INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS

PRÓ-REITORIA DE ENSINO

CÂMPUS GOIÂNIA

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

DANIEL DE SOUZA MIRANDA

FELIPE GERVASIO DE SOUZA

VITOR DA COSTA SILVA

Projeto e implementação de um serviço para auxiliar na seleção de máquinas virtuais em nuvem

Goiânia, 2022.

INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS

PRÓ-REITORIA DE ENSINO
CÂMPUS GOIÂNIA

BACHARELADO EM SISTEMAS DE INFORMAÇÃO

DANIEL DE SOUZA MIRANDA
FELIPE GERVASIO DE SOUZA
VITOR DA COSTA SILVA

Projeto e implementação de um serviço para auxiliar na seleção de máquinas virtuais em nuvem

Trabalho de Conclusão de Curso apresentado à Coordenação do Curso de Bacharelado em Sistemas de Informação como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Raphael de Aquino Gomes

Goiânia, 2022.

M672p Miranda, Daniel de Souza.
Projeto e implementação de um serviço para auxiliar na seleção de máquinas virtuais em nuvem / Daniel de Souza Miranda; Felipe Gervasio de Souza; Vitor da Costa Silva. - Goiânia : Instituto Federal de Educação, Ciência e Tecnologia de Goiás, 2022.
52f. ; il.

Orientação: Prof. Dr. Raphael de Aquino Gomes.

TCC (Trabalho de Conclusão de Curso) - Curso de Bacharelado em Sistemas de Informação, Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

1. Computação em nuvem. 2. Máquinas virtuais. 3. Custos. I. Souza, Felipe Gervasio de. II. Silva, Vitor da Costa. III. Gomes, Raphael de Aquino (orientação). IV. Instituto Federal de Educação, Ciência e Tecnologia de Goiás. V. Título.

CDD 004.678 2

Ficha catalográfica elaborada pela bibliotecária Lana Cristina Dias Oliveira CRB1/ 2.631
Biblioteca Professor Jorge Félix de Souza,
Instituto Federal de Educação, Ciência e Tecnologia de Goiás, Câmpus Goiânia.

INSTITUTO FEDERAL
Goiás

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
SISTEMA INTEGRADO DE BIBLIOTECAS

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[X] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico e Educacional - Tipo: _____	

Nome Completo do Autor: Vitor da Costa Silva
Matrícula: 2018101109034
Título do Trabalho: Projeto e implementação de um serviço para auxiliar na seleção de máquinas virtuais em nuvem

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Goiânia-GO, 10/03/2022
Local Data

Vitor da Costa Silva
Assinatura do Autor e/ou Detentor dos Direitos Autorais

INSTITUTO FEDERAL
Goiás

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA
PRÓ-REITORIA DE PESQUISA E PÓS-GRADUAÇÃO
SISTEMA INTEGRADO DE BIBLIOTECAS

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[X] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico e Educacional - Tipo: _____	

Nome Completo do Autor: DANIEL DE SOUZA MIRANDA
Matrícula: 2017101103200043
Título do Trabalho: PROJETO E IMPLEMENTAÇÃO DE UM SERVIÇO PARA AUXILIAR NA SELEÇÃO DE MÁQUINAS VIRTUAIS EM NUVEM

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Goiânia-GO, 10/03/2022
Local Data

DANIEL DE SOUZA MIRANDA
Assinatura do Autor e/ou Detentor dos Direitos Autorais

TERMO DE AUTORIZAÇÃO PARA DISPONIBILIZAÇÃO NO REPOSITÓRIO DIGITAL DO IFG - ReDi IFG

Com base no disposto na Lei Federal nº 9.610/98, AUTORIZO o Instituto Federal de Educação, Ciência e Tecnologia de Goiás, a disponibilizar gratuitamente o documento no Repositório Digital (ReDi IFG), sem ressarcimento de direitos autorais, conforme permissão assinada abaixo, em formato digital para fins de leitura, download e impressão, a título de divulgação da produção técnico-científica no IFG.

Identificação da Produção Técnico-Científica

[ ] Tese	[ ] Artigo Científico
[ ] Dissertação	[ ] Capítulo de Livro
[ ] Monografia – Especialização	[ ] Livro
[X] TCC - Graduação	[ ] Trabalho Apresentado em Evento
[ ] Produto Técnico e Educacional - Tipo: _____	

Nome Completo do Autor: Felipe Lemônio de Souza.
Matrícula: 2017101109 0361
Título do Trabalho: Projeto e Implementação de um serviço para auxiliar na relação de máquinas virtuais em nuvem.

Autorização - Marque uma das opções

1. (X) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso aberto);
2. ( ) Autorizo disponibilizar meu trabalho no Repositório Digital do IFG somente após a data ___/___/_____ (Embargo);
3. ( ) Não autorizo disponibilizar meu trabalho no Repositório Digital do IFG (acesso restrito).

Ao indicar a opção **2 ou 3**, marque a justificativa:

( ) O documento está sujeito a registro de patente.
( ) O documento pode vir a ser publicado como livro, capítulo de livro ou artigo.
( ) Outra justificativa: _____

DECLARAÇÃO DE DISTRIBUIÇÃO NÃO-EXCLUSIVA

O/A referido/a autor/a declara que:

i. o documento é seu trabalho original, detém os direitos autorais da produção técnico-científica e não infringe os direitos de qualquer outra pessoa ou entidade;
ii. obteve autorização de quaisquer materiais inclusos no documento do qual não detém os direitos de autor/a, para conceder ao Instituto Federal de Educação, Ciência e Tecnologia de Goiás os direitos requeridos e que este material cujos direitos autorais são de terceiros, estão claramente identificados e reconhecidos no texto ou conteúdo do documento entregue;
iii. cumpriu quaisquer obrigações exigidas por contrato ou acordo, caso o documento entregue seja baseado em trabalho financiado ou apoiado por outra instituição que não o Instituto Federal de Educação, Ciência e Tecnologia de Goiás.

Goiânia-GO, 10/03/2022.
Local Data

Felipe Lemônio de Souza.
Assinatura do Autor e/ou Detentor dos Direitos Autorais

MINISTÉRIO DA EDUCAÇÃO
SECRETARIA DE EDUCAÇÃO PROFISSIONAL E TECNOLÓGICA
INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA DE GOIÁS
CÂMPUS GOIÂNIA

DANIEL DE SOUZA MIRANDA

FELIPE GERVASIO DE SOUZA

VITOR DA COSTA SILVA

Projeto e implementação de um serviço para auxiliar na seleção de máquinas virtuais em nuvem

Trabalho de Conclusão de Curso apresentado ao Curso de Bacharelado em Sistemas de Informação do Instituto Federal de Educação, Ciência e Tecnologia de Goiás, como parte das exigências para obtenção do Título de Bacharel em Sistemas de Informação.

Goiânia, 08 de fevereiro de 2022.

BANCA EXAMINADORA

Prof. Dr. Raphael de Aquino Gomes
Orientador - IFG/Câmpus Goiânia

Prof. Dr. Júnio César de Lima
Membro Externo da Banca Examinadora - IF Goiano/Câmpus Urutaí

Prof. Dr. Eduardo Noronha de Andrade Freitas
Membro Interno da Banca Examinadora - IFG/Câmpus Goiânia

Documento assinado eletronicamente por:

- **Júnio César de Lima, JÚNIO CÉSAR DE LIMA - DOCENTE - INSTITUTO FEDERAL GOIANO (10651417000178)**, em 10/03/2022 09:29:03.
- **Eduardo Noronha de Andrade Freitas, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 10/03/2022 08:54:42.
- **Raphael de Aquino Gomes, PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 08/03/2022 10:40:48.

Este documento foi emitido pelo SUAP em 03/03/2022. Para comprovar sua autenticidade, faça a leitura do QRCode ao lado ou acesse https://suap.ifg.edu.br/autenticar-documento/ e forneça os dados abaixo:

Código Verificador: 251850
Código de Autenticação: f7a55a88ed

Agradecimentos

Eu Daniel de Souza Miranda, gostaria de agradecer a Deus por me dar forças de onde não tinha para enfrentar todas as dificuldades ao longo desta jornada. Aos meus familiares e amigos por ter entendido momentos de ausência para um bem maior e a motivação incondicional durante toda a caminhada. Agradeço aos meus pais Odimar e Marilene e a minha irmã Nathália, que sempre souberam me instruir e auxiliar em todas as ocasiões e decisões, sendo os pilares da minha vida. Aos meus avós Américo, Terezinha, Isabel e Osmar, que mesmo alguns não estando mais presentes em vida, foram responsáveis pela minha base e, sem eles, não seria possível chegar onde cheguei hoje. Aos meus colegas Vitor e Felipe, que apesar de todas as adversidades advindas, nada disso seria possível de acontecer sem o esforço individual de cada, onde soubemos enfrentar juntos cada desafio proposto. Ao nosso orientador Dr. Raphael de Aquino, nos guiando neste trabalho, nos deixando não só ensinamentos acadêmicos, mas lições de vida que iremos levar para toda construção da nossa carreira e desenvolvimento pessoal. Aqui deixo minha gratidão a todos que participaram desta etapa da minha vida.

Eu Felipe Gervásio de Souza, agradeço aos meus familiares e amigos por ter me dado tanta força e apoiar incondicionalmente durante toda a caminhada. Agradeço aos meus pais Delma e João, que sempre souberam me instruir e auxiliar em todas as ocasiões e decisões, sendo os pilares da minha vida. Agradeço ao meu irmão Vinícius por ter me mostrado esse caminho que tanto amo hoje. Aos meus avós Alaide, Vicente, Matina e João, que mesmo sendo pessoas tão simples e humildes, souberam transmitir um pouco de cada um para a minha formação como pessoa. A minha namorada Anna Caroline que me apoia e me motiva em todas as decisões, esteve presente em toda essa jornada e soube entender todo esse processo. Aos meus colegas Daniel e Vitor, que apesar de todas as dificuldades enfrentadas, sem eles nada disso seria possível de acontecer, onde soubemos enfrentar juntos cada desafio. Ao nosso orientador Dr. Raphael de Aquino, na qual soube nos ensinar e ajudar neste trabalho, mas nos passou diversos ensinamentos que iremos levar para toda a vida. Todos vocês são pessoas valiosas e aqui deixo minha gratidão.

Eu Vitor da Costa Silva agradeço principalmente aos meus pais, por aceitarem quando eu decidi fazer este curso e que sempre me apoiaram durante todos os anos que

estive cursando, ansiosos para me ver formando, graças a eles tive forças para chegar onde estou. Agradeço também aos meus amigos que conheci durante minha graduação e também os que conheci no meu ensino médio pois sempre estiveram junto a mim tentando me apoiar e vencer os obstáculos. De maneira particular ao meu grande amigo Wandersom Moura que foi certamente um valioso amigo, colaborador e parceiro, que sempre me apoiou, me ajudou e cedeu seu tempo fazendo questão de compreender o contexto do tema proposto e agregar de alguma forma os seus conhecimentos para este trabalho. A minha namorada que esteve comigo a maior parte do tempo comigo durante a graduação e a paciência de compreender, entender e me apoiar. Ao meu orientador Dr. Raphael de Aquino que teve paciência, dedicação, pontualidade e que nos mostrou tudo que precisávamos para seguir em frente apesar das dificuldades. Obrigado a todos os envolvidos pelo apoio e auxílio nesta jornada.

Nos negócios, a velocidade importa e, além disso, um ambiente de rápida tomada de decisões é mais divertido também.

Jeff Bezos,

Resumo

Título: Projeto e implementação de um serviço para auxiliar na seleção de máquinas virtuais em nuvem
Autores: Miranda, Daniel de Souza, Souza, Felipe Gervasio de, e Silva, Vitor da Costa
Orientador: Gomes, Dr. Raphael de Aquino

O modelo em nuvem oferece alto nível de flexibilidade e controle de gerenciamento sobre recursos computacionais. Atividades baseadas em tecnologia estão aderindo à computação em nuvem por vantagens como maior disponibilidade, escalabilidade, segurança, velocidade e, principalmente, redução de custos. Contudo, desafios surgem nas escolhas de recursos das máquinas virtuais pois existem muitas opções de seleção, levando a não fazer bom uso da nuvem devido a tomada de decisões inapropriadas. Em virtude disso, este trabalho parte do comparativo entre os preços praticados das máquinas virtuais em nuvem e de seus respectivos provedores, dando enfoque na otimização de custo para a utilização de recursos. É proposto um serviço que permite consultar tipos de máquina virtual a partir da configuração de hardware, auxiliando nas provisões e tomada de decisões. A ferramenta foi projetada e implementada em um protótipo que demonstrou possíveis ganhos na sua utilização.

Palavras-chave

Otimização de Custos, Máquinas Virtuais, Computação em nuvem, Tomada de decisão

Abstract

Title: Design and implementation of a service to assist in the selection of cloud virtual machines
Authors: Miranda,Daniel de Souza, Souza, Felipe Gervasio de, and Silva, Vitor da Costa
Advisor: Gomes, Dr. Raphael de Aquino

The cloud model offers a high level of flexibility and management control over computing resources. Technology-based activities are adhering to cloud computing for advantages such as greater availability, scalability, security, speed, and, above all, cost reduction. However, challenges arise in choosing resources for virtual machines as there are many selection options, leading to not making good use of the cloud due to inappropriate decision making. As a result, this work starts by comparing the prices of virtual machines in the cloud and their respective providers, focusing on cost optimization for the use of resources. A proposed service allows consulting virtual machine types from the hardware configuration, helping in provisions and decision making. The tool was designed and implemented in a prototype that demonstrated possible gains in its use.

Keywords

Cost Optimization, Virtual Machines, Cloud Computing, Decision Making

Lista de Figuras

Lista de Códigos de Programas

Lista de Abreviaturas e Siglas

API	Application Programming Interface
AWS	Amazon Web Services
Capex	Capital Expenditure
CDN	Content Delivery Network
CLI	Command Line Interface
CPU	Unidade Central de Processamento
DDL	Data Definition Language
DNS	Domain Name System
DRY	Don’t Repeat Yourself
EC2	Elastic Cloud Computing
HTTP	HyperText Transfer Protocol
HTTP	Hypertext Transfer Protocol
IaaS	Infraestrutura como serviço
JSON	JavaScript Object Notation
MTV	Model, Template e View
MVC	Model-View-Controller
ODBC	Open Database Connectivity
Opex	Operational Expenditure
ORM	Mapeamento Objeto Relacional
ORM	Object Relational Mapping
PaaS	Plataforma como serviço
RAM	Random Access Memory
RDS	Relational Database Service
REST	Representational State Transfer
S3	Simple Storage Service
SaaS	Software como Serviço
SDK	Software Development Kit
SGBD	Sistema Gerenciador de Bancos de Dados
SO	Sistema Operacional
TCO	Total Cost of Ownership
URL	Unified Resource Location
VM	Virtual Machine

Sumário

CAPÍTULO 1

Introdução

Com o rápido desenvolvimento da tecnologia nas últimas décadas houve grande impacto em vários setores da sociedade, o que gerou como consequência uma dependência tecnológica, antes não visualizada. Equipamentos que a pouco tempo poderiam ser considerados eficientes passam a ser defasados devido a demandas cada vez mais elevadas de processamento e armazenamento.

O avanço indicado se deve principalmente à necessidade de mais produtividades, integrações e melhorias nas atividades (KURASOV, 2021). Conforme o software sofre alterações, sua estrutura original passa a ser descaracterizada e sua complexidade aumenta, de modo que também aumente gradativamente os custos de sua manutenção até o momento que a manutenção passe a se tornar inviável. Dessa forma, a solução passa a ser aumentar os esforços em manutenções preventivas, o que resultaria no aprimoramento de sua estrutura, facilitando as manutenções posteriores (LEHMAN, 1979). Com isso, a alta complexidade do software se traduz em demanda de hardware mais robusto, além de manutenções e atualizações periódicas.

Manter recursos localmente gera custos consideráveis, visto que servidores e demais recursos de infraestrutura precisam de um grande investimento inicial, envolvendo a compra de hardware, aquisição de software e dispositivos periféricos (IPSENSE, 2021). Além disso, é necessário investir em espaço para acomodar os componentes e treinamento de uma equipe especializada para realizar assistências e melhorias. Como alternativa, cada vez mais as empresas estão investindo no uso de recursos através do modelo de Computação em Nuvem (Cloud Computing). O termo Cloud Computing foi utilizado pela primeira vez em 1997 por Ramnath Chellappa (CANTU, 2021). Em meados dos anos 2000 esse modelo passa a ganhar mais visibilidade e força no mercado pois começa a ser oferecida comercialmente para usuários, assim se popularizando, como citado por IPM (2021).

A computação em nuvem está se tornando cada vez mais importante e popular no atual mercado de trabalho pois, conforme Veras (2015), o objetivo disso é tornar qualquer serviço mais econômico. É possível perceber constantes mudanças e a necessidade que inovações tecnológicas sejam implementadas e criadas através das capacidades

computacionais que a nuvem oferece. Isso se dá porque o modelo não visa somente grandes aplicações que demandam muitos recursos, mas passou a ser a forma padrão de utilização para a maioria das aplicações. Segundo Flexera (2021), cerca de 93% do mercado aderiu a computação em nuvem para ajudar a impulsionar os negócios digitais.

Computação em Nuvem se caracteriza pela oferta de serviços medidos, o que significa que praticamente todas as operações têm cobranças relacionadas. Dentre estes serviços, os que representam maior custo é a instanciação de máquina virtual (no inglês Virtual Machine (VM)), como aponta Nutanix (2021). Ainda de acordo com Nutanix (2021), o motivo deste grande custo é devido à provisão incorreta de recursos deixando máquinas virtuais e armazenamento de dados com alta capacidade não utilizada.

A seleção de máquinas virtuais é um dos aspectos mais críticos em ambientes de nuvem pois características de desempenho e custo devem ser balanceadas. Existem diversos tipos de instâncias otimizadas e de uso geral para atender diversos casos de uso. Estes tipos consistem em várias combinações de núcleos de Unidade Central de Processamento (CPU), memória, tipos de armazenamento e capacidade de rede (AWS, 2021d).

Para se conseguir a otimização de custos, ou seja, a redução de custos financeiros computacionais ao máximo valor tolerado, é necessário ter conhecimento e planejamento na plataforma de nuvem na qual se deseja trabalhar. Os atuais serviços para auxílio de tomadas de decisões para obtenção de menor custo em máquinas virtuais, levando em consideração o desempenho e o custo, como o Vantage Slack Community (2022), são praticamente inexistentes. Dentre pesquisas nesta direção pode ser citado o trabalho de Camilo e Silva (2009), onde são analisados os lances de preços de máquinas virtuais usando o modelo de cobrança spot[1], a serem usados para minimizar o custo total, mantendo um nível aceitável de desempenho. Porém, o trabalho não descreve e também não trata detalhes de outros modelos de cobrança comercializados.

No trabalho de Costa et al. (2018) é comparado o desempenho e o custo de transições de dados em nuvem das plataformas Amazon Web Services (AWS) (AWS, 2021c) e Google Cloud (GOOGLE, 2021). Os resultados mostram que não há uma grande diferença nos desempenhos de ambas as nuvens mas que existe melhores valores cobrados quando se utiliza servidores temporários.

Há ainda ferramentas disponibilizadas pelos próprios provedores de nuvem para estimativas de preços como: calculador de preços, oferecido pela AWS (AWS, 2021a) e Calculadora do Custo Total de Propriedade (Total Cost of Ownership (TCO)) pela Microsoft Azure (MICROSOFT, 2021a). Todavia, as ferramentas só tem funcionalidade

[1]O modelo de cobrança spot é uma opção de compra que permite que um cliente compre a capacidade extra não utilizada da infraestrutura da nuvem a uma taxa reduzida.

no escopo da nuvem pública fornecida pelos seus próprios provedores, ou seja, não existe medida de comparação com outras nuvens públicas, dificultando encontrar recursos com valores mais atrativos.

Uma forma de auxiliar na seleção de recursos em nuvem é identificar padrões, observando os comportamentos medidos e constituindo suas características. De acordo com Alexander (1977) um padrão consiste em “uma regra de três partes, que se expressa em uma relação entre um determinado contexto, um problema e uma solução”, ou seja, padrões tem como objetivo resolver um problema de acordo com o contexto da situação na qual se encontra.

No contexto de mineração de dados, padrões consistem em regras de associação ou sequências temporais, para detectar relacionamentos sistemáticos entre variáveis (CETAX, 2021). Estes relacionamentos podem ser úteis em diversas situações como identificação e solução de problemas em banco de dados, tomada de decisões e principalmente para criação de técnicas para otimização, dando vantagens em estratégias financeiras. Assim, padrões consistem em unidades de informação que se repetem ou variam em uma determinada escala previsível, podendo assim facilitar previsões, melhorias no desempenho e também possibilidades de otimizar custos.

Na computação em nuvem existem vários padrões quando se leva em consideração toda a arquitetura nela contida. Porém, os padrões de preços de instâncias de máquinas virtuais não são evidenciados aos usuários de forma clara para que consigam realizar melhorias. Diante disso, o presente trabalho propõe possíveis formas para otimização de custo na utilização de máquinas virtuais em provedores de nuvem pública. A solução proposta busca demonstrar as instâncias e os preços mais comumente praticados por estes provedores, além de propor uma ferramenta para a construção e identificação de padrões e possíveis métricas para otimizações dos preços em provedores de nuvem. Além do projeto da ferramenta é descrita a implementação de um protótipo desenvolvido para demonstrar sua aplicabilidade. Sua eficácia foi também comprovada considerando dados dos principais provedores de nuvem.

1.1 Objetivos

O objetivo geral desse trabalho é apresentar um serviço para escolha de máquinas virtuais, permitindo análise do custo mediante a descrição de melhores preços identificados, juntamente com sua capacidade computacional. Este objetivo geral se traduz nos seguintes objetivos específicos:

- Propor uma forma de implementação do serviço viabilizando tolerância a falhas.
- Desenvolver técnicas para tratativa e manipulação de dados sobre tipos de máquinas virtuais.

• Propor uma ferramenta que pode ser estendida para considerar novas métricas.

1.2 Metodologia

A metodologia utilizada para o projeto é a pesquisa exploratória. Como forma de familiarizar com o tema, foram realizada pesquisas bibliográficas cujo objetivo também foi encontrar métodos, proposições e ferramentas relacionadas ao tema do projeto. Após fazer a análise dos trabalhos relacionados ao tema proposto, foi necessário realizar um levantamento de materiais para aprofundar no assunto e compreender sobre os tipos de cobrança realizados nos provedores de nuvem.

Posteriormente aos estudos foi constatado a necessidade de desenvolver métricas para extração de informações visando formas de otimização de custos das instâncias dado a grande quantidade de dados manipulados. Paralelamente, foi desenvolvido um serviço que possibilitou encontrar resultados expressivos a partir dos preços extraídos a partir da Application Programming Interface (API) desenvolvida.

A partir dos resultados foi observado a possibilidade de auxílio que o serviço propõe na escolha de instâncias para determinadas atividades, além de instrumentar melhores tomadas de decisões e melhores preços na utilização de instâncias.

1.3 Organização do Trabalho

A estruturação do presente trabalho foi desenvolvida em cinco capítulos. A seguir é apresentada uma descrição dos capítulos restantes.

O Capítulo 2 apresenta a fundamentação teórica do trabalho, descrevendo computação em nuvem, o processo de mineração de dados e otimização de custos em nuvem. Além disso, é demonstrado alguns métodos utilizados para extração e mineração dos dados, e sua aplicabilidade.

O Capítulo 3 descreve o projeto da solução proposta. Neste é discutido as trativas dos dados e o processo de mineração dos dados, o que inclui sua normalização.

O Capítulo 4 descreve a implementação do protótipo desenvolvido a partir da proposta projetada.

Por fim, o Capítulo 5 traz considerações finais e possibilidades para trabalhos futuros.

CAPÍTULO 2

Referencial Teórico

Neste capítulo serão apresentados os conceitos principais relacionados ao trabalho desenvolvido. Dentre estes, será discutido o que é mineração de dados e sua função na otimização de custos. Além disso, a definição de Computação em Nuvem será mais amplamente apresentada e como mineração de dados pode ser usado para resolver problemas de sua aplicabilidade.

2.1 Computação em Nuvem

De acordo com a AWS (2021b) a Computação em Nuvem é a estratégia de disponibilizar variados tipos de recursos computacionais através da internet. Este modelo de negócios tem por objetivo proporcionar o fornecimento de serviços para armazenamento, banco de dados, software de análise, entre outros; que são utilizados sob demanda, sendo o pagamento realizado conforme o uso.

A Computação em Nuvem transforma os investimentos de capital (Capital Expenditure (Capex)) em investimentos operacionais (Operational Expenditure (Opex)), fazendo com que o uso da computação se torne mais rápido e eficaz. Ela nasce como uma alternativa cada vez mais promissora para o modelo tradicional de computação no qual a obtenção de recursos de infraestrutura é muito mais demorado por demandar aquisição de hardware, instalação de software, dentre outros.

O modelo de computação em nuvem foi desenvolvido com o objetivo de fornecer serviços de fácil acesso e de baixo custo, garantindo disponibilidade e escalabilidade. Ele permite grandes benefícios, dos quais os três principais são (ARMBRUST et al., 2010):

- Agilidade: é referente à possibilidade de gerar recursos computacionais conforme a necessidade, sem grandes esforços, desde serviços de infraestrutura como também análises de dados e aprendizado de máquina.
- Elasticidade: a provisão de recursos computacionais em escala conforme a demanda aumenta, ou seja, a nuvem consegue suprir as necessidades de atividades de forma automática sem que a mesma seja interrompida.

• Economia de custos: as despesas com servidores e datacenters físicos são convertidas em despesas com serviços consumidos.

Conforme a AWS (2021b), existem três principais ofertas de computação em nuvem:

• Infraestrutura como serviço (IaaS): se refere aos principais componentes de hardware, como servidores virtuais físicos dedicados.
• Plataforma como serviço (PaaS): onde o foco é na implantação e gerenciamento de ferramentas utilizados para o desenvolvimento de aplicativos.
• Software como Serviço (SaaS): na qual é disponibilizado ferramentas, produtos e softwares totalmente implementados.

2.1.1 Otimização de custos em nuvem

Uma das estratégias mais comumente usadas para otimização de custos de máquinas virtuais em nuvem é utilizar o modelo de cobrança spot sob o qual o preço da instância muda constantemente de acordo com a oferta e a demanda do mercado. O cliente especifica o preço (lance) mais alto que está disposto a pagar ao fazer o pedido. O provedor então determina o preço de mercado com base nas propostas enviadas. Se o preço de mercado subir e ultrapassar o lance, a instância será revogada, mas somente após um período de aviso. Enquanto o preço estabelecido ainda for menor do que o lance, a instância ainda estará disponível e, mesmo se o lance for mais alto, o cliente pagará apenas o preço de mercado.

A Figura 2.1 mostra as mudanças nos preços de mercado do tipo m4.2xlarge oferecido pela AWS na zona de disponibilidade 1e do leste dos Estados Unidos, dentro de um mês. Cada zona de disponibilidade é um mercado diferente e os preços também podem ser diversos. Neste exemplo, o preço atingiu o valor de U$ 0,40 (10 vezes menor que o usualmente praticado).

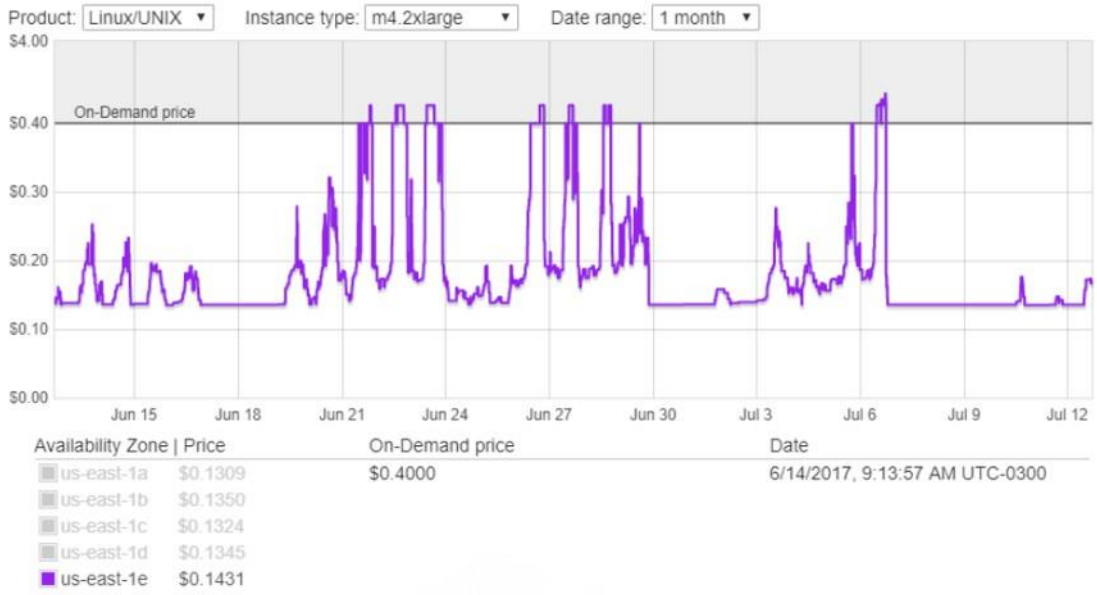

Figura 2.1: Variação do preço de uma instância usando o modelo de cobrança spot.

Fonte: (CAMARGO et al., 2016)

Devido à intermitência causada pelo modelo spot, geralmente os clientes adotam a contratação sob demanda, que garante disponibilidade sem necessidade de contratação prévia. Contudo, selecionar qual recurso alocar com base unicamente no preço atual pode representar riscos uma vez que este preço está sujeito a mudanças. Infelizmente, a plataforma de nuvem não divulga diretamente as estatísticas de cobrança de serviços, exigindo que os usuários as infiram indiretamente, como por meio de registros de histórico de preços. Portanto, encontrar um sistema de nuvem para selecionar a configuração de serviço mais adequada para atender às necessidades com base no preço histórico ou dados de disponibilidade é um desafio. Como exemplo, uma pesquisa recente mostrou que a redução do risco de revogação requer um sistema paralelo para diversificar suas necessidades de recursos, abrangendo vários tipos de serviços e envolvendo ainda mais a tomada de decisões (SHARMA; IRWIN; SHENOY, 2017).

O problema de seleção é agravado pelo grande número de opções de tipos disponíveis nos provedores: apenas na AWS há mais de 2500 opções de instâncias no serviço Elastic Cloud Computing (EC2).

Devido aos desafios listados, os provedores de nuvem começaram a oferecer ferramentas de seleção de serviços na qual se concilia com a proposta do trabalho apresentada. Como exemplo, Amazon SpotFleet (AWS, 2021b) automaticamente recoloca servidores revogados. No entanto, o serviço tem uma escolha limitada em termos de combinações de configurações que ele oferece e não resolve alguns dos desafios apresentados. Outra ferramenta, Amazon Spot Bid Advisor (AWS, 2021b), pode ajudar os usuários a selecionar servidores com base no preço, mas expõe apenas informações de volatilidade

oficial, como baixa, média ou alta categorização.

2.2 Mineração de dados

Segundo Goldschmidt, Passos e Bezerra (2015), a mineração de dados pode ser definida como um processo para obter dados utilizáveis de um conjunto maior de dados brutos. Isso acarreta na análise de modelos de dados em grandes lotes utilizando linguagens e softwares para otimização.

Este ramo da computação começou na década de 80 (UFOPA, 2013), quando peritos do assunto começaram a se preocupar com grandes quantidades de dados armazenados e não utilizados. Nesse período, o principal objetivo da mineração de dados era capturar informações de enormes bancos de dados de forma mais automatizada. Hoje em dia, a mineração de dados pode ser usada para analisar os dados extraídos, por exemplo, para apontar as necessidades reais e hipotéticas de cada cliente para realizar atividades de marketing. Por exemplo, uma empresa de cartão de crédito entende os hábitos de compra de cada um de seus milhões de clientes, o que inclui o que eles costumam comer, qual é o seu padrão de consumo, nível de endividamento. Para a empresa, essa informação é muito útil para estabelecer uma linha de crédito para cada cliente, além de conter dados muito valiosos sobre o comportamento de compra.

De acordo com Camilo e Silva (2009), aqui estão algumas das principais áreas que se beneficiam da mineração de dados e o fator de aplicação para cada uma:

- Busca e resgate de clientes: identificação de perfis para determinados produtos, vendas;
- Grandes corporações bancárias: identificar padrões para auxiliar no relacionamento com o cliente;
- Teleatendimento: acesso facilitado aos dados do cliente;
- Centrais de crédito: descobrir segmentos de mercado, identificar padrões de rotatividade;
- Cobrança: detecção de fraudes;
- Meio medicinal: indicação de diagnósticos mais precisos;
- Área da segurança: na detecção de atividades terroristas e criminais;
- Auxílio em pesquisas de biometria;
- Recursos humanos: captação de competências em perfis selecionados;
- Tomada de decisão: filtrar as informações relevantes, fornecer indicadores de probabilidade;
- Gestão empresarial: tratamento da grande massa de volume dados;
- Intensificação do tráfico de dados (navegação na Internet, catálogos online, etc).

2.2.1 As fases de processamento

Compreender o tipo de dados a serem processados é essencial para a escolha dos métodos e ferramentas mais adequados. Portanto, antes de aplicar qualquer ferramenta de inteligência ou mineração, é necessário explorar, descobrir e preparar dados.

De acordo com a Figura 2.2, as principais fases da mineração consistem em (GOLDSCHMIDT; PASSOS; BEZERRA, 2015):

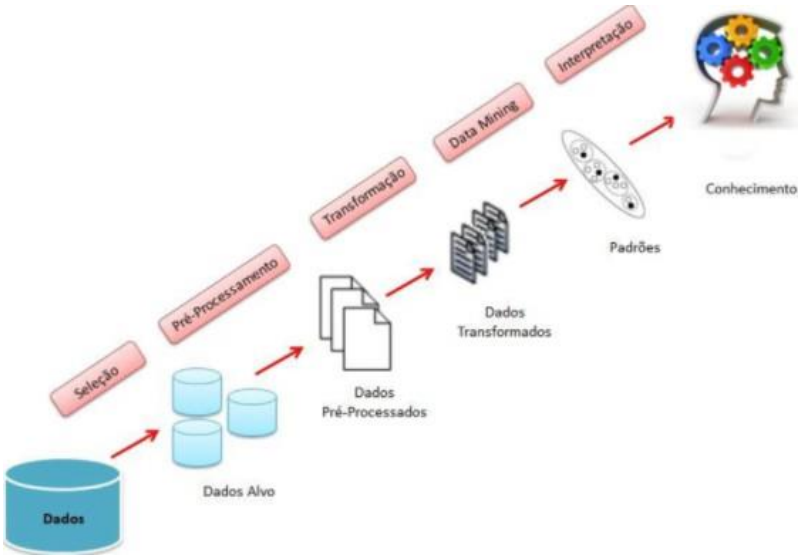

Figura 2.2: Fases da mineração de dados.
Fonte: (CAMARGO et al., 2016)

- Limpeza de dados: pode haver várias inconsistências nos dados, tal como registros incompletos e valores errados. Etapas de limpeza visam eliminar esses problemas para que não interfiram nos resultados do algoritmo utilizado (MITRA SUSHMITA; ACHARYA, 2003). As técnicas usadas nesta etapa vão desde a exclusão de registros problemáticos até a atribuição de valores padrão para a aplicação.
- Integração de dados: surge da necessidade de se obter um repositório único e consistente (MITRA SUSHMITA; ACHARYA, 2003). Os dados devem ser analisados minuciosamente, observando possível duplicação, diferentes categorias do mesmo valor, chaves diferentes, regras diferentes para os mesmos dados, etc.
- Modificação de dados: alguns códigos funcionam apenas em valores numéricos, outros funcionam apenas em valores categóricos. Portanto, é necessário converter valores numéricos em categorias ou valores categóricos em valores numéricos, converter dicionários em matrizes e converter matrizes em dicionários. Não existe um padrão único para modificação de dados, e diferentes técnicas podem ser usadas de acordo com o objetivo esperado (MITRA SUSHMITA; ACHARYA, 2003). Algumas das técnicas usadas nesta etapa são: suavização, agrupamento, generalização, normalização e criação de novos atributos.

• Redução de dados: A quantidade de dados usados na mineração geralmente é muito alta. Em alguns casos, esta quantidade é tão grande que a análise de dados e o processo de mineração se tornam impraticáveis (MITRA SUSHMITA; ACHARYA, 2003). Nesses casos, técnicas de redução de dados podem ser aplicadas para converter o volume de dados original em um volume de dados menor sem perder a representatividade dos dados originais. Isso permite que o algoritmo de mineração seja executado com mais eficiência, mantendo a qualidade dos resultados. As estratégias utilizadas nesta etapa são: criação de uma estrutura otimizada para os dados, seleção de subconjuntos de atributos, redução de dimensionalidade e discretização. Dentre as várias tecnologias, ela desempenha um papel muito importante na redução da dimensionalidade.

2.2.2 Áreas relacionadas

A mineração de dados possui grande importância em três grandes áreas:

- Estatística: De acordo com Sferra e Corrêa (2003), esta área se caracteriza como a análise de um volume de conjuntos de dados para descobrir relações e resumir os dados de uma forma que seja útil e fácil de entender para o proprietário dos dados.
- Banco de dados: é um campo de grande aprendizado que reúne tecnologias como: conhecimento de máquina, reconhecimento de padrões, estatísticas, bases de dados e visualização, com a capacidade de extrair informações dos dados (CAMILO; SILVA, 2009).
- Aprendizado de máquina: é uma etapa no processo de descoberta de conhecimento, incluindo a realização de análise de dados e aplicação de algoritmos de descoberta (SFERRA; CORRêA, 2003). Sob certas restrições computacionais, gera um conjunto de padrões a partir de um conjunto específico de dados.

2.2.3 Ferramentas utilizadas na mineração de dados

Um estudo apresentado em KDnuggets (2015) demonstra que as ferramentas mais utilizadas para mineração de dados se baseiam na linguagem Python (PYTHON, 2020) e no Sistema Gerenciador de Bancos de Dados (SGBD) MySQL (MYSQL, 2022).

Dentro do ecossistema Python, NumPy (OLIPHANT, 2006) é uma biblioteca para computação científica. Conforme ilustrado na Figura 2.3, implementa arrays multidimensionais e permite a fácil execução de operações matemáticas e lógicas, como ordenação, seleção, transformações, operações estatísticas, etc.

Outro exemplo é Pandas (WES, 2011), uma biblioteca desenvolvida em 2008 que, por sua vez, é a mais utilizada para análise de dados. Como ilustrado na Figura 2.4,

Figura 2.3: Funcionalidades do NumPy.

Fonte: Elaborado pelos autores.

ela é capaz de fornecer ferramentas para manipulação de estruturas de dados de forma extremamente simples. As operações complexas que trabalham com matrizes e vetores são facilmente implementadas obtendo um ótimo resultado de desempenho (MCKINNEY et al., 2011).

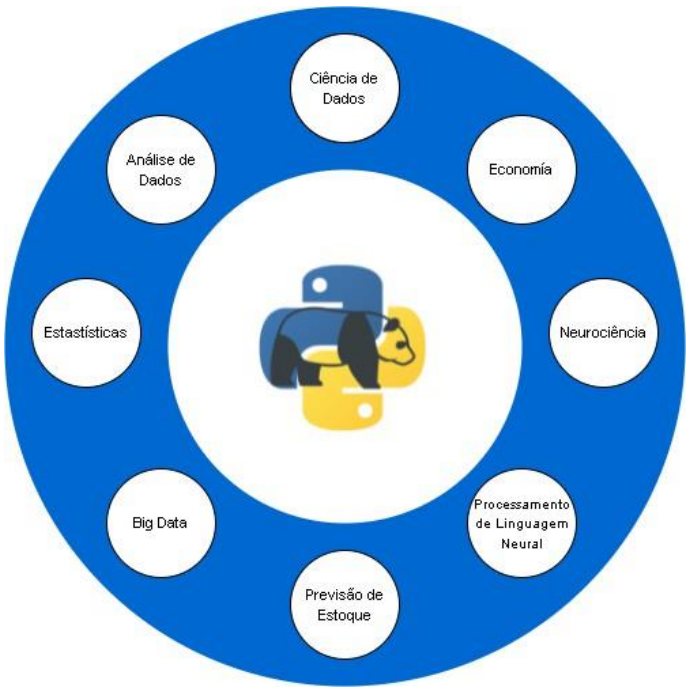

Figura 2.4: Funcionalidades do Pandas.

Fonte: Elaborado pelos autores.

A biblioteca Pandas visa reduzir lacunas de ferramentas de análise de dados dis-

poníveis em linguagens de computação científica, bem como plataformas de computação estatística. Não foi pretendido apenas fornecer funções equivalentes, mas também implementar muitas funções, como alinhamento automático de dados e indexação hierárquica, que não são fáceis de serem totalmente integradas em qualquer outra biblioteca ou ambiente de computação atuais. Embora originalmente desenvolvida para aplicações de análise de dados financeiros, é esperado que esta biblioteca torne o Python científico um ambiente de computação estatística mais atraente e prático para acadêmicos e profissionais da indústria (WES, 2011).

SQLAlchemy (SQLALCHEMY, 2021) é uma ferramenta de banco de dados e Mapeamento Objeto Relacional (ORM) para a linguagem Python, lançado em 2005. Desde o início, ela tentou fornecer um sistema ponta a ponta para lidar com bancos de dados relacionais em Python, usando uma API para interagir com o banco de dados. Os principais recursos incluem o manuseio de consultas SQL complexas, mapeamentos de objetos e a execução do modo “unidade de trabalho”, que fornece um sistema altamente automatizado para armazenamento de dados no banco.

Conforme a base de usuários continua a crescer, o SQLAlchemy, que começou com um conceito pequeno e mal implementado, progride rapidamente por meio de uma série de conversões e retrabalho representadas por novas iterações de sua arquitetura interna e API de origem pública (SQLALCHEMY, 2021). Quando a versão 0.5 foi lançada em janeiro de 2009, o SQLAlchemy começou a adotar uma forma estável que se provou em várias implantações de produção. Nas versões 0.6 (abril de 2010) e 0.7 (maio de 2011), as melhorias de arquitetura e API continuaram a tornar a biblioteca o mais eficiente e estável possível.

2.3 Interface de Programação de Aplicação

Uma Interface de Programação de Aplicação, ou API, pode ser definida como um conjunto de padrões de programação que permite a construção de aplicações, sendo que sua utilização é de maneira não tão evidente para os usuários (ORENSTEIN, 2000). Ela representa a maneira mais direta de como um software é disponibilizado em um servidor e se comunica com vários clientes. Trata-se também de um conjunto de padrões e rotinas estabelecidos por uma aplicação para que outras aplicações consigam usar suas funcionalidades.

A Figura 2.5 ilustra parte da abrangência do que uma API pode integrar, como web sites, aplicações nativas e até mídias sociais.

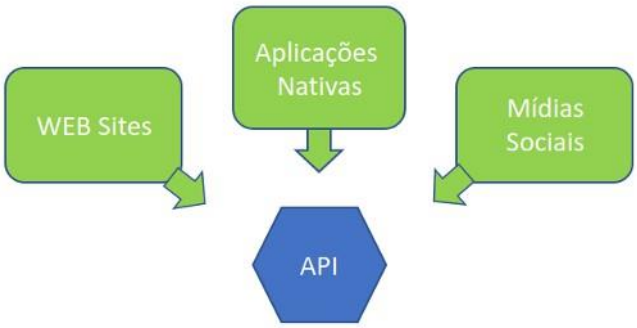

Figura 2.5: Softwares que comumente são integrados por meio de uma API.

Fonte: Elaborada pelos autores.

2.3.1 REST

O modelo Representational State Transfer (REST) foi proposto por Fielding (2000) e reúne uma série de princípios e restrições arquitetônicas para o desenvolvimento de aplicações distribuídas. As informações manipuladas pela abordagem REST são organizadas em recursos Web,que constituem um conjunto mínimo e coeso de dados.

Neste estudo, foi utilizado também o conceito de API REST, que trata-se de uma abstração da arquitetura da Web, consistindo em princípios e regras para que seja possível a realização da comunicação entre aplicações via protocolo HyperText Transfer Protocol (HTTP), utilizando seus métodos, como o GET e o POST, conforme representado na Figura 2.6.

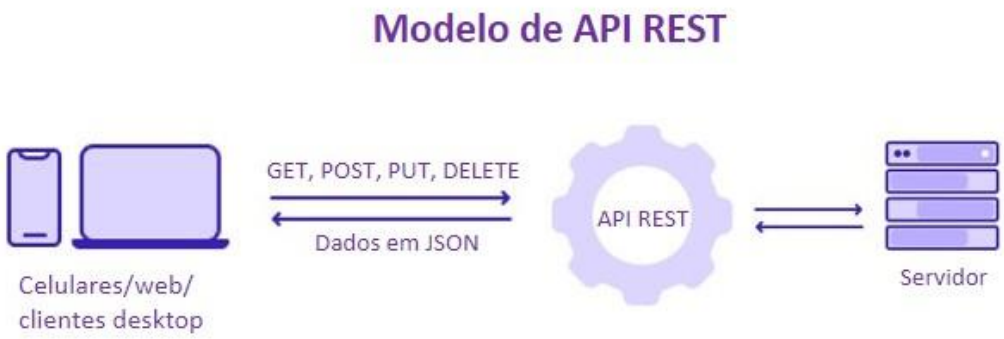

Figura 2.6: Interação com uma API REST.

Fonte: Adaptado de (VOXIMPLANT, 2020).

Web Services REST utilizam o protocolo HTTP para sua comunicação, implicando assim a obedecer a semântica deste protocolo, ou seja, utilizar corretamente seus verbos e os códigos de estado (NEUMANN; LARANJEIRO; BERNARDINO, 2021). Seu uso tem como objetivo possibilitar a transferência e integração de dados, permitindo que di-

ferentes aplicações troquem informações com um nível mais baixo de acoplamento, comparado ao compartilhamento de dados através de bases de dados.

2.4 Django Framework

Um framework nada mais é do que um template com diversas funções que podem ser usadas pelo desenvolvedor de uma aplicação, com o objetivo de agilizar o processo de desenvolvimento de soluções (IEEE. . . , 2020). No estudo em questão, foi utilizado o Django Framework (DJANGO, 2021a) juntamente com a linguagem Python para realizar a normalização e tratamento dos diversos dados capturados relacionados aos custos dos ambientes de nuvem.

Utilizando esse poderoso framework, é possível disponibilizar a aplicação que foi desenvolvida na Web em um ambiente acelerado, tornando as tarefas mais fáceis. Com isso, o projeto se torna totalmente escalável.

Conforme sua documentação (DJANGO, 2021a), a arquitetura trabalhada neste projeto é nomeada de Model View Template e consiste em um mapeamento objeto relacional que realiza a mediação entre os modelos de dados (classes Python) e um banco de dados relacional, um sistema para processamento de requisições HTTP com um sistema de modelos da Web (as chamadas Views) e um encaminhador de Unified Resource Location (URL) (chamado de Controller).

2.5 Considerações finais do capítulo

Neste capítulo foi realizada uma análise sobre os principais conceitos de mineração de dados voltados para otimização de custos em ambientes de nuvem. A demonstração dos conceitos discutidos é de suma importância para entender de forma ampla o que foi desenvolvido.

Vale ressaltar que, ao analisar todas as ferramentas e padrões utilizados no criação da aplicação, estas foram escolhidas com o intuito de otimizar o processo de desenvolvimento, utilizando as melhores ferramentas voltadas para a presente pesquisa.

CAPÍTULO 3

Proposta de serviço para auxílio na otimização de custos em nuvem

Este capítulo apresenta a proposta de serviço para o auxílio na otimização de custos em nuvem desenvolvida neste trabalho. São discutidos detalhes sobre a normalização dos dados e a arquitetura sugerida.

Conforme Veras (2015), a proposta principal da computação em nuvem é tornar a operação de qualquer serviço ou estrutura mais econômica e melhorar de alguma forma o uso de seus recursos. Também é claro que somente elasticidade propiciada pelos componentes de nuvem não é suficiente para garantir uma melhor qualidade dos recursos que uma determinada organização utiliza. No contexto de máquinas virtuais podemos dizer que além da elasticidade, a seleção do recurso em si representa um fator decisivo na organização.

A partir dessas premissas, foram realizadas novas revisões bibliográficas a fim de encontrar métodos de tomada de decisão, voltados especificamente à computação em nuvem. Foram selecionados os trabalhos de Veras (2015) e Saaty (1990), que visam esclarecer responsabilidades, direitos decisórios e métodos de resolução de tomadas de decisões usando múltiplos critérios. A partir da proposta e pesquisas mencionadas por Veras (2015) foi possível pensar e projetar um serviço que permitisse entregar ao cliente a melhor alternativa de máquina virtual, considerando características de desempenho e preço, respeitando os requisitos de componentes computacionais exigidos pela aplicação que será hospedada, auxiliando na tomada de decisões.

3.1 Arquitetura proposta

A arquitetura proposta para o serviço segue o padrão Model, Template e View (MTV) (PARR, 2004), bem similar ao comumente adotado Model-View-Controller (MVC) (LEFF; RAYFIELD, 2001). O MVC é um padrão popular porque isola a lógica da aplicação da camada de interface do usuário e diferentes partes da aplicação podem ser desenvolvidas separadamente. Nele, a parte do controlador é o código do software que controla

a interação entre o modelo e a visualização. Já no MVT, o próprio framework cuida da parte do controlador e o desenvolvedor não precisa se preocupar com como os dados se movem entre o modelo e a visualização.

O MVT atua como uma ferramenta que fornece uma maneira de criar e executar aplicações web (DJANGO, 2021b). Seus elementos são descritos a seguir:

- Model: é uma camada de abstração para estruturar e manipular os dados da aplicação web (DJANGO, 2021b). Ela atua como uma interface para a manutenção de dados e corresponde à uma estrutura de dados lógica por trás de todo a aplicação, ajudando a lidar com o banco de dados.
- View: esta camada encapsula a lógica responsável por processar a solicitação de um usuário e retornar uma resposta. É uma interface de usuário para executar a lógica e interagir com os modelos, sendo responsável por exibir todos ou parte dos dados ao usuário(DJANGO, 2021b).
- Template: fornece uma sintaxe amigável ao designer para renderizar as informações a serem apresentadas ao usuário. Ela contém as partes estáticas da saída HTML desejada junto com alguma sintaxe especial, descrevendo como o conteúdo dinâmico será inserido (DJANGO, 2021b).

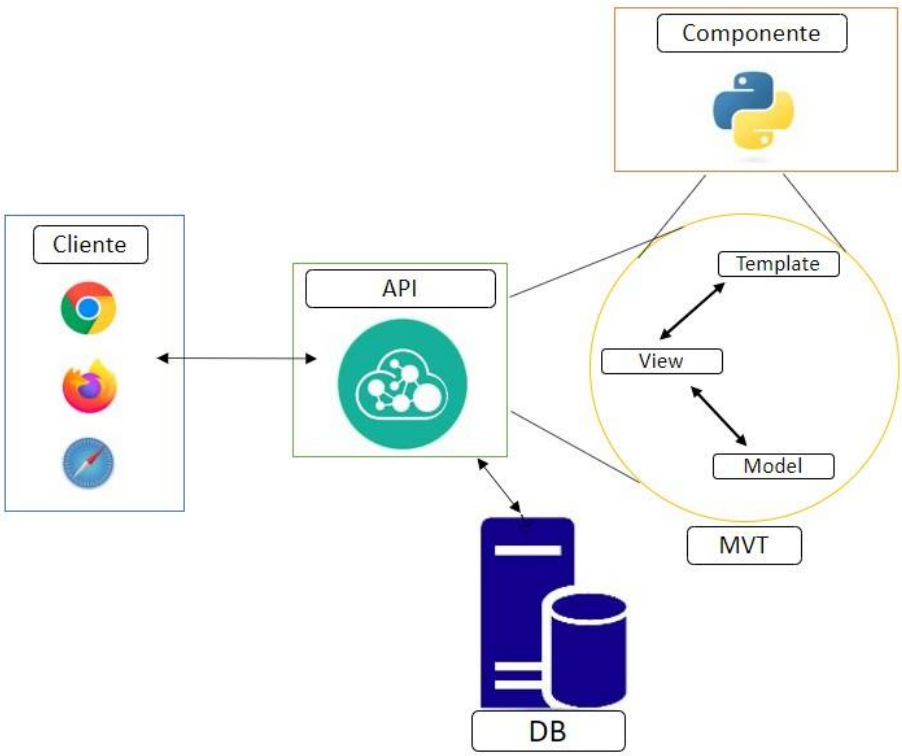

Figura 3.1: Arquitetura geral.

Fonte: Elaborado pelos autores.

Assim como ilustrado na Figura 3.1, a arquitetura proposta corresponde a uma implementação do padrão. Seu funcionamento se baseia no usuário requisitar à API,

através de uma rota (URL), um método que fora implementado na view. A partir disso, a view comunica à model que irá acessar a base de dados e retornar as informações, finalizando na template, que terá como função tornar visível as requisições pelo navegador (DJANGO, 2021a).

Para a manipulação de dados foi proposta a utilização de um componente baseado no padrão Object Relational Mapping (ORM) . Como pode ser visualizado na Figura 3.2, o componente responsável por esta tarefa possui duas camadas: ORM e core.

Figura 3.2: Arquitetura do componente de manipulação de dados.

Fonte: Elaborado pelos autores.

Na camada core, a interação com o SGBD é abstraída como sendo a API. Dessa forma, utilizando uma abordagem como Open Database Connectivity (ODBC) (GEIGER, 1995) é possível garantir a adoção e fácil migração entre soluções distintas. O acesso à API do SGBD é garantida por um motor de buscas que é dividido em duas partes: o dialeto responsável por alterar a sintaxe e os nomes dos comandos entre diferentes dialetos SQL; e um gerenciador de conexões que gerencia chamadas e mensagens enviadas pela rede à API do banco de dados. Além disso, há um componente chamado “Expressão de linguagem SQL” que constitui em um conjunto de ferramentas para construir expressões SQL representadas por objetos compostos. Com isso, é possível usar uma linguagem de alto nível, como Python, para se comunicar com o banco de dados, abstraindo todos os comandos SQL nativos.

Há também uma seção de esquemas e tipos, que são usados para converter tipos de dados existentes em uma linguagem de programação em tipos de dados que existem na linguagem SQL em diferentes implementações (SQLALCHEMY, 2021). Usando um banco de dados relacional algumas classes serão mapeadas para tabelas e os objetos que estão sendo instanciados serão armazenados como as tuplas dessas tabelas criadas. Como tal, as consultas ORM são recursos avançados que permitem programar de maneira muito natural e transparente a persistência de dados.

3.2 Proposta de implantação do serviço

Com a arquitetura do serviço apresentada há possibilidade de escalabilidade e agilidade ao usar uma plataforma de nuvem pública para a sua implantação e disponibilização. Como exemplo, é proposta a utilização da plataforma AWS para realizar essa atividade. A partir dos serviços disponibilizados por este provedor, foram identificados os serviços mais apropriados, conforme ilustrado na Figura 3.3.

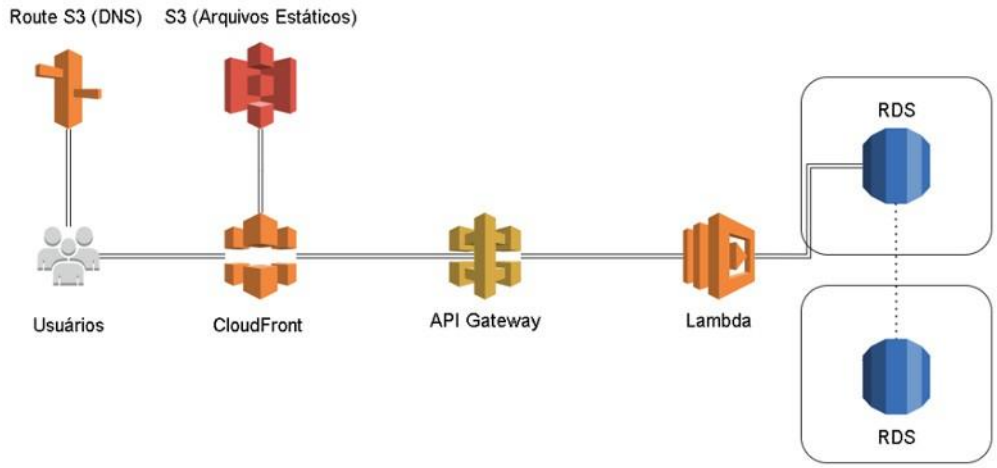

Figura 3.3: Arquitetura de implantação do serviço na nuvem AWS.

Fonte: Elaborada pelos autores.

Para a hospedagem do serviço é proposto a utilização do Lambda (AWS, 2022f), um serviço de computação sem servidor e orientado a eventos que permite executar código para praticamente qualquer tipo de aplicação ou serviço sem ser necessário provisionar ou gerenciar servidores. Sua adoção auxilia na redução de custos pois é cobrado apenas o tempo na qual o serviço for utilizado.

Também é proposto a utilização do serviço API Gateway (AWS, 2022a) para permitir a criação, publicação, monitoramento e proteção da API em qualquer escala com facilidade. Este serviço administra todas as tarefas envolvidas no recebimento e processamento de até centenas de milhares de chamadas de API simultâneas. Além disso, com o modelo de definição de preço em camadas, é possível reduzir os custos à medida que o uso da API é escalado.

Para a persistência de dados, Relational Database Service (RDS) (AWS, 2022c) é proposto pensando na otimização de recursos e desempenho para o banco de dados relacional, visto que a AWS oferece tipos de instâncias com foco neste tipo de demanda. Além disso, o serviço oferece facilidade de configuração, operação e também possibilidade de escalabilidade em nuvem.

O serviço Route 53 (AWS, 2022d) é proposto como alternativa para Domain Name System (DNS), que é um sistema distribuído de internet que mapeia nomes le-

gíveis por humanos. Por ser altamente disponível e escalável, auxilia na utilização do serviço mantendo-o sempre ativo. O serviço Route 53 também monitora continuamente a capacidade do serviço, sendo resiliente, conseguindo recuperar de falhas caso ocorra algo imprevisto.

Um dos mais utilizados serviços da AWS e com grande funcionalidade é o Simple Storage Service (S3) (AWS, 2022e) que foi proposto para armazenamento de objetos. O serviço pode ser utilizado para armazenar e proteger qualquer quantidade de dados de qualquer tipo de uso, sendo uma boa ferramenta para a implantação.

Para que o serviço seja entregue e distribuído é proposto o Amazon CloudFront (AWS, 2022b), um serviço de Content Delivery Network (CDN), ou seja, é um serviço que entrega conteúdo com segurança e baixa latência. Além disso, o serviço disponibiliza personalizações na entrega do conteúdo juntamente com o serviço Lambda. Isso proporciona um equilíbrio de custo e desempenho.

Todos os serviços apresentados são indispensáveis para o funcionamento do protótipo do serviço. A utilização de serviços serveless evita o gasto com servidores ociosos uma vez que o AWS Lambda cobra apenas pelas requisições processadas. Em contrapartida, o API Gateway (também serverless) é o ponto de entrada para redirecionar solicitações recebidas para funções do Lambda. Quanto à escalabilidade, o Lambda se auto gerencia e dimensionará para corresponder ao número de solicitações simultâneas em um determinado momento. Nota-se na proposta que foi priorizado otimização de custos, escalabilidade e tolerância a falhas para que sempre o serviço seja estável com uma boa usabilidade.

3.3 Considerações finais do capítulo

Neste capítulo foi apresentado a arquitetura e funcionamento do serviço proposto. Através desta foi especificado a forma de comunicação entre os componentes e a maneira na qual a aplicação trabalha para gerar resultados.

Esta arquitetura foi pensada de forma que a extração e tratamento de resultados seja realizada de forma ágil e organizada, trazendo dados consistentes e palpáveis para realização de análises. No próximo capítulo será apresentado como esta arquitetura foi implementada e avaliada.

CAPÍTULO 4

Implementação e Resultados Obtidos

Este capítulo tem como objetivo demonstrar a implementação realizada de acordo com a arquitetura proposta no capítulo anterior. Além disso, será discutido uma avaliação das funcionalidades e benefícios existentes na utilização do serviço proposto. Para tal, foi realizado uma estudo através dos dados minerados e posterior análise de resultados.

O serviço proposto foi implementado na linguagem Python, sendo que para implementação do padrão MVT foi utilizado o framework Django (DJANGO, 2021a). A adoção deste framework foi motivada pelo princípio de Don't Repeat Yourself (DRY) (VENNERS, 2003), fazendo com que a ferramenta fosse mais fácil de implementar ao aproveitar o máximo de código já criado, evitando ao extremo a repetição e reescrita de código. De forma mais detalhada, cada uma das camadas do padrão são implementadas como a seguir:

- Model: a classe django.db.models.Model é a fonte unitária e final de informações sobre os dados, contendo os campos e comportamentos essenciais dos dados armazenados. A geração de modelos é implementada criando subclasses desta, podendo incluir campos, métodos e metadados (DJANGO, 2021a). O modelo descrito no Código 4.1 apresenta um exemplo de mapeamento utilizado na presente classe, espelhando a base existente no banco de dados.
- Template: os arquivos nesta camada são documento de texto ou strings Python que usam uma linguagem própria do Django, baseada em HTML. Toda a renderização é tratada nestas entidades (DJANGO, 2021a).
- View: tem como principal função receber uma solicitação da web e retornar uma resposta usando o protocolo Hypertext Transfer Protocol (HTTP) (DJANGO, 2021a). Como exemplo, o Código 4.2 apresenta uma visualização que retorna a data e o horário atual.

Código 4.1 Exemplo de classe de modelo.

```
from django.db import models

class Machine(models.Model):
    machine_name = models.CharField(max_length=256, blank=True, null=True)
    memory = models.CharField(max_length=256, blank=True, null=True)
    storage = models.CharField(max_length=256, blank=True, null=True)
    processor = models.CharField(max_length=256, blank=True, null=True)
    colection_data = models.DateTimeField(blank=True, null=True)

    class Meta:
        managed = False
        db_table = 'machine'


class SystemMachine(models.Model):
    machine = models.ForeignKey(Machine, models.DO_NOTHING, blank=True, null=True)
    price = models.CharField(max_length=256, blank=True, null=True)
    region = models.CharField(max_length=256, blank=True, null=True)
    system_name = models.CharField(max_length=256, blank=True, null=True)
    description = models.CharField(max_length=256, blank=True, null=True)
    type_machine = models.CharField(max_length=256, blank=True, null=True)
    lease_contract_length = models.CharField(max_length=256, blank=True, null=True)
    offering_class = models.CharField(max_length=256, blank=True, null=True)
    purchase_option = models.CharField(max_length=256, blank=True, null=True)
```

Código 4.2 Exemplo de arquivos de visualização.

```
from django.http import HttpResponse
import datetime

def current_datetime(request):
    now = datetime.datetime.now()
    html = '<html><body>It is now %s.</body></html>'
    return HttpResponse(html)
```

A aplicação desenvolvida absorve toda a camada dos dados brutos e armazena em um banco de dados relacional, realiza análises a fim de identificar padrões e disponibiliza essas informações através de uma API. Sua implementação foi dividida em 3 etapas, descritas nas próximas seções.

4.1 Inclusão dos dados

As informações coletadas dos provedores em nuvem estão no formato JavaScript Object Notation (JSON), um formato de texto de intercâmbio de dados leves, o qual é fácil para humanos ler e escrever e facilita para máquinas analisar e gerar resultados. O JSON é baseado em duas estruturas: uma coleção de pares de nome e outra coleção de pares

valor, onde o nome representa a identificação daquele objeto e o valor é a informação que aquele nome tem.

Os arquivos sobre dados de oferta são buscados usando a API AWS Price List (AWS, 2022g), que permite consultar informações específicas sobre serviços, produtos e preços da AWS, utilizando o Software Development Kit (SDK) ou Command Line Interface (CLI). Essa API fornece informações sobre determinados produtos, preços ou regiões via JSON, permitindo obter informações sobre a precificação.

A API Price List retorna os preços atuais e não retém informações históricas. Com a API de consulta, é possível utilizá-la para buscar, por exemplo, informações de preços de instâncias do Amazon EC2 com 4 vCPU, 8 GiB de memória na região sa-east-1, conhecida como América do Sul - São Paulo.

Nos arquivos JSON retornados estão informações completas de toda a configuração das máquinas, como o nome do tipo, quantidades de núcleos de processamento, memória Random Access Memory (RAM), entre outras informações relevantes. A partir disso é iniciado o processamento dos dados, onde primeiramente é passado a localização da pasta que contém os arquivos JSON obtidos, provindos da API AWS Price List.

Com os dados no padrão esperado, primeiramente a aplicação inicializa a comunicação entre a API AWS Price List e o banco de dados. Em seguida, ela realiza a inclusão dos dados, que são disponibilizados pela API no formato JSON, no banco de dados relacional. Foi utilizado o SGBD MySQL pela sua facilidade de utilização e manipulação dos dados.

No contexto do presente trabalho, foi realizado a inclusão de todos os arquivos disponibilizados pela API AWS Price List durante o período entre março de 2020 e janeiro de 2021. Esses arquivos foram analisados, verificados a integração dos dados, realização de limpeza e transformação dos dados. Após a coleta, foi realizada a inserção na base de dados, de forma estruturada, respeitando cada informação individual de determinado tipo.

Antes de realizar a importação foi necessário realizar a padronização dos dados, onde foram identificadas as principais informações no arquivo JSON. Dentre as informações importantes está o Sistema Operacional (SO), região na qual a oferta de máquina virtual se encontra, além de informações de hardware do tipo.

4.2 Geração da base de dados

Para a geração da base de dados foi utilizado o SGBD MySQL. O uso de um SGBD torna fácil disponibilizar o banco de dados aos usuários apropriados e permite a consolidação de dados, tendo como consequência a imposição de padrões e o melhor gerenciamento deles (FONSECA, 2020).

O projeto do banco de dados se deu em três fases:

• Modelo Conceitual: a parte mais abstrata da modelagem, onde as circunstâncias reais são descritas de uma forma mais simples. Responsável por demonstrar o conceito da futura implementação (DEBASTIANI, 2015). A Figura 4.1 apresenta o modelo conceitual usando a notação entidade-relacionamento. De acordo com a modelagem, machine representa as propriedades de hardware de um determinando tipo de VM, ao passo que system_machine representa a oferta deste tipo considerando um determinado SO e região. Isso foi proposto diante do fato de um mesmo tipo ser ofertado em diferentes regiões e SOs.

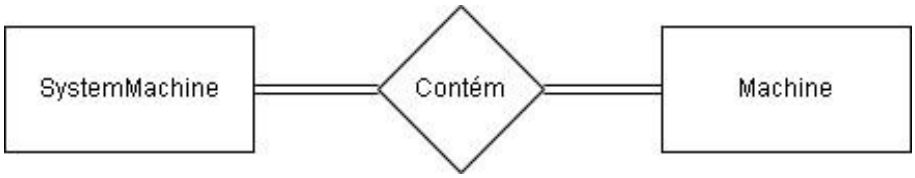

Figura 4.1: Modelo conceitual da base de dados.

Fonte: Elaborada pelos autores.

• Modelo Lógico: descreve o banco de dados no nível do SGBD, ou seja, depende do tipo particular de SGBD que será usado. O modelo lógico do banco de dados relacional deve definir quais as tabelas e o nome dos campos que compõem estas tabelas. Nesse momento é definido a estrutura de registro do banco, seus registros e números de campos com seus respectivos tamanhos (DEBASTIANI, 2015). A Figura 4.2 apresenta o modelo lógico elaborado.

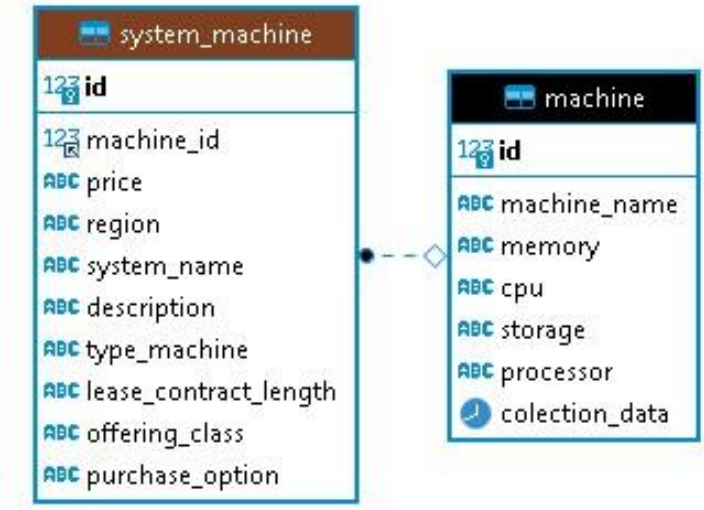

Figura 4.2: Modelo lógico da base de dados.

Fonte: Elaborada pelos autores.

• Modelo Físico: este modelo é desenvolvido a partir de um projeto lógico já antes definido. Nesse parte, é informada a estrutura física de armazenamento, tabelas,

índices, tipo dos registros, etc. (DEBASTIANI, 2015). As Figuras 4.3, 4.4 e 4.5 descrevem o modelo físico.

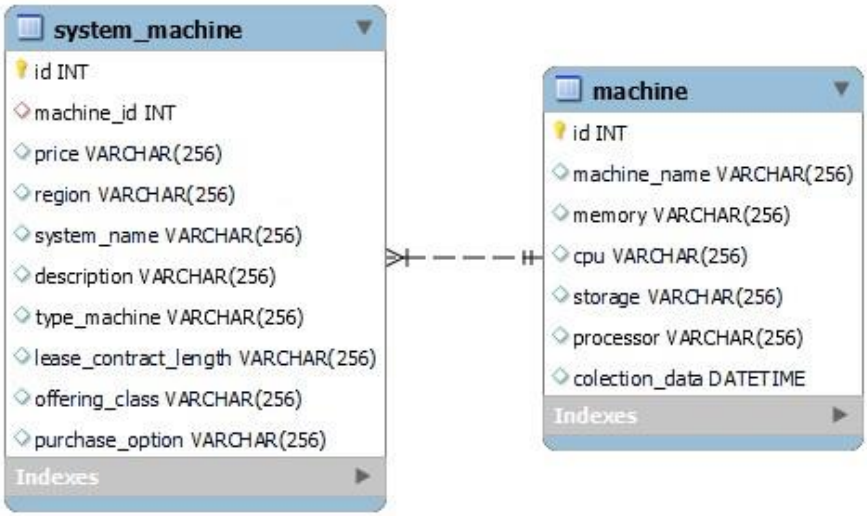

Figura 4.3: Modelo físico da base de dados.

Fonte: Elaborada pelos autores.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
machine_id	INT	☐	☐	☐	☐	☐	☐	☐	☐	NULL
price	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
region	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
system_name	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
description	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
type_machine	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
lease_contract_length	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
offering_class	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
purchase_option	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Figura 4.4: Tabela System Machine

Fonte: Elaborada pelos autores.

Column Name	Datatype	PK	NN	UQ	B	UN	ZF	AI	G	Default/Expression
id	INT	☑	☑	☐	☐	☐	☐	☑	☐	
machine_name	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
memory	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
cpu	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
storage	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
processor	VARCHAR(256)	☐	☐	☐	☐	☐	☐	☐	☐	NULL
colection_data	DATETIME	☐	☐	☐	☐	☐	☐	☐	☐	NULL
		☐	☐	☐	☐	☐	☐	☐	☐	

Figura 4.5: Tabela Machine

Fonte: Elaborada pelos autores.

É na modelagem física que será informado os scripts de criação da base de dados. A linguagem utilizada por padrão, Data Definition Language (DDL), define comandos para a criação, alteração e remoção das tabelas. Os Códigos 4.3 e 4.4 descrevem os scripts equivalentes.

Código 4.3 DDL da classe Machine

```
CREATE TABLE `machine` (
  `id` int NOT NULL AUTO_INCREMENT,
  `machine_name` varchar(256) DEFAULT NULL,
  `memory` varchar(256) DEFAULT NULL,
  `cpu` varchar(256) DEFAULT NULL,
  `storage` varchar(256) DEFAULT NULL,
  `processor` varchar(256) DEFAULT NULL,
  `colection_data` datetime DEFAULT NULL,
  PRIMARY KEY (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=6250 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

Código 4.4 DDL da classe SystemMachine

```
CREATE TABLE `system_machine` (
  `id` int NOT NULL AUTO_INCREMENT,
  `machine_id` int DEFAULT NULL,
  `price` varchar(256) DEFAULT NULL,
  `region` varchar(256) DEFAULT NULL,
  `system_name` varchar(256) DEFAULT NULL,
  `description` varchar(256) DEFAULT NULL,
  `type_machine` varchar(256) DEFAULT NULL,
  `lease_contract_length` varchar(256) DEFAULT NULL,
  `offering_class` varchar(256) DEFAULT NULL,
  `purchase_option` varchar(256) DEFAULT NULL,
  PRIMARY KEY (`id`),
  KEY `machine_id` (`machine_id`),
  CONSTRAINT `system_machine_ibfk_1` FOREIGN KEY (`machine_id`) REFERENCES `machine` (`id`)
) ENGINE=InnoDB AUTO_INCREMENT=842367 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4_0900_ai_ci
```

4.2.1 Manipulação de dados

Para implementar o componente de persistência de dados foi usada a ferramenta SQLAlchemy (SQLALCHEMY, 2022). Dessa forma, ao usar o ORM, o processo de configuração primeiro descreve as tabelas do banco de dados manipuladas, e então são definidas as classes que serão mapeadas para essas tabelas (SQLALCHEMY, 2021). As duas tarefas geralmente são executadas juntas, usando um sistema chamado extensão declarativa, que permite criar classes contendo instruções para descrever as tabelas reais do banco de dados para as quais serão mapeadas (SQLALCHEMY, 2021).

4.2.2 Normalização do banco de dados

De acordo com a Microsoft (2021b), a normalização de dados é muito utilizado em aprendizado de máquina para auxiliar na preparação dos dados. Neste trabalho foi

utilizada a biblioteca de desenvolvimento Pandas e Numpy adotando uma técnica já implementada para transformar o conjunto de dados obtidos através da API em uma escala comum, sem distorcer e perder informações.

Antes de utilizar a técnica de normalização foi necessário identificar padrões nos arquivos JSON coletados dos provedores de nuvem. Foi realizada a identificação de informações através das tags descritas no arquivo JSON, sendo constatada a existência de uma mesma estrutura mas com dados provenientes de diferentes instâncias. Dessa forma foi notado um padrão de informações relevantes que poderia ser manipulado. Esse padrão consiste basicamente em: nome da máquina, preço e região, onde foi notado modificações nos preços de máquinas virtuais quinzenalmente, sendo que em dias próximos quase não houve alterações. Isso foi importante para executar um filtro de relevância dos dados.

Após identificação dos padrões foi realizada a normalização dos dados, que é o processo de organização de dados no banco de dados. Essa normalização foi realizada de forma a eliminar redundâncias.

4.3 Integração dos dados via API

Após as etapas de normalização e tratamento dos dados, foi iniciada a passagem desses dados para a API REST. Desde a versão 1.8 do Django, existe a classe JsonResponse, que é uma subclasse de HttpResponse, responsável por criar uma resposta HTTP codificada em JSON.

Para realizar esasa passagem dos dados tratados para a API REST, foi necessário iniciar a sua estrutura básica de acordo com o padrão do framework. Com isso, foram implementadas as classes Python destinadas à realização da normalização e tratamento dos dados.

De maneira simples, a API é implementada em dois arquivos Python:

- views: uma view basicamente é uma função Python que recebe uma solicitação web e retorna uma resposta. Essa resposta, no estudo em questão, é justamente o retorno JSON do método implementado anteriormente. Foi nesta etapa que a classe JsonResponse foi utilizada para realizar o retorno dos dados.
- urls: no Django, as URLs são essenciais para o seu funcionamento, pois é nesse arquivo que são mapeadas as views da aplicação. No estudo, foram criadas cinco URLs e cada uma contém um retorno JSON com valores específicos aos cálculos realizados.

4.4 Funcionalidades e Análise dos Resultados

Após análise das normalizações e padrões encontrados, foi desenvolvido um protótipo de serviço onde foi realizado o aprimoramento do processamento de dados, viabilizando a geração de resultados.

O protótipo do serviço atualmente aborda duas grandes funcionalidades: Comparação de Instâncias e Análise de Preços, onde o serviço realiza a comparação entre máquinas virtuais e seus respectivos preços e sugere instâncias com capacidades suficientes para desempenhar adequadamente a aplicação na qual será utilizada.

Como forma de validação, foi realizado comparações entre máquinas virtuais sugeridas pelo serviço desenvolvido, com máquinas virtuais já utilizadas em empresas, selecionadas por estas empiricamente.

Com o objetivo de ampliar e melhorar o entendimento dos resultados, ao invés de comparar apenas preços (mesmo este sendo o principal método de análise), foram considerados outros fatores importantes e muita das vezes com grande relevância para empresas e usuários do serviço. Inicialmente foi considerado a região como um dos principais pontos de relevância, uma vez que determinadas aplicações exigem certos requisitos de latência ou até mesmo por opção do usuário.

Mesmo com toda a redução de possibilidades de instâncias a serem sugeridas devido a consideração das regiões, o serviço desenvolvido foi capaz de auxiliar na comparação e análise de preços por instâncias. Para isso, foi realizado como principal cálculo o preço da máquina virtual utilizado na empresa menos o preço da máquina virtual equivalente oferecido pelo serviço. Isso possibilitou gerar análises mensais de custos e observar a totalidade de gastos mensais a mais das empresas em comparação a utilização do serviço. A Figura 4.6 apresenta os resultados.

De acordo com esse resultado é possível realizar uma análise sobre a economia proporcionada pelo uso do serviço. Conforme mostrado na Figura 4.7, ao se comparar os valores atualmente pagos por instâncias nas empresas com os valores que poderiam ser pagos utilizando o serviço, é possível economizar até U$ 149,00 mensais e até U$ 1.790,00 anuais, sendo valores tanto quanto consideráveis

A próxima análise realizada foi repetir as consultas desconsiderando o fator região, visando somente a seleção do sistema operacional. A Figura 4.8 apresenta os resultados. É importante observar que neste caso a redução de custo foi muito mais significativa, pois foi deixando de lado o parâmetro de latência.

Continuando a análise, na Figura 4.9 é possível verificar que os registros de valores apresentados ao usar o serviço tem números bastante expressivos e muito abaixo dos gastos atuais das empresas. Como exemplo, gastos de U$ 2.917,87 passariam a ser somente U$ 1.717,99 dólares para a Empresa C. O melhor cenário se refere à Empresa D

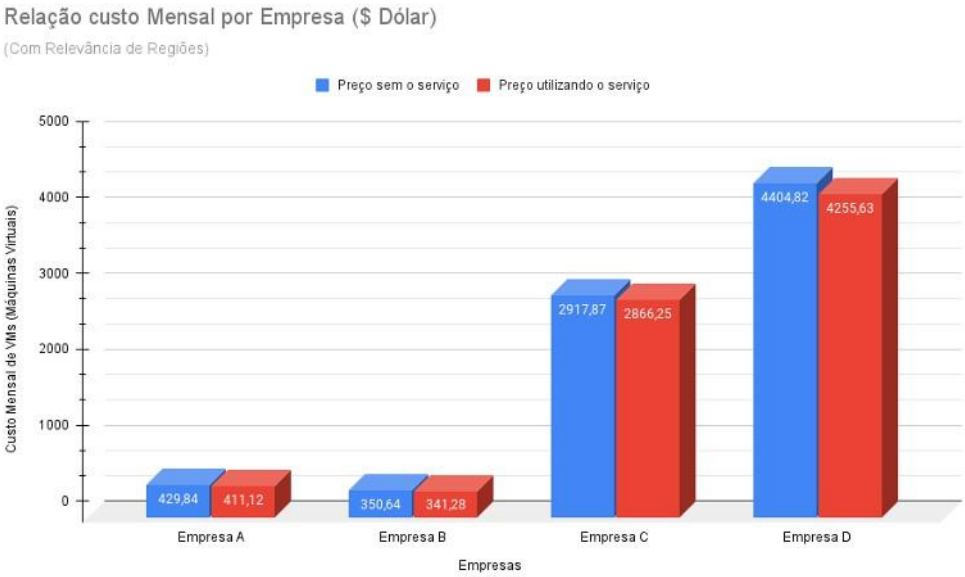

Figura 4.6: Custo mensal por empresa considerando a região.

Fonte: Elaborada pelos autores

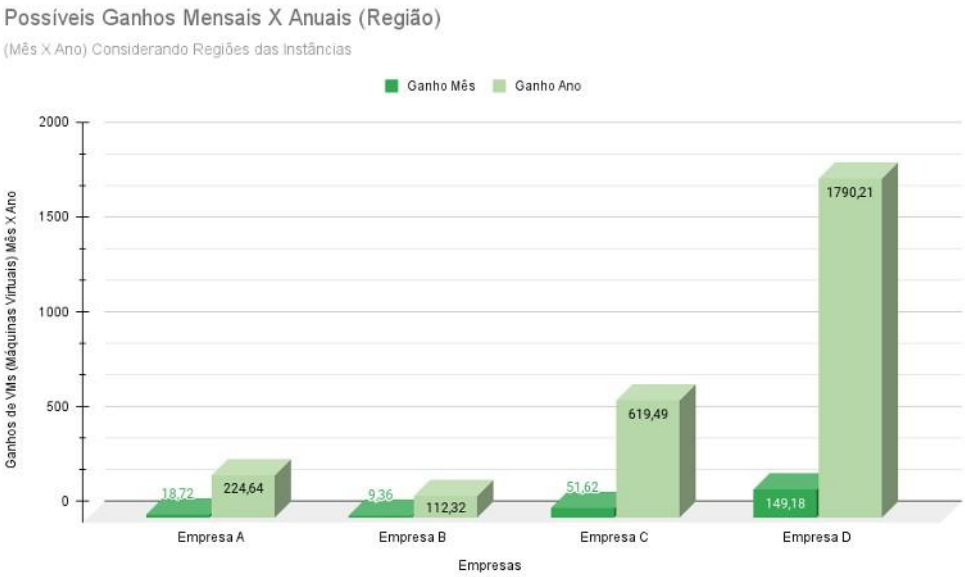

Figura 4.7: Ganhos mensais e anuais considerando regiões.

Fonte: Elaborada pelos autores

com ganhos mensais de U$ 1.233,79 mensais e ganhos anuais superando os U$ 14 mil, justificando a utilização do serviço e o quanto seu uso pode ser lucrativo.

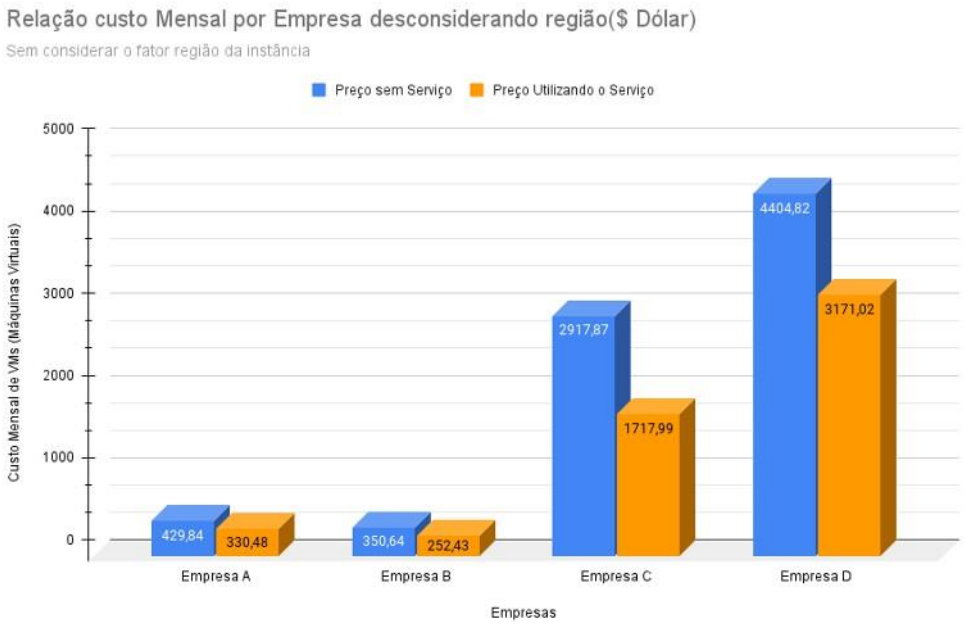

Figura 4.8: Custo mensal por empresa desconsiderando a região.

Fonte: Elaborada pelos autores

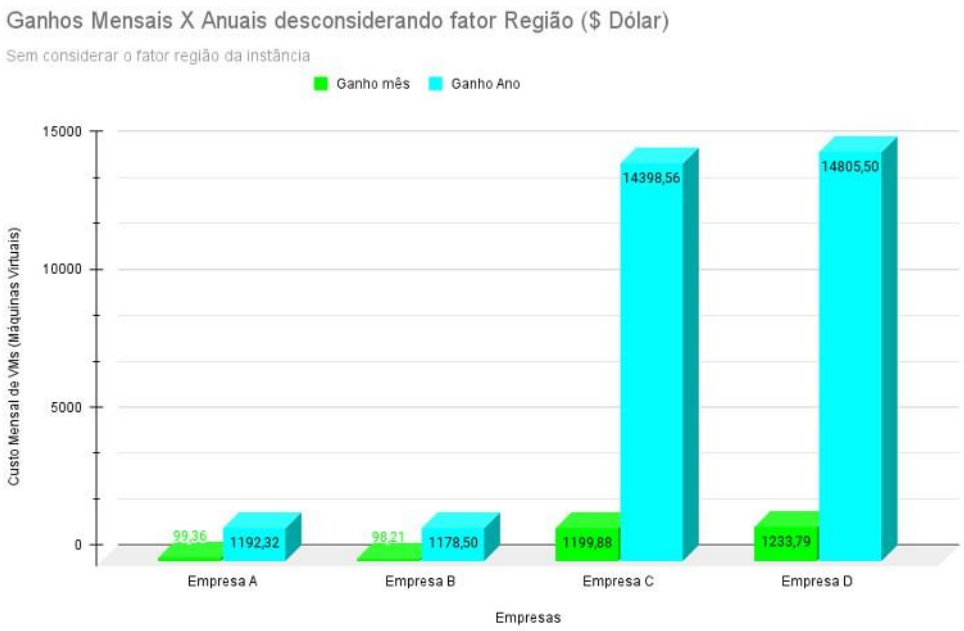

Figura 4.9: Ganhos mensais e anuais desconsiderando regiões.

Fonte: Elaborada pelos autores

4.5 Possibilidades com o serviço

A transparência de como é precificado as máquinas virtuais em nuvem foi um dos principiais focos da pesquisa. Além disso, API em questão torna mais fácil acessar

programaticamente os dados de preços da nuvem.

A partir desses resultados será mais fácil e objetivo explorar os preços e cálculos por trás das estimativas e encontrar os tipos de instâncias disponíveis com os melhores custos que atendam às necessidades. Isso permitirá uma melhor decisão sobre o uso da nuvem.

CAPÍTULO 5

Considerações Finais

O objetivo deste trabalho foi desenvolver um serviço com a finalidade de gerar indicadores sobre otimização de custos sobre máquinas virtuais em nuvem. Com isso em mente, tornou-se necessário avaliar a ferramenta proposta para identificar os pontos de melhoria e validar os resultados. Com este protótipo e os resultados obtidos espera-se a entrega de uma solução relevante.

Os resultados obtidos através deste trabalho apresentam contribuições importantes para a diminuição do custo e esforço na escolha de uma máquina virtual em ambientes em nuvem, facilitando a adoção desta arquitetura no desenvolvimento de aplicações para a web.

Foram notadas grandes perspectivas de utilização para trabalhos futuros com serviço desenvolvido. Inicialmente, visando uma melhor forma de orquestração dos componentes, sugere-se para futuras versões o orquestrador Zappa (WES, 2019), um pacote Python que agrupa aplicações web escritos em Django e os implanta no AWS Lambda. Esta ferramenta terá como principal funcionalidade a criação, implantação e gerenciamento de software serverless na infraestrutura AWS.

Outros trabalhos futuros incluem a implementação de novas funcionalidades na API e o suporte a pesquisa em múltiplos provedores de nuvem, visando suporte ainda maior para otimização de custos.

Referências Bibliográficas

ALEXANDER, Christopher. A pattern language: towns, buildings, construction. [S.l.]: Oxford university press, 1977. 19

ARMBRUST, Michael; FOX, Armando; GRIFFITH, Rean; JOSEPH, Anthony D; KATZ, Randy; KONWINSKI, Andy; LEE, Gunho; PATTERSON, David; RABKIN, Ariel; STOICA, Ion. A view of cloud computing. Communications of the ACM, ACM New York, NY, USA, v. 53, n. 4, p. 50-58, 2010. 21

AWS. AWS Pricing Calculator. 2021. Disponível em: https://calculator.aws. Acesso em: 08 mar. 2022. 18

____. O que é cloud computing (computação em nuvem)? - Amazon Web Services. 2021. Disponível em: https://aws.amazon.com/pt/what-is-cloud-computing/?nc1=f_cc. Acesso em: 08 mar. 2022. 21, 22, 23

____. Serviços de computação em nuvem - Amazon Web Services (AWS). 2021. Disponível em: https://aws.amazon.com/pt/. Acesso em: 08 mar. 2022. 18

____. Tipos de instância do Amazon EC2. 2021. Disponível em: https://aws.amazon.com/pt/ec2/instance-types/. Acesso em: 06 ago. 2021. 18

____. Amazon API Gateway | Gerenciamento de APIs | Amazon Web Services. 2022. Disponível em: https://aws.amazon.com/pt/api-gateway/. Acesso em: 08 mar. 2022. 34

____. Amazon CloudFront - Entregue conteúdo com segurança com baixa latência e altas velocidades de transferência. 2022. Disponível em: https://aws.amazon.com/pt/cloudfront/. Acesso em: 08 mar. 2022. 35

____. Amazon Relational Database Service (RDS). 2022. Disponível em: https://aws.amazon.com/pt/rds/. Acesso em: 08 mar. 2022. 34

____. Amazon Route 53 - Uma forma confiável e econômica de encaminhar usuários finais para aplicativos de Internet. 2022. Disponível em: https://aws.amazon.com/pt/route53/. Acesso em: 08 mar. 2022. 34

___. Amazon S3 - Armazenamento de objetos construído para armazenar e recuperar qualquer volume de dados de qualquer local. 2022. Disponível em: https://aws.amazon.com/pt/s3/. Acesso em: 08 mar. 2022. 35

___. AWS Lambda - Execute código sem se preocupar com servidores ou clusters. 2022. Disponível em: https://aws.amazon.com/pt/lambda/. Acesso em: 08 mar. 2022. 34

___. Using the AWS Price List API - AWS Billing. 2022. Disponível em: https://docs.aws.amazon.com/awsaccountbilling/latest/aboutv2/price-changes.html. Acesso em: 08 mar. 2022. 38

CAMARGO, Alex; AMARAL, Érico; SILVA, Roger; HEINEN, Milton; PEREIRA, Francisco. Mineração de dados eleitorais: descoberta de padrões de candidatos a vereador na região da campanha do rio grande do sul. Revista Brasileira de Computação Aplicada, v. 8, p. 64-73, 04 2016. 23, 25

CAMILO, Cássio Oliveira; SILVA, João Carlos da. Mineração de dados: Conceitos, tarefas, métodos e ferramentas. Cluster Computing, rozero, p. 1-29, 2009. 18, 24, 26

CANTU, Ana. A História e o Futuro da Computação em Nuvem. 2021. Disponível em: https://www.dell.com/learn/br/pt/brbsdt1/sb360/social_cloud. Acesso em: 03 ago. 2021. 17

CETAX. Data Mining: O que é, conceito e definição. 2021. Disponível em: https://www.cetax.com.br/blog/data-mining/. Acesso em: 08 ago. 2021. 19

COSTA, Breno GS; REIS, Marco Antonio Sousa; ARAÚJO, Aletéia PF; SOLIS, Priscila. Performance and cost analysis between on-demand and preemptive virtual machines. In: CLOSER. [S.l.: s.n.], 2018. p. 169-178. 18

DEBASTIANI, Carlos Alberto. Definindo Escopo em Projetos de Software. [S.l.]: ovatec. ISBN 978-85-7522-429-8, 2015. 39, 40

DJANGO. Django documentation. 2021. Disponível em: https://docs.djangoproject.com/en/3.2/. Acesso em: 25 jul. 2021. 30, 33, 36

___. Django Model View Template. 2021. Disponível em: https://www.onlinetutorialspoint.com/django/django-model-view-template-mvt-overview.html. Acesso em: 08 mar. 2022. 32

FIELDING, R. T. Rest: Arrchitectural styles and the designof network-based software architectures. University of California, Irvine, USA, 2000. Disponível em: https://www.ics.uci.edu/~fielding/pubs/dissertation/rest_arch_style.htm. 29

FLEXERA. Flexera 2020 State of the Cloud Report. 2021. Disponível em: https://info.flexera.com/SLO-CM-REPORT-State-of-the-Cloud-2020?utm_source=Blog&utm_medium=Blog&utm_campaign=. Acesso em: 05 ago. 2021. 18

FONSECA, George Henrique Godim da. Ciência de dados. 2020. Disponível em: http://professor.ufop.br/sites/default/files/george/files/2020-2_apostila_cdd003.pdf. Acesso em: 08 mar. 2022. 38

GEIGER, Kyle. Inside ODBC. [S.l.]: Microsoft Press, 1995. 33

GOLDSCHMIDT, Ronaldo; PASSOS, Emmanuel; BEZERRA, Eduardo. Data Mining: Conceitos, técnicas, algoritmos, orientações e aplicações. 2. ed. [S.l.]: Elsevier, 2015. ISBN 978-85-352-7822-4. 24, 25

GOOGLE. Serviços de computação em nuvem - Google Cloud. 2021. Disponível em: https://cloud.google.com/. Acesso em: 08 mar. 2022. 18

IEEE Standard for an Architectural Framework for the Internet of Things (IoT). IEEE Std 2413-2019, p. 1-269, 2020. 30

IPM. História da computação em nuvem: como surgiu a cloud computing? 2021. Disponível em: https://www.ipm.com.br/administracao-geral/historia-da-computacao-em-nuvem-como-surgiu-a-cloud-computing/. Acesso em: 03 ago. 2021. 17

IPSENSE. On Premise e Cloud Servers: entenda as principais diferenças. 2021. Disponível em: https://www.ipsense.com.br/blog/on-premise-e-cloud-servers-entenda-as-principais-diferencas/. Acesso em: 04 ago. 2021. 17

KDNUGGETS. Mineração de dados: o que é, para que serve e como fazer? 2015. Disponível em: https://www.kdnuggets.com/polls/2015/analytics-data-mining-data-science-software-used.html. Acesso em: 23 jan. 2021. 26

KURASOV, DA. Computer-aided manufacturing: Industry 4.0. In: IOP PUBLISHING. IOP Conference Series: Materials Science and Engineering. [S.l.], 2021. v. 1047, n. 1, p. 012153. 17

LEFF, Avraham; RAYFIELD, James T. Web-application development using the model/-view/controller design pattern. In: IEEE. Proceedings fifth ieee international enterprise distributed object computing conference. [S.l.], 2001. p. 118-127. 31

LEHMAN, M.M. On understanding laws, evolution, and conservation in the large-program life cycle. Journal of Systems and Software, v. 1, p. 213-221, 1979. ISSN 0164-1212. Disponível em: https://www.sciencedirect.com/science/article/pii/0164121279900220. 17

MCKINNEY, Wes et al. Pandas: a foundational python library for data analysis and statistics. Python for high performance and scientific computing, Dallas, TX, v. 14, n. 9, p. 1-9, 2011. 27

MICROSOFT. Calculadora do TCO (Custo Total de Propriedade) | Microsoft Azure. 2021. Disponível em: https://azure.microsoft.com/pt-br/pricing/tco/calculator/. Acesso em: 08 mar. 2022. 18

____. Normalizar Dados: referência de componente - Azure Machine Learning | Microsoft Docs. 2021. Disponível em: https://docs.microsoft.com/pt-br/azure/machine-learning/component-reference/normalize-data. Acesso em: 08 mar. 2022. 41

MITRA SUSHMITA; ACHARYA, Tinku. Data mining: Multimedia, soft computing and bioinformatics. Hoboken: John Wiley & Sons, Inc, 2003. 25, 26

MYSQL. MySQL 8.0 Reference Manual. 2022. Disponível em: https://dev.mysql.com/doc/refman/8.0/en/. Acesso em: 08 mar. 2022. 26

NEUMANN, Andy; LARANJEIRO, Nuno; BERNARDINO, Jorge. An analysis of public rest web service apis. IEEE Transactions on Services Computing, v. 14, n. 4, p. 957-970, 2021. 29

NUTANIX. Cloud Usage Report 2020 for AWS & Azure. 2021. Disponível em: https://www.nutanix.com/go/cloud-usage-report-2020#cloud-form. Acesso em: 05 ago. 2021. 18

OLIPHANT, Travis E. Guide to numpy. MDTDR, p. 1-378, 2006. 26

ORENSTEIN, David. Application Programming Interface. 2000. Disponível em: https://www.computerworld.com/article/2593623/application-programming-interface.html. Acesso em: 15 jul. 2021. 28

PARR, Terence John. Enforcing strict model-view separation in template engines. In: Proceedings of the 13th international conference on World Wide Web. [S.l.: s.n.], 2004. p. 224-233. 31

PYTHON. Documentação Python 3.9.10. Outubro 2020. https://docs.python.org/3.9/. (Acesso em 01/29/2022). 26

SAATY, Thomas L. How to make a decision: the analytic hierarchy process. European journal of operational research, Elsevier, v. 48, n. 1, p. 9-26, 1990. 31

SFERRA, Heloisa Helena; CORRêA ÂngeoanM. C. Jorge. Conceitos e aplicações de data mining. p. 1-16, 2003. 26

SHARMA, P.; IRWIN, D.; SHENOY. Resource management for transient cloud servers. proceedings of the acm on measurement. Portfoliodriven, p. 5, 2017. 23

SQLALCHEMY. Sqlalchemy documentation. sqlalchemy, p. 1, 2021. 28, 33, 41

____. SQLAlchemy Documentation. Janeiro 2022. Https://www.sqlalchemy.org/. (Acesso em 02/04/2022). 41

UFOPA. Data Mining. 2013. Disponível em: http://www.ufopa.edu.br/lsd/index. php?option=com_content&view=article&id=8:beginners&catid=19&Itemid= 260. Acesso em: 25 jul. 2021. 24

Vantage Slack Community. Amazon EC2 Instance Comparison. Dezembro 2022. https://instances.vantage.sh/. (Acesso em 01/07/2022). 18

VENNERS, Bill. rthogonality and the dry principle a conversation with andy hunt and dave thomas. artima, artima, p. 1-2, 2003. 36

VERAS, Manoel. Computação em Nuvem. [S.l.]: Editora Brasport, 2015. 17, 31

VOXIMPLANT. What is a rest api? 2020. Disponível em: https://voximplant.com/ blog/what-is-a-rest-api. Acesso em: 27 nov. 2020. 29

WES, McKinney. pandas: a foundational python library for data analysis and statistics. portaldata, p. 1-9, 2011. 26, 28

____. Zappa 2019a. https://www.zappa.io/, 2019. 47