias camadas de segurança física e virtual, garantindo a segurança e integridade dos dados.

A AWS oferece um conjunto de serviços de infraestrutura e hardware como serviços, divididos em subcategorias listadas abaixo:

- *Compute* – conjunto de serviços relacionados ao hardware como serviço. Este conjunto de serviços oferece um hardware computacional escalonável em nuvem, além de serviços para implantar e gerenciar aplicativos, entre outros.
- *Storage & Content Delivery* – como o próprio nome diz, esta subcategoria reúne serviços responsáveis pelo armazenamento de dados (arquivos) e entrega de conteúdo para usuários finais com segurança.
- *Database* – conjunto de serviços responsável pelo armazenamento de dados em bancos, além de migração de banco de dados. A Amazon oferece vários modelos de bancos de dados, desde relacionais até os do tipo NoSQL. Também pode utilizar memória cache para entregar informações rapidamente, melhorando o desempenho das aplicações.
- *Networking* – serviços relativos ao espaço virtual privado em nuvem (VPC), além de oferecer maneiras de utilizar o serviço DNS para criar e manter um domínio.
- *Developer Tools* – além de criar e gerenciar repositórios Git para o usuário, esta categoria permite automatizar completamente implementações de código, e gerenciar, modelar e visualizar os passos para lançar o aplicativo.
- *Management Tools* – conjunto de serviços responsáveis por gerenciar as ferramentas e serviços utilizados pelo cliente, provendo informações de uso, valores dos serviços e dá até mesmo opiniões sobre gastos desnecessários.
- *Security & Identity* – relacionado ao acesso e possíveis problemas de segurança dos serviços em nuvem da AWS, além de prover um gerenciador de arquivos em nuvem.
- *Analytics* – serviços de *data mining*, análise de logs, aprendizado de máquina, *streaming* em tempo real, entre outros.

- *Internet of Things* – composto por um serviço de conexão de aplicações em nuvem para outros dispositivos ou outras aplicações.
- *Mobile Services* – conjunto de serviços relacionados ao desenvolvimento, teste e instalação de aplicações mobile, além de sincronização de dados com outros dispositivos mobile.
- *Application Services* – serviços relacionados a APIs, conversores de media para nuvem, armazenamento, envio e recebimento de mensagens, streaming de aplicativos e jogos, fluxos de trabalho (*workflows*) e gerenciadores de busca para sites e aplicativos.
- *Enterprise Applications* – serviços de espaços de trabalho (*workspaces*) em nuvem, serviços de e-mail e armazenamento de documentos com controles administrativos fortes, ideal para armazenar arquivos da empresa e organizá-los através de permissões.
- *Game Development* – criação de games *multiplayer* em nuvem.

Como se pode ver existem vários serviços divididos em categorias, uns básicos e outros mais complexos. Na figura 8 pode-se ver como o dispositivo, neste caso um computador de mesa, acessa alguns destes serviços. Nos subcapítulos subsequentes serão vistos alguns destes.

Figura 8 - Ilustração do acesso aos serviços da Amazon.

4.1 AMAZON ELASTIC COMPUTE CLOUD (EC2)

O Elastic Compute Cloud (EC2) é um dos serviços mais utilizados da Amazon Web Services. Responsável por prover capacidade computacional que possa ser escalável. É basicamente uma hospedagem de um servidor virtual sob demanda, permitindo várias

instâncias criadas em minutos, para que não haja problemas em relação ao tempo, com uma interface bem intuitiva. Além disso, oferece controle de recursos computacionais, mudando a economia de computação para permitir o pagamento relacionado ao que se é usado, e o usuário poderá modificar o hardware de forma específica, desde memórias até CPUs. Por fim, o serviço fornece aos usuários ferramentas para isolar falhas comuns, ajudando a construir aplicativos resistentes às mesmas.

Ao utilizar o serviço, o usuário pode selecionar o sistema operacional a ser instalado no servidor, juntamente com um número de instâncias. As instâncias podem ser caracterizadas como servidores virtuais diferentes. Pode-se utilizar o sistema Windows Server em uma instância de EC2, e na outra utilizar o Ubuntu Server, por exemplo. Estas instâncias podem ser gerenciadas, com controle de saída das instâncias, podendo até mesmo interromper e inicializar determinada instância, mantendo os dados em uma partição de inicialização.

O serviço também trabalha em conjunto com o Amazon S3 para armazenamento de dados não estruturados, Amazon RDS para criação e gerenciamento de banco de dados relacional, SimpleDB para a criação e gerenciamento de banco de dados não relacional, SQS para serviço de e-mail, e VPC para segurança, que serão vistos mais a seguir.

4.2 AMAZON SIMPLE STORAGE SERVICE (S3)

Amazon Simple Storage Service (S3) é um serviço da Amazon que oferece armazenamento de objetos escalonável, com uma interface web simples, permitindo guardar qualquer dado em qualquer parte da web, sem custos de instalação e configuração.

Existem várias categorias de armazenamento para usos diferentes do mesmo. A categoria Standard armazena dados com frequência de acesso para aplicativos variados, o Standard – Ocasional Access para dados que têm longa duração, mas que são acessados com menos frequência. Por fim tem-se o Amazon Glacier, que trata de arquivos que têm longa duração.

Assim como o Amazon EC2, o Simple Storage Service pode ser usado com outros serviços, como o próprio *Elastic Compute Cloud* e o *Identity and Access Management* (IAM), entre outros. Além disso, contempla uma significativa segurança dos dados armazenados, pois contém certificados expressivos de segurança.

Utilizando os serviços do Amazon S3, o usuário poderá criar *buckets*, responsáveis por separar o conteúdo de uma aplicação ou conjunto de aplicações para outras.

Assim, o usuário precisará apenas criar um *bucket* e colocar arquivos dentro dele, sem se preocupar com a intervenção de outros *buckets*.

4.3 AMAZON RELATIONAL DATABASE SERVICE (RDS)

O Relational Database Service é um serviço que oferece banco de dados relacional altamente escalável ao usuário, além de gerenciar as tarefas de administração deste, para que o usuário se concentre na aplicação e no negócio. O RDS oferece seis SGBDs relacionais conhecidos, para que o usuário possa escolher aquele que precisar. São eles:

- MySQL;
- MariaDB.
- Amazon Aurora;
- Oracle;
- Microsoft SQL Server;
- PostgreSQL.

A Amazon também oferece serviços de banco de dados de diferentes tipos, como, por exemplo DynamoDB, capaz de criar bancos de dados NoSQL. Basta ao usuário escolher o tipo de banco de dados mais conveniente.

4.4 AMAZON VPC

O Virtual Private Cloud (VPC) permite seccionar a nuvem em partes isoladas logicamente, para executar os serviços da AWS em seções definidas pelo usuário. Assim, o usuário será capaz de criar sub-redes, roteamento, gateways, e intervalos de IP.

As configurações de rede podem ser facilmente personalizadas para cada VPC. Pode-se colocar, por exemplo, um *backend* responsável pelo banco de dados em uma sub-rede sem conexão com a internet, e criar outra sub-rede responsável pela comunicação com o público, com acesso à internet. Além disso, pode-se criar uma extensão para o *datacenter* do usuário com a nuvem da Amazon.

Este serviço apresenta várias opções de conectividade. O usuário pode estabelecer comunicação entre o VPC e a internet, outro *datacenter* ou até mesmo com outros VPCs.

Várias são as utilidades do Amazon VPC, dentre elas se destacam a utilização das nuvens privadas para a construção e disponibilização de um site para o público, hospedagem de aplicativos em multicamadas que estejam hospedados em um *datacenter*, ou até mesmo hospedados na própria VPC, com suporte a recuperação de desastres.

Em termos práticos, o VPC serve para isolar em camadas de acesso os diferentes aplicativos utilizados por uma determinada conta. Assim, um determinado VPC, por exemplo, poderá ter acesso apenas ao RDS, enquanto outro VPC poderá ter acesso ao EC2, e assim por diante.

5 DESENVOLVIMENTO DA SOLUÇÃO WEB NA AWS

Para que se possa utilizar os serviços da Amazon, é necessário que se crie uma conta na mesma, utilizando-se de um cartão internacional mesmo quando não há gastos pela utilização dos serviços da mesma. Alguns serviços são gratuitos na Amazon, desde que o usuário não ultrapasse o limite imposto por este provedor de nuvem pelo período de um ano. Este conjunto de serviços são conhecidos como "free tier eligible". Dentre os serviços gratuitos, destacam-se:

- Amazon EC2 – utilizando a instância t2.micro de Linux, RHEL, SLES ou Windows, o usuário terá 750 horas de uso mensal distribuídos entre instâncias diferentes. Com este tempo, o usuário poderá criar uma instância e mantê-la ativa por um mês inteiro, pois utilizará em média 720 horas, caso o mês seja de 30 dias. Já com duas instâncias, este período cai para duas semanas.
- Amazon S3 – 5GB de armazenamento, 20 mil requisições GET e 2 mil requisições PUT. Os arquivos já colocados anteriormente não serão pagos quando o serviço grátis expirar.
- Amazon CloudFront – 2 milhões de requisições HTTP ou HTTPS e 50GB de transferência de dados para fora.
- Amazon RDS – 750 horas utilizando instâncias Single-AZ db.t2.micro, 20GB de armazenamento para Propósito Geral (SSD) ou Magnético e mais 20GB para Backups, e 10 milhões de I/Os.
- Amazon ElastiCache – 750 horas de cache.t2.micro.
- Amazon SNS – 1 milhão de publicações e *push*, 100 mil entregas HTTP/S e mil entregas de e-mail, e não expira ao final dos 12 meses de uso da AWS.

Antes de se partir para a criação, é necessário que o usuário entenda as condições de "free tier" e saber quando ele está sendo taxado ou não pela cobrança de algum serviço utilizado. Para isso, basta utilizar o serviço de *Billing & Cost Management*, que poderá ser acessado clicando no menu do nome do usuário, na figura 9.

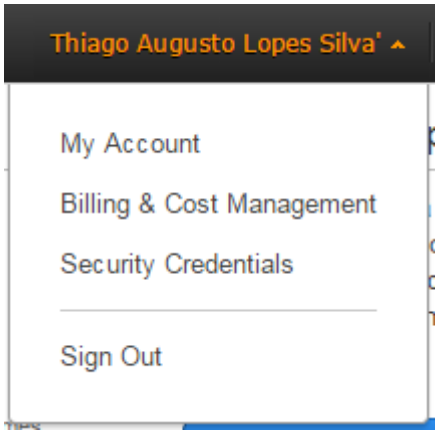


Figura 9 - Opções de usuário no site da Amazon (AWS, 2016).

Ao clicar no menu em questão, o usuário acessará uma página com gráficos e serviços utilizados. Caso o serviço seja *free tier eligible*, o usuário poderá ver a porcentagem de uso pelo mesmo, e monitorar os mesmos de tal forma que possa receber notificações quando um determinado valor definido pelo usuário for atingido (figura 10).

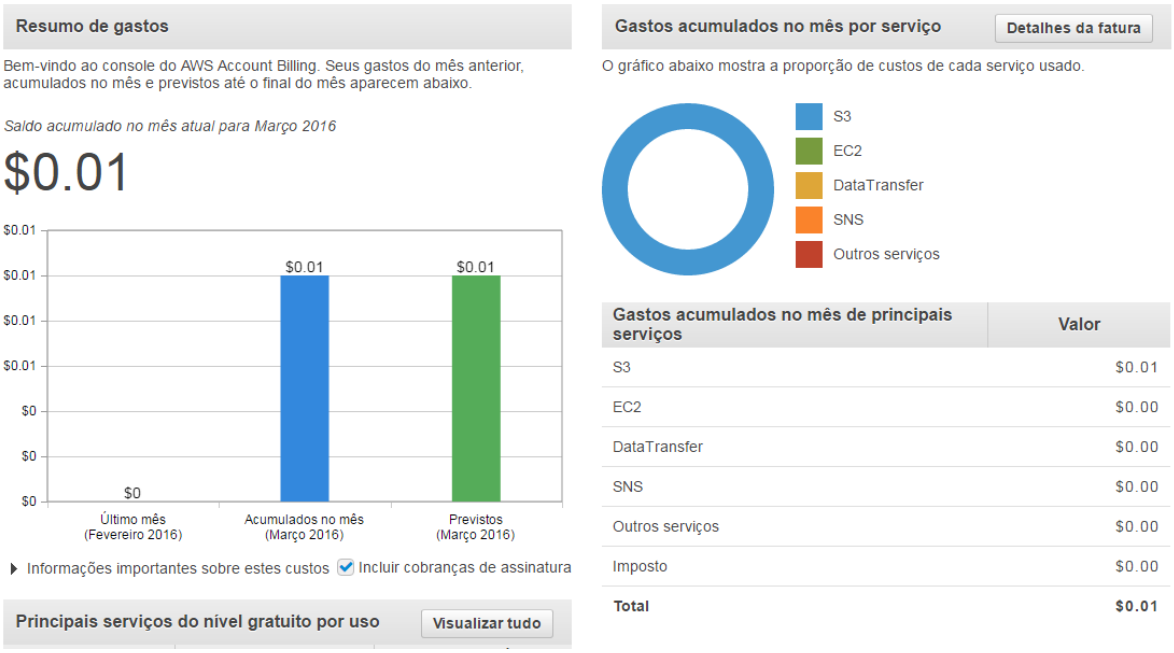


Figura 10 - Sistema de gerenciamento de custos da Amazon (AWS, 2016).

Assim, caso haja cobranças, o usuário poderá entender o motivo da mesma e evitar cobranças futuras.

5.1 FUNCIONALIDADES DA SOLUÇÃO WEB

Para que se possa começar a implantação do software, é necessário que se possa explicar quais as funcionalidades do mesmo. O sistema web, alocado em um servidor Ubuntu (Amazon EC2), é basicamente um acumulador de arquivos, como o Dropbox, o qual o usuário poderá entrar em sua conta (Amazon RDS) para acessar os arquivos colocados na nuvem (Amazon S3). Em resumo, o sistema deverá ser capaz de acessar o banco de dados do Amazon RDS, através da instância EC2, de onde os arquivos do aplicativo web estão alocados, e, através dos dados da instância (identificação), acessar a pasta referente ao próprio usuário no Amazon S3.

O sistema deverá se portar como um "mini-Dropbox", onde o usuário poderá acessar a sua conta e os arquivos respectivos à mesma. Além disso, ele poderá visualizar através do navegador o tipo e nome dos arquivos, e poderá fazer o upload dos arquivos da sua máquina através de um simples formulário. O conjunto de ferramentas LAMP será utilizado. Assim, o desenvolvimento ocorrerá em duas fases:

- Fase centralizada – todos os serviços estarão concentrados em um único local, o Amazon EC2. Serão instaladas as ferramentas LAMP na máquina virtual em nuvem do serviço, e os arquivos não dinâmicos (do cliente que usará o aplicativo), juntamente com os arquivos do sistema, serão colocados dentro da máquina virtual.
- Fase descentralizada – depois que todos os arquivos forem colocados dentro do EC2 com o conjunto de ferramentas LAMP, ocorrerá a fase de descentralização, ou seja, cada "parte" do software será realocada em outros serviços oferecidos pela Amazon, neste caso o S3 e RDS, responsáveis pelo armazenamento de arquivos não dinâmicos e pelo banco de dados, respectivamente.

Ao final, a instância EC2 deverá guardar todos os arquivos do sistema, e disponibilizá-los para acesso aos clientes, e irá se comunicar com o serviço de armazenamento de arquivos S3, e o banco de dados MySQL oferecido pelo serviço RDS. Além disso, os serviços oferecidos serão "*free tier eligible*", para demonstrar a potencialidade da Amazon, mesmo quando os serviços são gratuitos.

5.2 INSTÂNCIA EC2

A instância EC2 é responsável por ser o host, que irá armazenar os arquivos do servidor. Neste primeiro momento, uma instância será lançada, e será instalado um conjunto LAMP de softwares dentro desta instância, e assim fazer o upload dos arquivos do servidor para esta instância e configurar o mesmo para que possa disponibilizar o website para o mundo.

Para acessar a instância EC2 é necessário ter um conjunto de ferramentas para comunicação SSH, ou a utilização de softwares para upload e download de arquivos para o servidor web EC2, como o FileZilla. Estes aplicativos serão instalados ao longo do procedimento de desenvolvimento da solução web.

Vale ressaltar que o procedimento será feito em um computador com sistema operacional Windows 7 Ultimate, de 64bits.

5.2.1 Criação

Para se criar uma instância EC2, basta acessar o menu *Services*. A figura 11 lista todos os serviços que a Amazon oferece aos usuários.

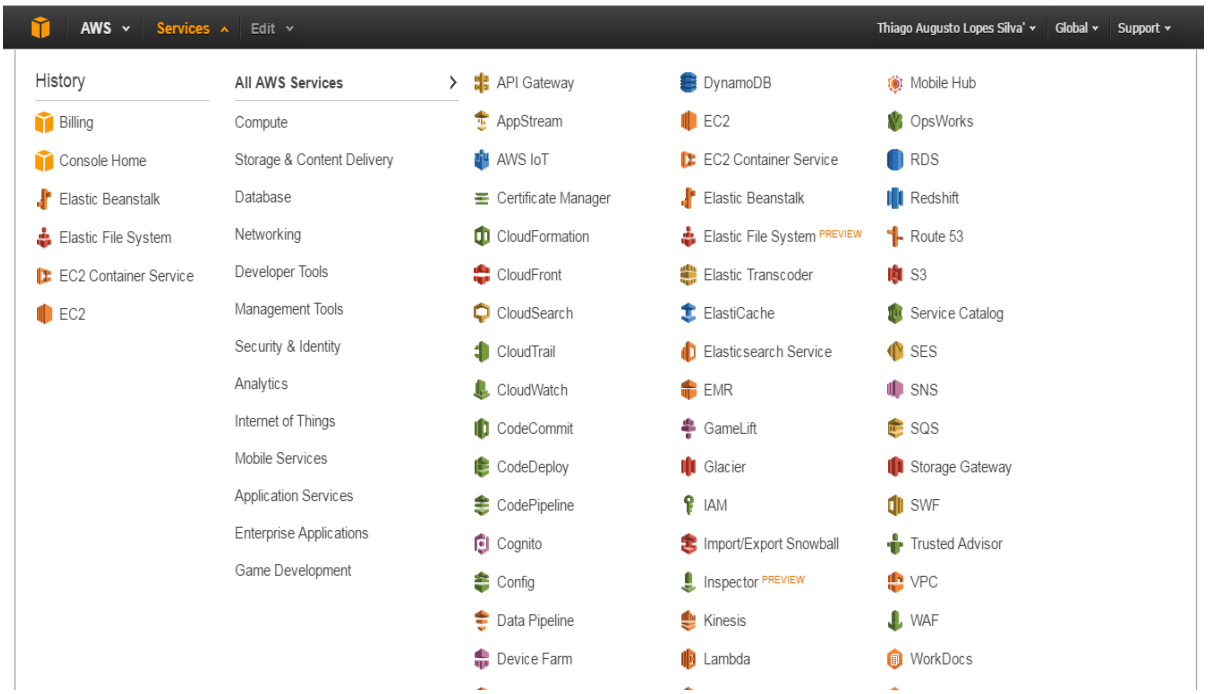


Figura 11 - Lista de serviços AWS (AWS, 2016).

Na seção *History*, o usuário visualizará os aplicativos acessados recentemente pelo mesmo. Esta seção é necessária, pois a AWS oferece diversos serviços, e o usuário poderá se perder nos aplicativos oferecidos. Na seção *All AWS Services*, o usuário poderá visualizar todos os serviços, ou utilizar as subseções para visualizar serviços das mesmas. O Amazon EC2, por exemplo, poderá ser visualizado na subseção *Compute*.

Ao clicar no serviço *Amazon EC2*, o usuário será levado para uma página que mostrará quantos grupos de seguranças e volumes criados, snapshots, entre outros. Navegando no menu lateral esquerdo pode-se ver quantas instâncias estão criadas, quantos hosts dedicados, entre outros. A figura 12 mostra a página referente ao serviço EC2.

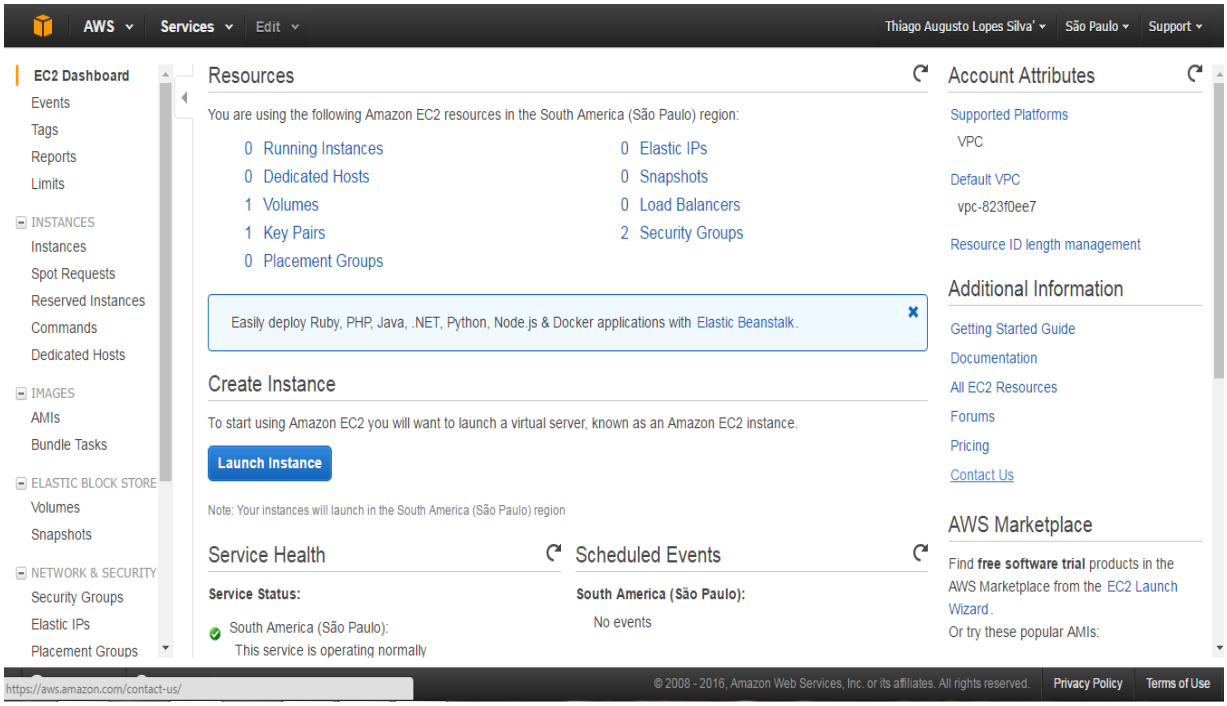


Figura 12 - Seções e subseções Amazon EC2 (AWS, 2016).

Antes de criar uma instância EC2, deve-se checar em qual região (Availability Zone) a mesma será instalada. Na foto acima, ao lado do nome do usuário, a região utilizada é São Paulo. Esta é a única região da América do Sul disponível, conhecida como “sa-east-1”. Alguns serviços não são suportados no datacenter de São Paulo, mas o usuário poderá selecionar outra zona para utilizar o serviço, mesmo que o EC2, por exemplo, esteja hospedado em São Paulo. Os serviços utilizados para a criação da solução web em questão são suportados pelo datacenter da América do Sul, e este datacenter será utilizado para o

desenvolvimento do mesmo. Depois de selecionar o datacenter correspondente, será criado uma instância no EC2.

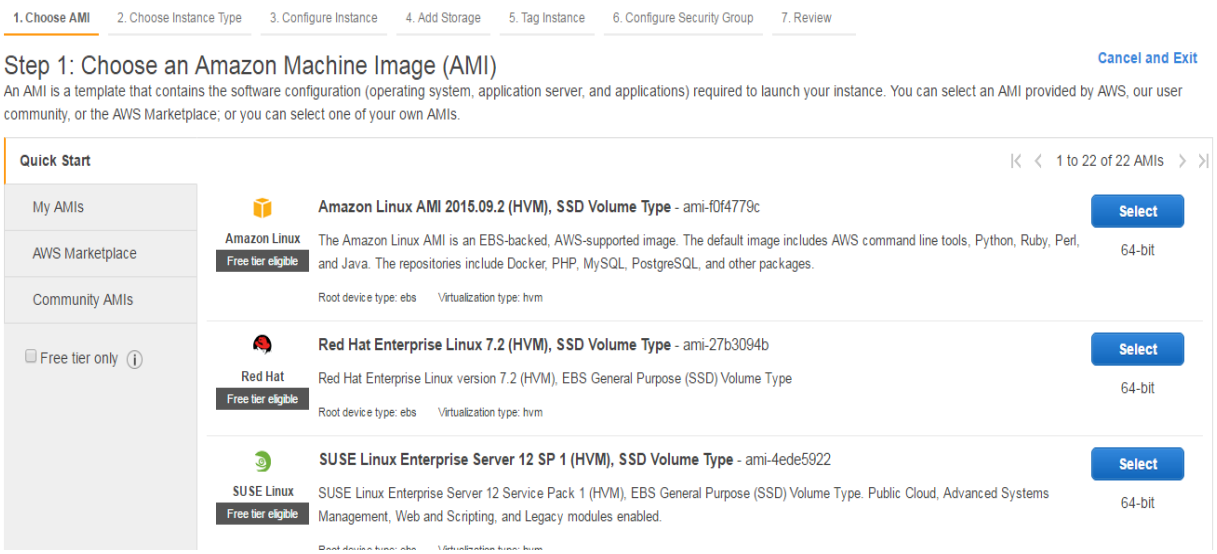


Figura 13 – Primeiro passo: Selecionar uma imagem da máquina Amazon (AWS, 2016).

Neste primeiro passo (figura 13) deve-se selecionar qual imagem o servidor deverá ter. Fazendo uma analogia, escolher a imagem do nosso servidor seria a mesma coisa de escolher um sistema operacional para um computador.

Algumas imagens não são grátis, portanto deve-se tomar muito cuidado ao selecionar a imagem que o usuário queira. Caso uma imagem seja gratuita, aparecerá *"free tier eligible"* abaixo da logo da imagem em questão. A figura 14 ilustra a instância escolhida para este trabalho.

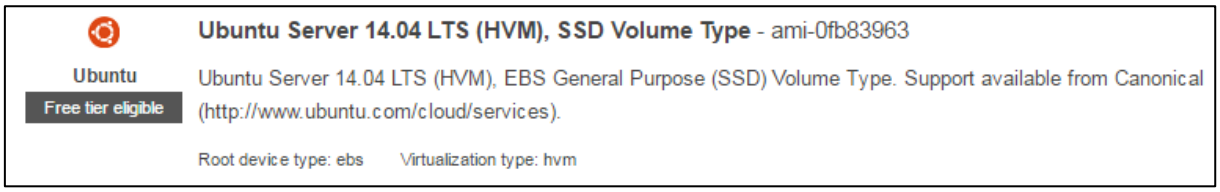
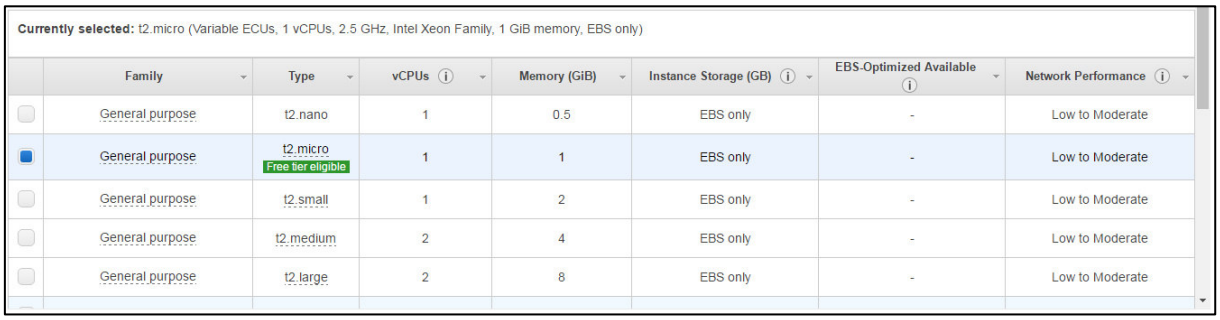


Figura 14 – AMI Ubuntu Server 14.04 (AWS, 2016).

O escolhido para a aplicação fora o Ubuntu Server pelo fato de ser distribuição Linux, código aberto e software livre, além de ser *free tier eligible*, e pela familiaridade com o sistema. Pode-se instalar, também, o LAMP stack de forma rápida através de comandos *sudo apt-get*, comandos de listagem de pastas, entre outros. Depois de escolher a imagem, deverá

se escolher o tipo de instância. Nesta seção, a instância escolhida será a t2.micro, assim como a figura 15, pois é *free tier eligible*.



Currently selected: t2.micro (Variable ECUs, 1 vCPUs, 2.5 GHz, Intel Xeon Family, 1 GiB memory, EBS only)

Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)	EBS-Optimized Available	Network Performance
General purpose	t2.nano	1	0.5	EBS only	-	Low to Moderate
General purpose	t2.micro Free tier eligible	1	1	EBS only	-	Low to Moderate
General purpose	t2.small	1	2	EBS only	-	Low to Moderate
General purpose	t2.medium	2	4	EBS only	-	Low to Moderate
General purpose	t2.large	2	8	EBS only	-	Low to Moderate

Figura 15 – Tipos de instância de EC2 (AWS, 2016).

Após a seleção do tipo de instância, é necessário configurar os detalhes da instância, como o número de instâncias, network, subnet, IP público, além de selecionar o tamanho (em GB) da instância, adicionar Tags e grupos de segurança. Ao configurar a instância, demorarão alguns minutos para que a mesma esteja pronta para ser utilizada. Vale ressaltar que se deve entender quais os limites da elegibilidade da instância grátis, para que não ocorra erros na configuração da mesma. Além disso, caso a instância seja *free tier*, é importante que se utilize a mesma apenas pelo tempo necessário, e parando a mesma quando não for utilizar a mesma.

Quando a instância for instalada, será necessário pegar o *Keypair* da mesma. Este conjunto de senhas será necessário para fazer a conexão com o EC2. Esta key só será disponibilizada para download apenas uma vez, portanto é imprescindível que ela esteja guardada de tal forma que possa ser recuperada facilmente, pois em caso de perdas a única solução viável seria fazer uma nova instância.

Para saber o estado que a instância se encontra, basta ir à *Instances* na página do EC2, e procurar por *Instance State*, como na figura 16.

Figura 16 – Estado da instância EC2 (AWS, 2016).

Para parar a instância, deve-se marcar a mesma, ir em *Actions*/*Instance State*/*Stop*. Isto fará com que a instância fique inutilizável pelo período que estiver parada, mas não a deletará do sistema. Para voltar é necessário apenas repetir o mesmo processo, mas selecionando Start. Para acabar com a instância, caso a mesma não seja mais necessária, basta

escolher a opção *Terminate*, que irá deletar a instância e tudo que foi instalado nela. Estas e mais opções são mostradas na figura 17.

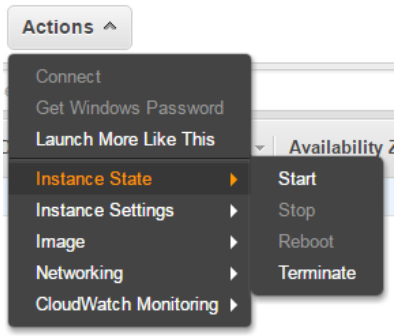


Figura 17 – Possíveis estados da instância (AWS, 2016).

Além disso, quando se selecionar a instância, aparecerá os dados da instância. Alguns dados são necessários para se conectar com a instância como, por exemplo, o Public DNS, Public IP, Private IPs (caso se utilize VPCs diferentes para cada serviço).

Existem duas maneiras de se conectar ao servidor criado: via SSH utilizando o prompt de comando Cygwin, e utilizando programas que possam utilizar conexão SFTP, como o FileZilla. Serão utilizados ambos os tipos, um para cada propósito. O primeiro é necessário para que se instale o conjunto de softwares LAMP, e o segundo para enviar os arquivos do aplicativo web para o servidor sem precisar utilizar linhas de comando do prompt.

5.2.2 Utilizando o Cygwin para conexão SSH

O Cygwin é um software que cria um meio ambiente Unix-Like e uma interface de comando para o Windows, além de prover integração nativa para os aplicativos baseados em Windows, dados e outros recursos, além de várias ferramentas.

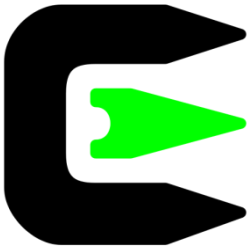


Figura 18 – Logo Cygwin (Wikipedia, 2016).

Neste caso, a instalação do Cygwin, com a logo destacada na figura 18, é necessária para se utilizar o Open SSL, que será responsável por estabelecer a conexão com a

instância EC2 e instalar os recursos necessários para o desenvolvimento da aplicação (Linux, Apache, MySQL/MariaDB, PHP/Python). A instalação do Cygwin deve ser personalizada, pois o Open SSL algumas vezes não está marcado para a instalação. Portanto, uma instalação personalizada é necessária.

Uma vez concluída a instalação, conexão do computador com a instância EC2 pode ser estabelecida. Para tal, deve-se abrir o Cygwin e levá-lo para a pasta do *Keypair* respectivo. Ao acessar a pasta que se encontra o mesmo, basta utilizar o comando de conexão SSH abaixo, e também na figura 19:

ssh –i <nome_do_key> <usuário_do_AMI_correspondente>@<ip_do_servidor_EC2>

Onde,

- <nome_do_key> - Nome do *Keypair* baixado na criação da instância
- <usuário_do_AMI_correspondente> - Todo AMI tem um usuário correspondente, deve-se ver na instalação do mesmo qual será o usuário. Mais informações sobre o usuário podem ser obtidas na documentação da AWS. No meu caso, o usuário se chama *ubuntu,* pois o servidor é Ubuntu Server.
- <ip_do_servidor_EC2> - Este IP é encontrado nos dados do servidor criado, basta entrar na página das instâncias criadas e procurar pelo *Public IP ou Public DNS*. Neste caso utilizarei o IP público.

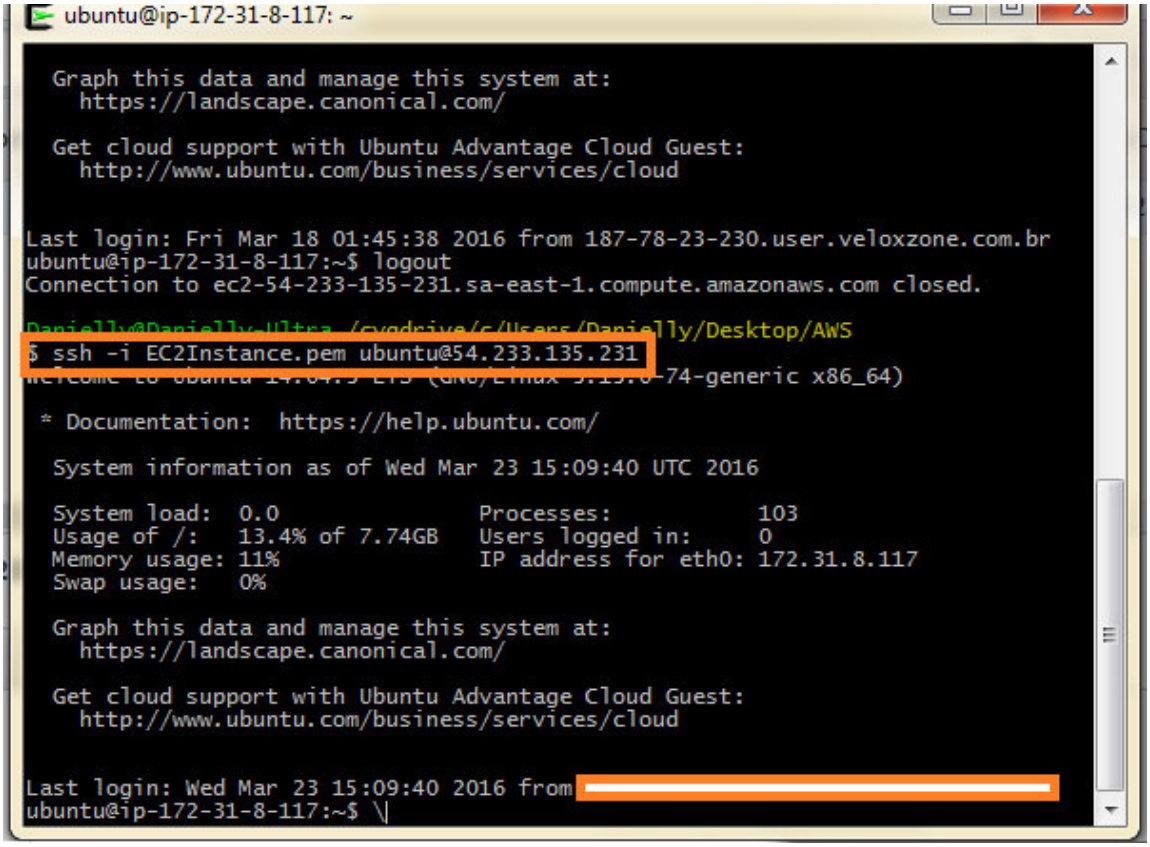


Figura 19 – Conexão com a Instância EC2 – Terminal do Cygwin.

Caso haja problemas na conexão em relação à permissão não concedida, é necessário mudar as permissões de usuário do *Keypair*, pode-se utilizar o próprio terminal do Cygwin para mudar tais permissões.

Uma vez conectado, deve-se instalar os softwares do conjunto LAMP com comandos de administrador, ou super usuário. Utilizando o terminal do Cygwin, basta escrever os comandos de *sudo apt-get install* para instalar os programas necessários:

sudo apt-get install apache2 libapache2-mod-php5 mysql-server php5-mysql php5

Ao entrar com o comando acima, o servidor pedirá que se entre com uma senha para o usuário "root". Ao concluir a instalação, é aconselhável recomeçar o aplicativo utilizando o comando sudo *service apache2 restart.* Pode-se também utilizar o comando *mysql_secure_installation* para mudar senhas do usuário e remover usuários conectados, entre outros.

Talvez seja necessário atualizar para que se possa instalar os softwares LAMP. Neste caso, escreva no console o comando *sudo apt-get update*.

Para acessar os arquivos do site hospedado no servidor, ao terminar a instalação dos softwares, basta acessar a pasta */var/www/html* do servidor. O terminal também pode ser utilizado para acessar o banco de dados MySQL instalado:

mysql –u <nome_do_usuário_mysql> -p

Logo após o comando, o servidor pedirá para que se escreva a senha criada na instalação dos softwares. Ao acessar o MySQL, pode-se utilizar os comandos SQL para o gerenciamento do mesmo. O gerenciamento desses bancos podem ser feitos de duas maneiras: pelo prompt de comando e/ou utilizando o *phpmyadmin*. Para utilizar o *phpmyadmin* basta instalá-lo utilizando o *sudo apt-get phpmyadmin*. Importante ressaltar que os usuários poderão acessar o site através do *Public DNS,* que se encontra nos dados do servidor EC2 no site da AWS, ilustrado na figura 20.

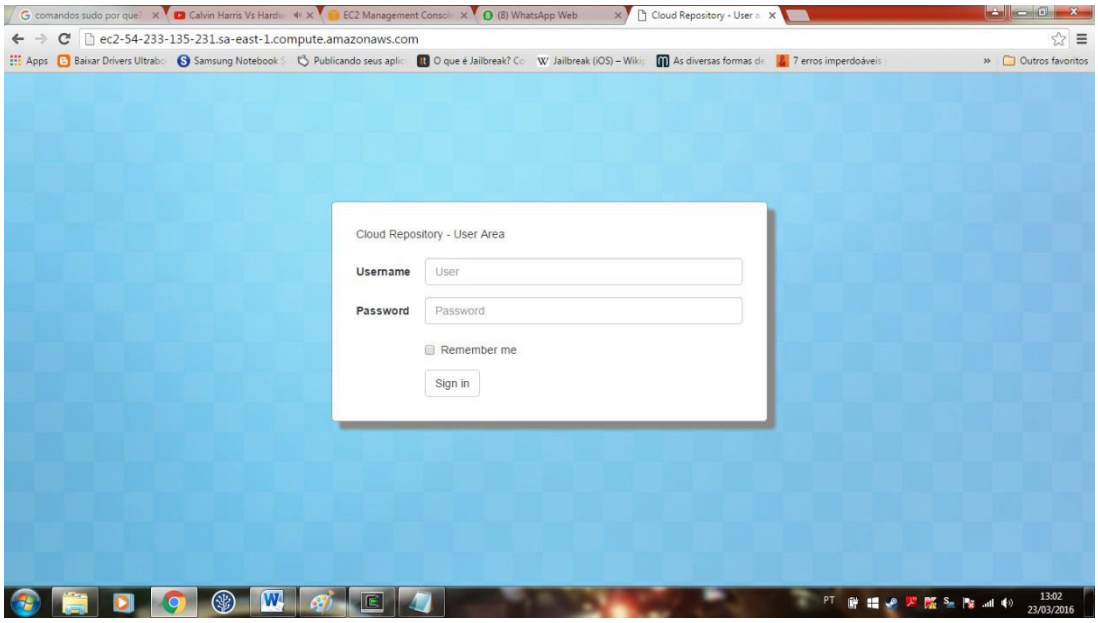


Figura 20 – Aplicativo web acessado através do Public DNS

5.2.3 Utilizando o FileZilla para acessar os arquivos do servidor

Outra maneira de acessar os arquivos do servidor é utilizando um programa que estabeleça a conexão SFTP (SSH). Neste caso utilizarei o FileZilla para estabelecer a conexão. Basta baixar e instalar o programa para começar a sua utilização. A figura 21 destaca a logo do aplicativo.

Figura 21 – Logo Filezilla (Google, 2016).

Para conectar ao servidor EC2 através do FileZilla, é necessário que se tenha o Public DNS, o tipo de protocolo SFTP, o usuário e a senha, se ela existir, como na figura 22. Ao conectar-se, o usuário poderá enviar os arquivos do site para o servidor.

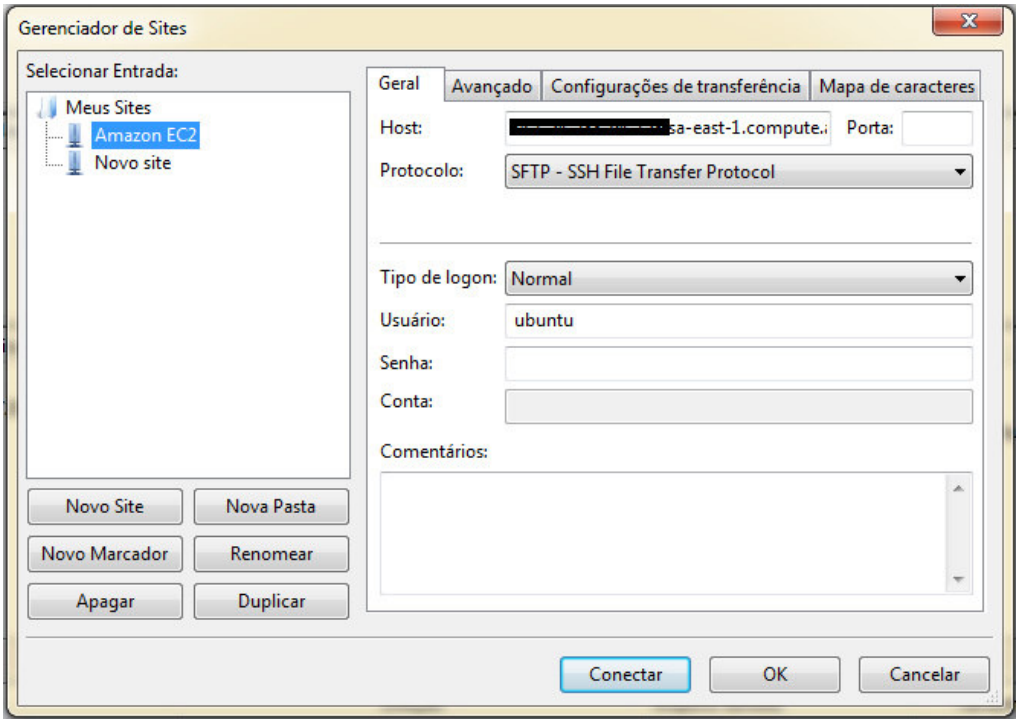


Figura 22 – Gerenciador de sites com os dados do EC2.

Apesar do Amazon EC2 apresentar todos os componentes necessários para a criação de um aplicativo web, pois já apresenta hospedagem de arquivos, banco de dados, e a apresentação desses arquivos na internet, ainda se faz necessário a utilização de outros dois serviços neste caso: o Amazon Simple Storage Service (S3) e o Amazon Relational Database Service (RDS). Estes serviços são necessários para que a empresa possa separar os serviços e designar os grupos de gerenciamentos para cada um dos serviços. Em resumo, pode-se designar um grupo para gerenciar o armazenamento de arquivos estáticos (fotos, imagens, arquivos de texto) com o Amazon S3, um para gerenciar os arquivos do EC2, e para gerenciar o banco de dados.

Assim cada grupo será responsável por uma parte do aplicativo. Além disso, há a questão segurança, pois através do VPC responsável pelo Amazon RDS pode-se colocar o acesso do mesmo apenas pelo servidor EC2, utilizando o IP Estático do mesmo. Neste caso, apenas o EC2 poderá requisitar os serviços do RDS, impedindo que usuários quaisquer, mesmo que da própria empresa, acessem indevidamente o serviço que não lhe compete.

Outro fator importante se dá pelo *free tier eligible*, pois pode-se combinar um conjunto de serviços grátis para fazer o aplicativo web. Assim não será necessário se preocupar com os 8GB apenas do Amazon EC2, pois os S3 e RDS também têm os seus próprios limites de armazenamento, bastando apenas checar quais são estes limites.

5.3 ARMAZENAMENTO DE ARQUIVOS NÃO DINÂMICOS NO AMAZON S3

Como explicado acima, a divisão do software em serviços diferentes influencia em três aspectos: custo, segurança e gerenciamento. Portanto a utilização do Amazon S3 para o armazenamento de arquivos se faz necessário.

5.3.1 Criando um bucket

Para se armazenar arquivos no Amazon Simple Storage Service é necessário se criar um *bucket*, que em português significa “balde”. Os “baldes” são necessários para separar arquivos de um determinado aplicativo para outro, ou se a pessoa quiser manter baldes que armazenem apenas um tipo de arquivo cada. São várias as aplicações que se pode fazer utilizando *buckets* diferentes. Dentro de cada *bucket* pode se criar pastas e subpastas ao critério do usuário contratante. O serviço é basicamente um gerenciador de arquivos em nuvem, onde o usuário pode fazer o download e upload, e até visualizar os arquivos armazenados em nuvem.

Para criar um *bucket*, basta acessar o serviço e clicar em *Create Bucket*, inserir o nome da região do mesmo. A região a ser escolhida deverá ser a mais próxima possível da sua localidade, ou da localidade dos seus clientes, para diminuir a latência, diminuir custos da máquina, ou até mesmo para regulamentação de endereços. Na figura 23, São Paulo foi a região escolhida para o *bucket*.

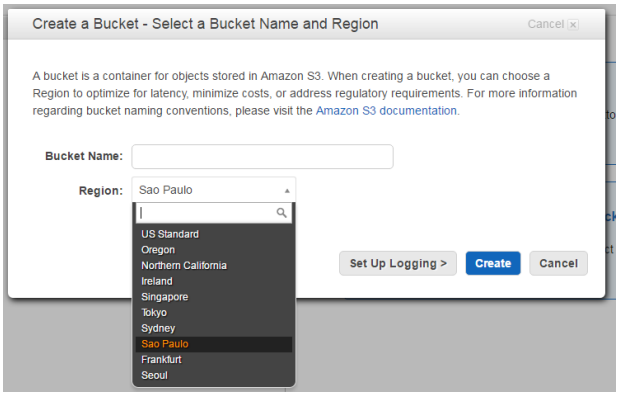


Figura 23 – Formulário de criação do bucket (AWS, 2016).

Após a criação do *bucket*, o usuário poderá selecionar o mesmo e visualizar os arquivos salvos, como na figura 24.

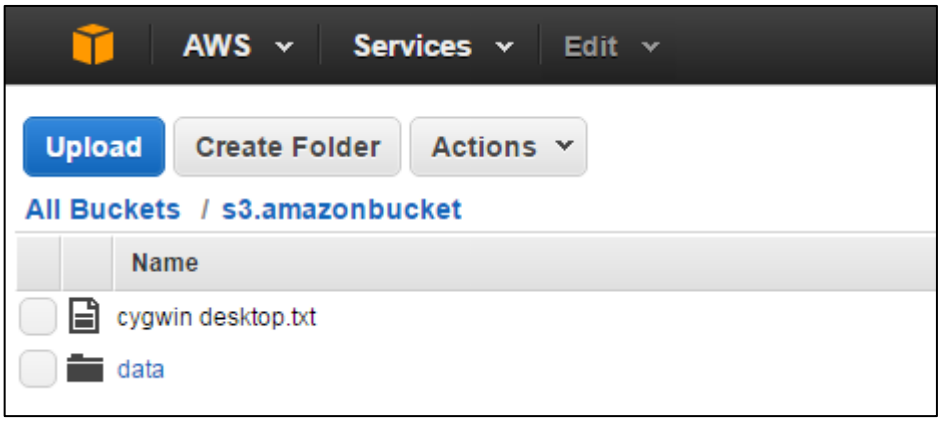


Figura 24 – Arquivos armazenados dentro do bucket (AWS, 2016).

Agora que o *bucket* já foi criado e alguns arquivos foram inseridos dentro do mesmo, obedecendo ao sistema de arquivos necessário para a aplicação web, é necessário que se faça a comunicação entre a mesma e o Amazon S3. Para que isso ocorra, é necessário a utilização de um kit de desenvolvimento da Amazon, conhecido como Amazon SDK.

5.3.2 Instalando o Amazon SDK for PHP

O Amazon SDK é uma biblioteca de código aberto para o desenvolvimento de aplicações que queiram utilizar um ou mais serviços da prestadora, para diversas linguagens, como PHP e Python. Ele é necessário para fazer a comunicação entre os serviços através do próprio aplicativo web, como o S3.

Existem três maneiras para instalar o Amazon SDK for PHP na aplicação web: via composer, phar e através de um arquivo compactado no formato ZIP. A diferença entre estes seria apenas o método de instalação e o arquivo a ser referenciado no código. No caso do arquivo compactado, a referência se dá através do documento *'aws-autoloader.php'*. Basta baixar o arquivo Zip no site da Amazon e extraí-lo em uma pasta dentro do diretório do aplicativo.

5.3.3. Utilizando o SDK para acessar arquivos no S3

Agora que o Amazon SDK foi extraído e colocado na pasta do aplicativo, basta referenciar o mesmo no código PHP que utilizar o mesmo, através da função *require*. O uso do SDK é feito através de instâncias de um objeto do cliente para o serviço da Amazon que se queira interagir, e estes objetos possuem métodos que correspondem às operações com a API. Neste caso precisa-se estabelecer uma comunicação com o Amazon S3, então o cliente utilizado é o *S3Client*. O método chamado irá retornar um objeto Array, ou lança uma exceção em caso de erros.

Para criar um cliente, basta passar um Array associativo de opções para o construtor de clientes. Será necessária a utilização de credenciais para que o desenvolvedor possa acessar o serviço, por questões de segurança, ilustrados na figura 25.

```php
<?php
// Include the SDK using the Composer autoloader
require 'vendor/autoload.php';

$s3 = new Aws\S3\S3Client([
    'version' => 'latest',
    'region'  => 'us-east-1'
]);
```

Figura 25 – Código PHP para a instanciação do cliente S3 (AWS Documents, 2016).

Neste exemplo percebe-se que não há credenciais passadas para o construtor. Por questões de segurança, as credenciais são necessárias para que o desenvolvedor possa acessar o serviço. Elas devem ser detectadas pelo SDK utilizando variáveis de ambiente do Windows, por um arquivo .ini no diretório HOME, AWS IAM (Identity and Access Management) ou

provedores de credenciais. O aplicativo web utilizou as credenciais codificadas, ou seja, que estão presentes no próprio código PHP.

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

//S3
$s3 = new S3Client(array(
    'region' => 'sa-east-1',
    'version' => 'latest',
    'ssl.certificate_authority' => '../cacert.pem',
    'scheme' => 'http',
    'credentials' => [
        'key' => 'AKIAJMXEL6I4MGRFCU3Q',
        'secret' => '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',
    ]
));
```

Figura 26 – Código para instanciação de um cliente S3 utilizando credenciais codificadas.

O código acima mostra a instanciação de um cliente para o Amazon S3. Os parâmetros passados para o construtor são:

- region – Região a qual o cliente irá atuar;
- version – Versão a qual o cliente irá utilizar. Neste caso está sendo utilizada a versão mais atual.
- ssl.certificate_authority – tipo de certificado que será usado para a validação das credenciais. Este certificado poderá ser baixado através da internet e referenciado nesta tag.
- scheme – Esquema utilizado para a criação do cliente S3. Como o aplicativo utilizado será hospedado na web, o esquema será HTTP.
- credentials – Array que passará as credenciais do usuário. As credenciais são divididas em chave e segredo.

Para criar as credenciais basta acessar o site da Amazon, entrar com o usuário e senha, e acessar o Security Credentials clicando no nome do usuário do lado superior direito do site, depois acessar a subseção *Users*, criar um novo usuário e pegar os dados (Access Key ID e Security), pois são necessários para se colocar na instanciação do objeto cliente no S3. Neste caso, criou-se um usuário com poderes de administrador para melhor entendimento.

Depois de pegar a chave e o segredo, colocam-se os dados correspondentes no código PHP, e quando o mesmo for utilizado, as credenciais serão comparadas com as que

foram criadas na conta da AWS, como na figura 5.18. Lembrando que é necessário a utilização do *cacert.pem*, um modelo de certificados para que o SDK possa ser utilizado.

Depois da inserção dos dados e, conferido os mesmos, o cliente S3 será criado, e com ele vários procedimentos que podem ser utilizados apenas chamando-os através do código PHP. O aplicativo web em questão tem como requisitos: fazer o upload do arquivo no *bucket* em questão, através de um formulário no próprio aplicativo, e mostrar para o usuário quais são estes arquivos. Para estes dois casos, existem dois procedimentos que podem ser utilizados: *getIterator* e *putObject*. O *getIterator* é uma função que irá retornar um *array* de objetos do tipo arquivo, que serão postos em um *foreach* para acessar cada objeto separadamente, e dispostos na página web. Já o *putObject* é uma função para colocar o arquivo no *bucket* que desejar, desde que este tenha sido criado pelo usuário que criou a chave e segredo utilizados para a conexão.

O aplicativo web deve mostrar todos os arquivos daquele usuário, assim a função *getIterator* ficará da seguinte maneira:

```
$objects = $s3->getIterator('ListObjects', [
    'Bucket' => 's3.amazonbucket',
]);
//var_dump($objects);
```

Figura 27 – Procedimento getIterator.

Percebe-se que o procedimento deverá ser chamado a partir da variável que foi utilizada para a instanciação do objeto cliente, neste caso a variável será *$s3* (figura 27). Além disso, deve-se colocar o *bucket* aos quais os arquivos serão mostrados. Este código deve ser colocado no início do site, pois o mesmo será mostrado no corpo da página através de uma função *foreach*, como a abaixo:

```
foreach($objects as $object){
```

Figura 28 – Função foreach.

Neste laço de repetição utiliza-se a variável *object*, na figura 28, para receber cada objeto separadamente do iterador da figura 5.19, e disponibilizá-lo em um código HTML, através da função *echo*, na figura 5.21.

```
echo "<div style='width:20%;float:left;'><img src='img/icons/".$fileTypeImg."'
style='width:70%;'><br>...".substr($object['Key'], -12, -1)."</div>";
```

Figura 29 – Função echo.

Já para fazer o upload de arquivos para o S3, deve-se saber qual a pasta em que o mesmo está na máquina do cliente. Neste caso faz-se necessária a importação deste para uma pasta temporária dentro do aplicativo web, na pasta *temp*. Para tanto, cria-se um formulário em HTML para que o usuário possa selecionar o arquivo, e assim o servidor começar a importar o arquivo para a pasta temporária, explicitado no código da figura 30.

```
if(isset($_FILES['file'])){
    $files = $_FILES['file'];
    // Detalhes do arquivo
    $FileName = $files['name'];
    $FileTmpName = $files['tmp_name'];
    //var_dump($files);
    $FileExtensionName = explode('.',$FileName);
    $FileExtensionName = strtolower(end($FileExtensionName));
    //var_dump($FileExtensionName);

    // Detalhes do arquivo temporário
    $FileKey = md5(uniqid());
    $FileTmpName2 = "{$FileKey}.{$FileExtensionName}";
    $FileTmpPath = "tmp/{$FileTmpName2}";
    //var_dump($FileTmpPath);
    //Moving the File
    move_uploaded_file($FileTmpName, $FileTmpPath);
```

Figura 30 – Código PHP que move o arquivo para uma pasta temporária.

Como se pode ver na figura acima, o código é responsável por pegar o nome temporário do arquivo e coloca-lo dentro da pasta *tmp* com a função *move_uploaded_file*. Depois de mover o arquivo para a pasta temporária, o aplicativo deverá mover este para o *bucket* em questão através do procedimento *putObject* através do método try-catch, de acordo com a figura 31. Depois do procedimento, o arquivo temporário é apagado, para que não consuma memória do servidor EC2, responsável por guardar os arquivos do aplicativo.

```
try{
    $s3->putObject([
       'Bucket' => 's3.amazonbucket',
       'Key' => "data/{$_SESSION['tb_id']}/{$FileName}",
       'Body' => fopen($FileTmpPath, 'r'),
       'ACL' => 'public-read',
    ]);
   gc_collect_cycles();
    //Removendo o arquivo
    unlink($FileTmpPath);
    header("Location:index.php?msg=upload");
}catch(S3Exception $e){
    var_dump($e);
}
```

Figura 31 – Método para inserção de arquivos no bucket do S3.

No procedimento *putObject* deve-se colocar:

- O *bucket* em questão;
- A *Key*, que determina o local e o nome do arquivo a ser enviado. Neste caso ele está sendo colocado em "*data/<id_usuário>/<nome_arquivo>.<extensão>*".
- *Body*, que determina qual o conteúdo do arquivo. Neste caso o arquivo estará sendo aberto, o que significa que o que estiver dentro do arquivo será o conteúdo dentro do mesmo.
- ACL, que determina quais são as permissões que o usuário terá. Neste caso o arquivo é parametrizado como *public-read*, ou seja, o arquivo pode ser lido pelo público.

Enquanto o arquivo temporário não for enviado para o *bucket* por completo, o usuário não poderá deletar este. Para tanto usa-se a função *gc_collect_cycles*. Esta função é responsável por fazer com que o código pare de ser lido e executado até que um determinado ciclo esteja completo. Logo após o ciclo estar completo, a função *unlink* será responsável por deletar o arquivo temporário do Amazon EC2. Caso ocorram erros, uma exceção é lançada através do *catch*. O *S3Exception* é um tipo de exceção lançado para este fim, e deve ser referenciado no começo do código através de um *use*, mostrado na figura 32.

```
use Aws\S3\Exception\S3Exception;
```

Figura 32 – Inclusão do pacote de exceções do S3.

Agora que o aplicativo web já pode acessar, disponibilizar e fazer o upload de documentos no *bucket* do S3, o próximo passo é utilizar o Amazon *Relational Database* System para criar um banco de dados relacional e fazer com que o mesmo possa se comunicar com o Amazon EC2.

5.4 UTILIZANDO BANCO DE DADOS ATRAVÉS DO AMAZON RDS

Antes de utilizar o Amazon RDS, o banco de dados MySQL está sendo utilizado pela própria instância EC2, quando fora instalado o pacote LAMP via linha de comando do servidor Linux. O principal problema de se utilizar o conjunto LAMP na própria instância é a falta de espaço, pois o usuário seleciona o valor, em GB, do quanto a instância poderá comportar de arquivos em geral. O ideal, neste caso, será utilizar o banco de dados em outro serviço, e estabelecer a comunicação entre estes.

O Amazon RDS oferece vários tipos de banco de dados, dentre estes o MySQL, MariaDB, Postgre SQL e Oracle. Neste caso o MySQL será utilizado, pela familiaridade com o mesmo, e por haver desenvolvido o aplicativo de forma que o mesmo se comunicasse com o banco de dados do servidor local que utiliza o MySQL como banco (*wampserver*), e por estar instalado na própria instância do EC2.

5.4.1 Criando uma instância de banco de dados MySQL

Para se criar uma instância, deve-se acessar a AWS e procurar pelo serviço RDS, que fica na subseção *Database*. Ao clicar em *Instances*, pode-se saber quais são as instâncias que o usuário tem na sua conta, e logo depois em *Create Instance*.

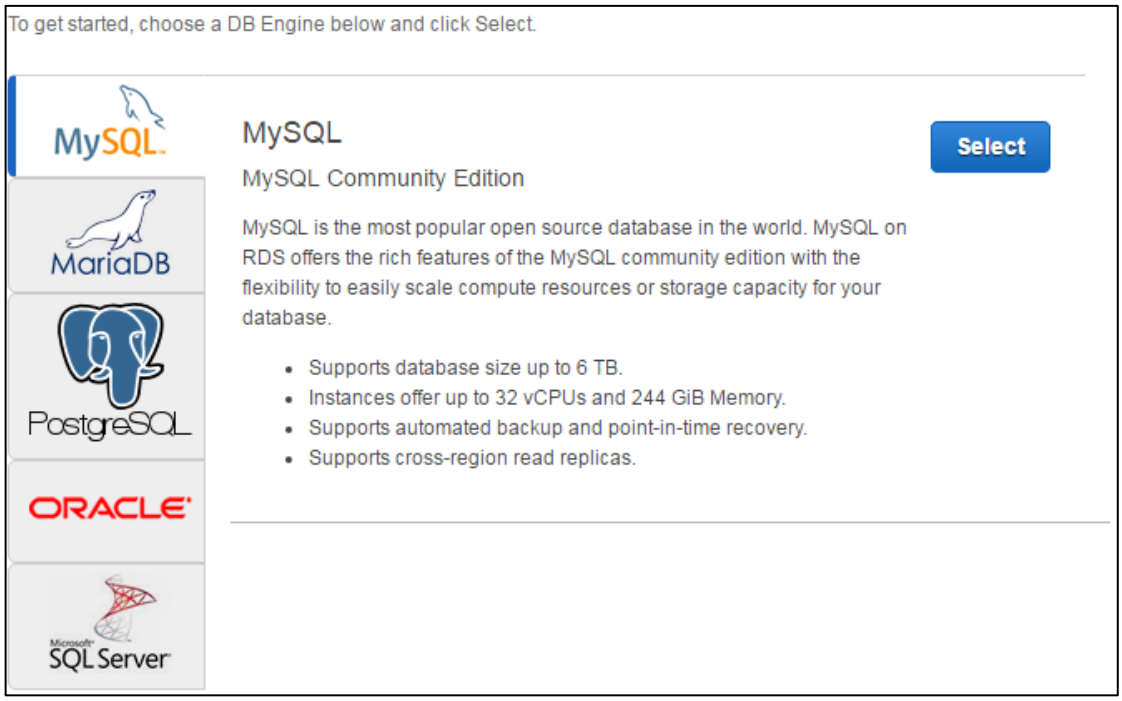


Figura 33 – Bancos de dados suportados pelo Amazon RDS (AWS, 2016).

A figura 33 mostra quais são os bancos de dados oferecidos pelo RDS na região da América do Sul (São Paulo). O banco de dados Aurora não é suportado pela região escolhida, mas é suportado por outras regiões, como as regiões dos Estados Unidos, entre outros. O banco de dados selecionado será o MySQL, por motivos mencionados anteriormente.

Uma pergunta aparecerá para o usuário, que deverá escolher o motivo para a criação do banco de dados. Se o usuário quiser utilizar o *free tier*, deverá selecionar a opção *Dev/Test*. Depois, o mesmo deverá configurar o banco de dados, podendo selecionar apenas opções de *free tier* ou não, dependendo do propósito, e escolher o limite de tamanho do banco, em GB. O usuário pode ainda selecionar a opção "*Only show options that are eligible for RDS Free Tier*" para que ele configure apenas as opções que podem ser utilizadas pelo plano grátis. Vale ressaltar que o Multi-AZ será cobrado caso seja utilizado. Por fim, deve-se guardar os dados do banco em relação ao usuário e senha. Estes dados são:

- DB Instance Identifier – nome do banco;
- Master Username – usuário mestre, utilizado para o acesso ao banco via PHP ou linha de comando;
- Master Password – senha mestre, utilizado para o acesso.

Criada a instância, deve-se acessá-la para criar as tabelas do aplicativo web.

5.4.2 Acessando a instância MySQL

Existem algumas maneiras de acessar o banco de dados criado, dentre as ferramentas há o MySQL Workbench e Cache Monitor, além de poder acessar através da própria instância EC2 criada anteriormente, pois a mesma é *Ubuntu Server* 14.04.

Neste caso utilizou-se a instância do EC2, a qual é Linux, para acessar o Amazon RDS. No primeiro momento, deve-se fazer os passos para acessar a instância EC2. Ao fazer isso, basta digitar na linha de comando:

mysql -h <hostname> -u <rdsusername> -p

Onde:

- *<hostname>* - Necessário para identificar onde está o banco, pois o mesmo não se encontra no Amazon EC2. Neste caso é o Endpoint, que pode ser adquirido ao visualizar a instância do RDS criado.
- *<rdsusername>* - Nome do usuário mestre da instância criada.

Depois de entrar com o comando, o terminal irá pedir a senha, que deverá ser a senha mestre criada. Caso a senha digitada conferir com a criada, será acessada a instância MySQL. Para visualizar ou fazer qualquer alteração na instância, comandos SQL serão necessários. Visualizar as bases de dados, por exemplo, requer o comando "*show databases;*" na figura 34.

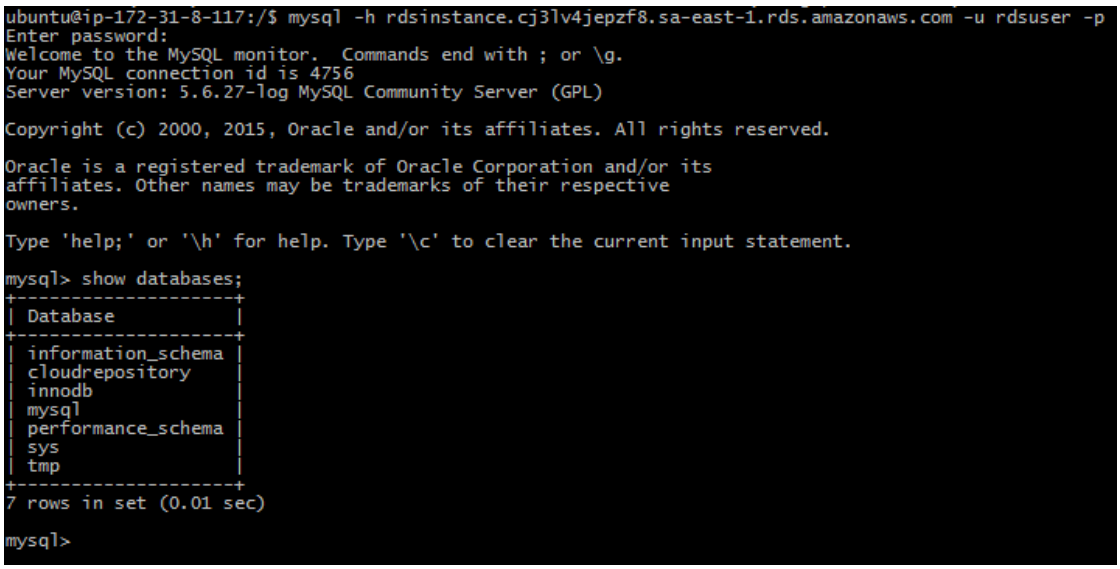


Figura 34 – Comando "show databases;"

Assim, o usuário pode criar as tabelas do banco de dados. Existem algumas maneiras de se fazer isso. A primeira é importar do banco de dados MySQL da instância EC2 ou do próprio computador. A segunda é escrever o comando SQL para criar as mesmas.

Como o software proposto foi desenvolvido em servidor local, e depois instalado em uma instância EC2 com o LAMP Stack, vale a pena exportar as tabelas do aplicativo criado localmente, copiar o código e simplesmente colar na linha de comando, visto que a exportação das tabelas MySQL resulta em um conjunto de comandos MySQL, e há poucos dado inseridos nas tabelas criadas. Quando há muitos dados e muitas tabelas, o melhor a se fazer é importar o próprio banco de dados através de linhas de comando, quando se acessa o mesmo através do comando de acesso ao Amazon RDS.

Ao terminar com estes comandos, basta digitar o comando "*exit;*" para sair da instância. Uma mensagem de despedida irá aparecer no console, e o usuário voltará para a instância EC2, a qual fora utilizada para acessar o banco.

5.4.3 Modificando o código para acessar o Amazon RDS

Agora que a instância fora criada, e a base de dados e tabelas foram inseridas com sucesso, basta modificar o código PHP para que o mesmo possa acessar a instância RDS e fazer todos os comandos pertinentes ao MySQL: visualizar, inserir, alterar e remover.

O código para acessar o banco de dados localmente fora desenvolvido utilizando PDO, um conjunto de funções do próprio PHP para acessar bancos de forma mais segura, pois o usuário poderá preparar os comandos SQL antes de executá-los de qualquer forma a fim de evitar SQL Injections.

Criou-se uma classe chamada *Connection* para que os outros arquivos apenas chamassem a mesma para criar uma conexão com banco de dados, além de ter uma função para conferir se o usuário e senha estão corretos, e retornar os dados do usuário em caso de sucesso, e retorna falso caso contrário.

Para fazer o software reconhecer e utilizar a instância RDS, deve-se modificar alguns dados para o estabelecimento da conexão: o host, usuário, senha e o nome do banco. Na prática, estes dados são basicamente os mesmos utilizados para a conexão via linha de comando, então se pode deduzir que o host é o *Endpoint* encontrado nos dados da instância, o usuário e senha são os mestres, e o nome do banco fora o mesmo importado da instância EC2 ou servidor local. A figura 5.27 mostra as variáveis de conexão utilizadas para que se possa conectar com o Amazon RDS.

```php
<?php
    class Connection{
        private $DB_HOST=[redacted].sa-east-1.rds.amazonaws.com';
        private $DB_USER = 'rdsuser';
        private $DB_NAME = 'cloudrepository';
        private $DB_PASSWORD = 'rdspassword';
```

Figura 35 – Variáveis de conexão PDO com o Amazon RDS.

Ao modificar os dados necessários e enviá-los para o Amazon EC2, o software estará pronto para uso.

CONCLUSÃO

Neste trabalho estudou-se o que é computação em nuvem e como ela pode ser utilizada para a criação de soluções web a partir de um conjunto de serviços oferecidos pelo provedor de nuvem AWS, da Amazon.

Ao final, o aplicativo deverá ser capaz de acessar o banco de dados RDS para validação de usuário e senha, bem como acessar a pasta do indivíduo, de onde o mesmo poderá visualizar os nomes e tipos de arquivos guardados no Amazon S3, além de poder fazer download e upload do mesmo através do *bucket*. O site deverá ser disponibilizado para todo o mundo, e ser acessado via browser. Além disso, todos os serviços utilizados deverão resultar no menor custo possível para a aplicação.

A utilização de serviços de computação em nuvem da Amazon (AWS) se mostrou bastante competente e viável de acordo com a necessidade de desenvolvimento da solução proposta. Conclui-se, assim, que o uso dos serviços da Amazon proporcionam melhorias no custo e gerenciamento doservidor. Além disso, a mesma oferece outros recursos que podem ser explorados de acordo com o projeto de desenvolvimento.

Para trabalhos futuros, diversos serviços podem ser acrescentados para o aperfeiçoamento do software e estudo de nuvem da Amazon, como o Route 53, um serviço de domínio utilizado para a instância EC2, uma vez que o DNS deste é mutável. Além disso, pode-se analisar o uso do *Internet of Things* (IOT), Amazon Cloudfront (serviço para *streaming* de vídeos), entre outros. Portanto, o desenvolvedor pode escolher as peças para montar o software de acordo com a sua necessidade, pagando somente o que for utilizado.

REFERÊNCIAS BIBLIOGRÁFICAS

ALBUQUERQUE, Fernando. *Banco de Dados.* Universidade de Brasília (UNB). Disponível em: <http://cic.unb.br/~fernando/matdidatico/apostilas/resumo/bdadossp.pdf> . Acesso em 10 fev. 2016.

ALECRIM, Emerson. *O que é cloud computing (computação nas nuvens)?* Info Wester. Disponível em <http://www.infowester.com/cloudcomputing.php> Acesso em 4 mar. 2016

AMAZON Web Services (AWS). Disponível em: <https://aws.amazon.com/pt/about-aws/> Acesso em: 10 jan. 2016

AMOROSO, Danilo. *O que é Computação em Nuvens?.*, TecMundo. Disponível em <http://www.tecmundo.com.br/computacao-em-nuvem/738-o-que-e-computacao-em-nuvens-.htm> Acesso em 4 mar. 2016.

Apache HTTP Server Project. Disponível em: <https://httpd.apache.org/> Acesso em 8 fev. 2016.

ARMBRUST, Michael, *et al*. *Above the Clouds:* A Berkeley View of Cloud Computing. Disponível em: <http://d1smfj0g31qzek.cloudfront.net/abovetheclouds.pdf>. Acesso em: 3 mar 2016.

AUGUSTO, Henrique. *SaaS, PaaS e IaaS* – Os serviços de computação em nuvem. Qi Network. Disponível em: <http://www.qinetwork.com.br/saas-paas-iaas-os-servicos-de-computacao-em-nuvem/> Acesso em: 4 mar. 2016.

BRITO, Edivaldo. *Linux:* o núcleo dos sistemas operacionais de código aberto. 14 out. 2015. Disponível em: <http://www.techtudo.com.br/tudo-sobre/linux.html> Acesso em 05 fev. 2016.

FERREIRA, Ricardo. *As 10 melhores IDE's de programação para Linux*. 14 jan. 2014. Disponível em: <http://www.linuxdescomplicado.com.br/2014/01/as-10-melhores-ides-de-programcao-para.html> Acesso em 6 fev.2016.

GIOVANANGELO, Felipe. *Um pouco sobre banco de dados hierárquico*. 07 dez. 2011. Disponível em: <https://www.profissionaisti.com.br/2011/12/um-pouco-sobre-banco-de-dados-hierarquico/> Acesso em 09 fev. 2016.

GIOVANI, Jefferson. *Surge um novo SGBD:* MariaDB (adeus MySQL?). Disponível em: <http://jeffe.com.br/surge-um-novo-sgbd-mariadb-adeus-mysql.html> Acesso em 18 fev. 2016.

GREIN, Dirceu; TOMITA, Kátia Francelino. *Sistemas de gerenciamento de banco de dados orientado a objetos.* Bate Byte. Coluna do Estagiário Celepariano. Disponível em: <http://www.batebyte.pr.gov.br/modules/conteudo/conteudo.php?conteudo=560> Acesso em 14 fev. 2016.

MACEDO, Diego. *Introdução a Linguagem SQL:* Comandos Básicos e Avançados – Parte 1. 21 dez. 2011. Disponível em <http://www.diegomacedo.com.br/introducao-a-linguagem-sql-comandos-basicos-e-avancados-parte-1/> Acesso em 12 fev. 2016.

MariaDB Foundation – Ensuring continuity and open collaboration. Disponível em: <https://mariadb.org/about/> Acesso em 18 fev. 2016.

MATTOSO, Marta. *Introdução a Banco de Dados:* O modelo Relacional. Universidade Federal do Rio de Janeiro (UFRJ). Disponível em: <http://www.cos.ufrj.br/~marta/BdRel.pdf> Acesso em 10 fev. 2016.

MySQL – The world's most popular open source database. Disponível em: <http://www.mysql.com/> Acesso em 16 fev. 2016.

NASCIMENTO, Jean. *NoSQL* – você realmente sabe do que estamos falando?. iMasters. Disponível em: <http://imasters.com.br/artigo/17043/banco-de-dados/nosql-voce-realmente-sabe-do-que-estamos-falando/> Acesso em 14 fev. 2016.

Hypertext PreProcessor (PHP). Disponível em: <http://www.php.net/> Acesso em 25 fev. 2016.

Klaus Peter. *10 Motivos para você aprender a programar em Python.* Profissionais TI. LAUBE. Disponível em: <https://www.profissionaisti.com.br/2009/01/10-motivos-para-voce-aprender-a-programar-em-python/> Acesso em 28 fev. 2016.

INFO WESTER. *O que é cloud computing (computação nas nuvens)?* Disponível em: <http://www.infowester.com/cloudcomputing.php> Acesso em 3 mar. 2016.

LINS, Xerxes. GNU e Linux: amigos para sempre. Disponível em: <https://www.vivaolinux.com.br/artigo/GNU-e-Linux-amigos-para-sempre> Acesso em 5 de fev. 2016.

nVIDIA. *The power of cloud gaming*. Disponível em: <http://www.nvidia.com/object/cloud-gaming.html> Acesso em: 4 mar. 2016.

O Sistema Operacional GNU. Disponível em: <http://www.gnu.org/philosophy/free-sw.pt-br.html> Acesso em 3 fev. 2016.

Open Source Initiative (OSI). Disponível em: <https://opensource.org/> Acesso em 2 fev. 2016.

PISA, Pedro. *O que é e como usar o MySQL?.* TechTudo. Disponível em: <http://www.techtudo.com.br/artigos/noticia/2012/04/o-que-e-e-como-usar-o-mysql.html> Acesso em 16 fev. 2016.

Porquê usar Ubuntu?. UbuntuPT, comunidade portuguesa. Disponível em: <https://ubuntu-pt.org/content/porque-usar-ubuntu> Acesso em 4 fev. 2016.

ProfissionaisTI, *Desenvolvimento Web com Software Livre*. Disponível em: <https://www.profissionaisti.com.br/2008/12/desenvolvimento-web-com-software-livre/> Acesso em 2 fev. 2016.

Python Brasil. Disponível em: <http://wiki.python.org.br/> Acesso em 28 fev. 2016.

Python Software Foundation. Disponível em: <https://www.python.org/about/> Acesso em 27 fev. 2016.

RICARDO, Antônio. *O que é SaaS, IaaS, PaaS em Cloud Computing?* (Conceitos Básicos). Disponível em: <https://antonioricardo.org/2013/03/28/o-que-e-saas-iaas-e-paas-em-cloud-computing-conceitos-basicos/> Acesso em 4 mar. 2016.

ROUSE, Margaret. *SearchEnterpriseLinux, LAMP*. Disponível em: <http://searchenterpriselinux.techtarget.com/definition/LAMP> Acesso em 2 fev. 2016.

SABINO, Vanessa; KON, Fábio. *Licensas de Software Livre: Histórias e Características*. Disponível em: <http://ccsl.ime.usp.br/files/relatorio-licencas.pdf> Acesso em 3 fev. 2016.

SGBD: O que é?. EspacoInfo.net. Disponível em: <http://espacoinfo.net/o-que-e-sgbd-bd-ii/> Acesso em 10 fev. 2016.

SILVA, Jefferson Estanislau da. *GNU/Linux: Depois dele o mundo não é mais o mesmo!.* 08 mai. 2009. Disponível em: <https://www.vivaolinux.com.br/artigo/GNU-Linux-Depois-dele-o-mundo-nao-e-mais-o-mesmo> Acesso em 5 de fev. 2016.

SISNEMA. Cloud Computing – Novo modelo de computação. Disponível em: <http://sisnema.com.br/Materias/idmat019433.htm> Acesso em: 3 mar. 2016.

SOFTWARE Livre Brasil. Disponível em: <http://softwarelivre.org/portal/o-que-e>. Acesso em 2 fev. 2016.

The Apache Software Foundation. Disponível em: <http://www.apache.org/foundation/> Acesso em 8 fev. 2016.

The Apache Software Foundation. *How it Works.* Disponível em: <http://www.apache.org/foundation/how-it-works.html> Acesso em 8 fev. 2016.

Website Front-end and Back-end. 3NY Technology. Disponível em: <http://www.3nytechnology.com/website-frontend-and-backend/> Acesso em 23 fev. 2016.

WIKIPÉDIA, *Dicionário de dados na Wikipédia – Enciclopédia Livre.* Disponível em: <https://pt.wikipedia.org/wiki/Dicion%C3%A1rio_de_dados>. Acesso em 10 fev. 2016.