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

MATHEUS COIMBRA MORAES

CONTROLE DE INVENTÁRIO USANDO INTERNET DAS COISAS E SERVIÇOS EM NUVEM

São Luis
2018

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

MATHEUS COIMBRA MORAES

CONTROLE DE INVENTÁRIO USANDO INTERNET DAS COISAS E SERVIÇOS EM NUVEM

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. Ma. Alana de Araújo Oliveira Meireles Teixeira

São Luis
2018

Ficha gerada por meio do SIGAA/Biblioteca com dados fornecidos pelo(a) autor(a).
Núcleo Integrado de Bibliotecas/UFMA

Moraes, Matheus Coimbra.
CONTROLE DE INVENTÁRIO USANDO INTERNET DAS COISAS E SERVIÇOS EM NUVEM / Matheus Coimbra Moraes. - 2018.
61 f.

Coorientador(a): Prof. Dr. Mário Antônio Meireles Teixeira.
Orientador(a): Prof. Ms. Alana de Araújo Oliveira Meireles Teixeira.
Monografia (Graduação) - Curso de Ciência da Computação, Universidade Federal do Maranhão, Centro de Ciências Exatas e Tecnologia, 2018.

1. Aws. 2. Computação em nuvem. 3. Controle patrimonial. 4. Internet das coisas. 5. Raspberry pi. I. Teixeira, Prof. Dr. Mário Antônio Meireles. II. Teixeira, Prof. Ms. Alana de Araújo Oliveira Meireles. III. Título.

MATHEUS COIMBRA MORAES

CONTROLE DE INVENTÁRIO USANDO INTERNET DAS COISAS E SERVIÇOS EM NUVEM

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.

Data da Defesa: 11 de julho de 2018
Conceito:

Banca Examinadora

Alana de J. O. Meireles Teixeira

Prof. Ma. Alana de Araújo Oliveira Meireles Teixeira
Universidade Federal do Maranhão
Orientadora

Mário Teixeira

Prof. Dr. Mário Antônio Meireles Teixeira
Universidade Federal do Maranhão
Coorientador

Carlos de Salles Soares Neto

Prof. Dr. Carlos de Salles Soares Neto
Universidade Federal do Maranhão
Professor

Davi Viana dos Santos

Prof. Dr. Davi Viana dos Santos
Universidade Federal do Maranhão
Professor

São Luis
2018

Este trabalho é dedicado a todos os professores da Universidade Federal do Maranhão que de alguma forma contribuíram em minha graduação.

AGRADECIMENTOS

Agradeço, primeiramente, a Deus pela dádiva da vida, por Seu amor infinito, por todas as conquistas até aqui e por se fazer presente em cada obstáculo da minha vida. Por ter colocado em meu caminho pessoas tão especiais, que não mediram esforços em me ajudar durante a graduação e realização deste trabalho de conclusão de curso.

Agradeço também, especialmente, à minha orientadora, Prof.ª Ms. Alana de Araújo Oliveira Meireles Teixeira, de quem obtive todo suporte necessário, demonstrando grande paciência e sabedoria no norteamento de todas as etapas deste trabalho.

Ao meu coorientador, Prof. Dr. Mário Antônio Meireles Teixeira, pelo qual, desde o começo da minha graduação, sempre tive enorme respeito e admiração. Pelo auxílio, direcionamento e incentivos.

Aos meus pais, Maria Gorete Coimbra e José de Alberto Moraes, pelo apoio e incentivo que me permitiram chegar até aqui, por serem incansáveis na dedicação em me guiar nos caminhos corretos.

Ao meu irmão, Rômulo Coimbra, por desde o início da minha graduação, mostrar o caminho para obter êxito em mais essa etapa da minha vida. E, por me mostrar uma realidade mais amadurecida do mundo.

À minha segunda mãe, Euzamar Lima, que auxiliou na minha criação, com amor incondicional. Por me nortear a caminhos que me ajudaram a amadurecer.

Aos meus amigos pelo apoio e paciência que tiveram todos os momentos. Especialmente, agradeço à Candida Braga por me acompanhar e incentivar em grande parte da minha graduação, e me propiciar os melhores anos da minha vida.

“Nossas dúvidas são traidoras e nos fazem perder o que, com frequência, poderíamos ganhar, por simples medo de arriscar.”
(William Shakespeare)

RESUMO

O desenvolvimento deste trabalho tem como objetivo a construção de um protótipo para controle de inventário dos bens físicos da Universidade Federal do Maranhão com uso de Internet das Coisas e serviços em Nuvem, podendo, posteriormente, servir para controle de inventário de outras instituições. Para realização da aplicação web, é utilizado o padrão de desenvolvimento MVC (do inglês "Model, View and Controller") utilizando a tecnologia da linguagem Python, em conjunto com o microcomputador Raspberry Pi 3, na qual, este tem como papel principal, o cadastro dos bens físicos através de um módulo de leitura de código de barra; a identificação dos professores também com uso de serviços em Nuvem, e autenticação dos funcionários, através de um módulo de leitura de RFID (do inglês "Radio-Frequency IDentification"), que será o usuário final do sistema. Toda a plataforma do sistema ficará hospedado no serviço de nuvem da Amazon, conhecida como AWS (Amazon Web Services).

Palavras-chave: controle patrimonial, internet das coisas, raspberry pi, computação em nuvem, aws.

ABSTRACT

The development of this work has as objective the construction of a prototype to control inventory of the physical assets of the Federal University of Maranhão with use of Internet of Things and services in Cloud, and can later serve to control inventory of other institutions. To implement the web application, the MVC (Model, View and Controller) development standard is used using the Python language technology, together with the Raspberry Pi 3 microcomputer, in which the main role is the registration of physical goods through a barcode reading module; the identification of teachers also using Cloud services, and employee authentication through an RFID (Radio-Frequency Identification) module, which will be the end user of the system. The entire system platform will be hosted on the Amazon cloud service, known as AWS (Amazon Web Services).

Keywords: patrimonial control, internet of things, raspberry pi, cloud computing, aws.

LISTA DE ILUSTRAÇÕES

LISTA DE CÓDIGOS

LISTA DE ABREVIATURAS E SIGLAS

NIST	National Institute of Standards and Technology
IoT	Internet of things
RFID	Radio-Frequency IDentification
ICT	Information and Communication Technologies
M2M	Machine-to-Machine
ESB	Enterprise Service Bus
API	Application Programming Interface
SOA	Service-Oriented Architecture
WSN	Wireless Sensor network
RMI	Remote Method Invocation
XML	Extensible Markup Language
REST	Representational State Transfer
SPE	Stream Processing Engines
CDN	Content Distribution Network
CoAP	Constrained Application Protocol
MQTT	MQ Telemetry Transport
HTTP	Hypertext Transfer Protocol
IAB	Internet Architecture Board
JSON	JavaScript Object Notation
PC	Personal Computer
RISC	Reduced Instruction Set Computer
USB	Universal Serial Bus
UFMA	Universidade Federal do Maranhão
EC2	Elastic Compute Cloud
ONS	Operador Nacional do Sistema Elétrico
MB	Megabyte
SSH	Secure Shell
UML	Unified Modeling Language

MAC	Macintosh
TI	Tecnologia da Informação
MVC	Model, View and Controller
ARPANET	Advanced Research Projects Agency Network
IP	Internet Protocol address
DNS	Domain Name System

SUMÁRIO

1 INTRODUÇÃO

A grande quantidade de dados coletado pelos mais diversos tipos de sistema de informação, fez com que novas tecnologias surgissem para armazenar e processar esses dados de forma segura, econômica e sem grandes esforços para implantação. Com o passar do tempo, essas tecnologias foram se aperfeiçoando cada vez mais, e passaram a oferecer mais serviços para as empresas e usuários, mudando a forma de pensar de diversas empresas e mudando o cotidiano de seus usuários. Como consequência desse desenvolvimento tecnológico contínuo, a web passou a atuar como a principal fornecedora de serviços devido a aplicações mais dinâmicas.

Nesse novo contexto, surgiram novas áreas de pesquisas e novas tecnologias, entre elas a Computação em Nuvem e a Internet das Coisas. A Internet das Coisas é uma revolução tecnológica que tem sido uma grande aposta nas grandes companhias, que busca a conexão de dispositivos eletrônicos, que utilizamos em nosso dia a dia, à internet. Esses dispositivos, e sistemas web dinâmicos coletam uma grande quantidade dados para serem processados, e um dos principais serviços usado para armazenamento e possível inferência sobre esses dados são os serviços oferecidos pela Computação em Nuvem.

O conceito de Computação em Nuvem gera grande confusão por não se tratar de uma nova tecnologia, mas da evolução e convergência de várias outras tecnologias e conceitos. Uma maneira de simplificar, é entender sua definição como uma referência a aplicações e serviços que executam em uma rede distribuída utilizando recursos virtualizados e acessados pela Internet.

Com o aumento da demanda, e a consequente expansão necessária para o atingimento dos interesses coletivos, têm feito com que entidades públicas procurem cada vez mais por ferramentas que possam vir a contribuir na realização de um controle patrimonial cada vez mais eficiente e eficaz dentro das instituições.

Devido ao fato dos bens de uma Universidade Federal estarem a serviço do atingimento dos interesses coletivos, não possuindo proprietário nominal, muitos funcionários da instituição acham-se no direito de usufruir e deslocar os mesmos conforme sua vontade, o que acaba por constituir um grande problema para o setor de patrimônio, que muitas vezes precisa ir a campo para detectar o destino que foi dado a estes bens e sob que condições o mesmo está submetido. Isso torna substancial a importância da proteção e controle do patrimônio público (VIECELLI, 2014; MATIAS, 2015).

Dentre os bens de uma Universidade Federal, destacam-se os bens físicos como retroprojetores e computadores. Uma ferramenta que facilite o processo de inventário pela utilização de códigos de barra nas plaquetas de identificação acelera e facilita o trabalho do gestor patrimonial.

Na universidade Federal do Maranhão, para fazer o empréstimo de um equipamento para um professor ou um aluno, é preciso preencher um formulário com os seguintes dados: nome do equipamento, data de empréstimo, a sala onde o equipamento estará localizado, o nome da pessoa que solicita o empréstimo e, do funcionário que é responsável por fazer o empréstimo. No caso do aluno que solicita

algum equipamento emprestado, o seu cartão de identificação da UFMA fica retido com o funcionário, e é devolvido após devolução do equipamento. Todo esse procedimento é dispendioso por requerer o preenchimento de um grande formulário e, por ser mais difícil de manter tais formulários de forma organizada para um possível processamento por parte do funcionário. Como o rastreamento do equipamento, por exemplo.

Dentro desse contexto, este trabalho propõe a construção de um protótipo que automatize o processo de empréstimo feito por parte do funcionário, além de manter o registro de todos os equipamentos disponíveis para empréstimo, resultando em um maior controle do patrimônio da instituição e agilização do processo de empréstimo.

1.1 Objetivos

1.1.1 Objetivo Geral

O objetivo deste trabalho é a construção de um protótipo para automatizar o processo de empréstimo de equipamentos da Universidade Federal do Maranhão e, controlar os equipamentos disponíveis para empréstimo da instituição.

1.1.2 Objetivos Específicos

- Controle dos bens físicos.
- Automatização do processo de empréstimo.
- Identificação, controle e manutenção do patrimônio da instituição usando um Raspberry Pi 3 e um leitor de código de barras.
- Identificação, controle e manutenção dos funcionários que utilizarão o sistema.
- Identificação do funcionário e professor por um módulo leitor Rfid conectado a um Raspberry Pi 3.
- Utilização dos serviços da AWS para coletar dados do RaspBerry Pi e para a implantação do protótipo.

1.2 Organização do Trabalho

O capítulo 2 explica o conceito de computação em nuvem, assim como os diferentes modelos de serviços oferecidos pelos provedores de Nuvem. Também aborda os tipos de Nuvem e, os benefícios e vantagens da utilização da computação em Nuvem.

O capítulo 3 apresenta o conceito de Internet das Coisas, as arquiteturas, e o uso de outras tecnologias que permitiram o crescimento de aplicativos IoT.

O capítulo 4 mostra a arquitetura proposta para desenvolvimento do protótipo.

O capítulo 5 apresenta as especificações e modelagem da aplicação web desenvolvida.

2 COMPUTAÇÃO EM NUVEM

A ampla disponibilidade de rede de baixa latência e alta largura de banda, permitiu à internet a prestação de serviços como redes sociais, entrega de conteúdo e, e-commerce em uma escala sem precedentes. Esta tendência tecnológica levou o desenvolvimento da computação em nuvem, um paradigma que aproveita as capacidades massivas dos data centers para apoiar a entrega de serviços online de maneira eficiente e econômica. Em um ambiente de computação em nuvem, o papel tradicional dos provedores de serviços é dividida em duas: *provedores de nuvem* que possuem o data center físico e os recursos que aluga (por exemplo, máquinas virtuais ou VMs) para provedores de serviços; e *provedores de serviços* que usam os recursos alugados por provedores de nuvem para executar aplicativos. Ao alavancar as economias de escala de centro de dados, a computação em nuvem pode fornecer uma redução significativa nos gastos operacionais. Ao mesmo tempo, também suporta novas aplicações, como análise de big data (por exemplo, MapReduce) que processam grandes quantidades de dados de foma escalável e eficiente. O surgimento da computação em nuvem causou um profundo impacto no desenvolvimento da industria de TI nos últimos anos. Enquanto grandes empresas como Google, Amazon, Facebook e Microsoft têm desenvolvido suas próprias plataformas e tecnologias de nuvem, muitas empresas pequenas também estão adotando a computação em nuvem, aproveitando o software de código aberto e implantando serviços em nuvens públicas.

No entanto, apesar da ampla adoção da computação em nuvem na indústria, as atuais tecnologias de nuvem ainda estão longe de atingir todo o seu potencial. Na verdade, computação em nuvem era conhecida como uma buzzword por vários anos, e muitas empresas de TI estavam incertas sobre como fazer um investimento bem-sucedido em computação em nuvem. Felizmente, com a atração significativa da indústria e da academia, a computação em nuvem está evoluindo rapidamente, com avanços em quase todos os aspectos, desde o projeto da arquitetura de data center, agendamento e gerenciamento de recursos, virtualização de servidores e redes, armazenamento, frameworks de programação, gerenciamento de energia, preços, conectividade de serviços para segurança e privacidade (SOSINSKY, 2015).

O NIST (do inglês "National Institute of Standards and Technology") forneceu uma definição relativamente padrão e amplamente aceita de computação em nuvem da seguinte forma: “a computação em nuvem é um modelo para possibilitar acesso de rede onipresente e adequado a um conjunto compartilhado de recursos computacionais ajustáveis (por exemplo, servidores, redes, serviços e aplicativos) que podem ser provisionados de forma rápida, e liberados com pequeno esforço de gerenciamento ou interação com o provedor de serviços”.

O NIST definiu ainda cinco características essenciais, três modelos de serviço e quatro modelos de implantação, para computação em nuvem. As cinco características essenciais incluem o seguinte:

1. Autoatendimento sob demanda, que afirma que um consumidor (por exemplo, um provedor de serviços) pode adquirir recursos com base na demanda de

serviços;

2. Acesso de rede amplo, que afirma que os serviços de nuvem podem ser acessados remotamente de plataformas de clientes heterogêneos (por exemplo, telefones celulares);

3. Pool de recursos, onde os recursos são reunidos e compartilhados pelos consumidores em um arquitetura multi-tenant;

4. Elasticidade rápida, que afirma que os recursos da nuvem podem ser provisionados rapidamente e liberados com envolvimento humano mínimo;

5. Serviço medido, que afirma que os recursos são controlados (e possivelmente precificados), aproveitando uma capacidade de medição (por exemplo, pay-per-use) que é apropriada para o tipo de serviço.

Essas características fornecem uma imagem relativamente precisa de como os sistemas de computação em nuvem deve se apresentar. Deve ser mencionado que nem todo sistema de computação em nuvem possue todas as cinco características listadas anteriormente. Por exemplo, em uma nuvem privada, onde o provedor de serviços possui o data center físico, o recurso de medição pode não ser necessário porque não há necessidade de limitar o uso de recursos do serviço, a menos que ele atinja os limites de capacidade do data center. No entanto, apesar da definição e das características mencionadas, a computação em nuvem ainda pode ser realizada de várias maneiras, e, portanto, pode-se argumentar que a definição ainda não é suficientemente precisa. Hoje, a computação em nuvem comumente refere-se a um modelo de computação em que os serviços são hospedados usando recursos em data centers, e entregue aos usuários finais pela Internet.

2.1 Tipos de Nuvem

A computação em nuvem é geralmente separada em dois conjuntos distintos de modelos:

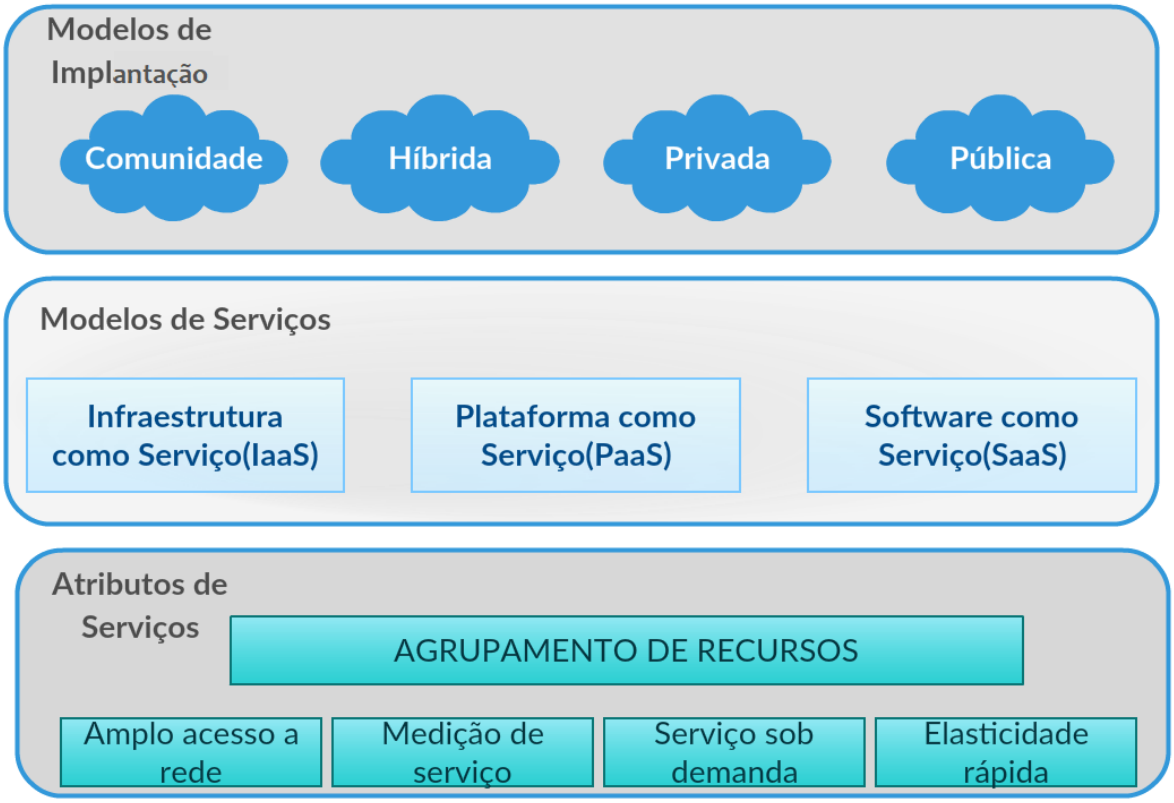
- **Modelos de Implantação**: refere-se à localização e ao gerenciamento dos recursos da infraestrutura da nuvem.
- **Modelos de Serviço**: consistem nos tipos específicos de serviços que o usuário pode acessar em uma plataforma de computação em nuvem.

2.1.1 O modelo NIST

O governo dos Estados Unidos é o maior consumidor de serviços de computação e, portanto, um dos maiores usuários de redes de computação em nuvem. O Instituto Nacional de Padrões e Tecnologia (NIST) dos EUA tem um conjunto de definições de trabalho que separam a computação em nuvem em modelos de serviço e modelos de implantação. Esses modelos e sua relação com características essenciais da computação em nuvem são mostrados na figura 1.

O modelo originalmente do NIST não exigia uma nuvem para usar a virtualização para reunir recursos, nem exigia que uma nuvem oferecesse suporte a Multi-inquilino nas primeiras definições de computação em nuvem. Uma arquitetura multi-inquilinos é fundamental para um ambiente em nuvem, já que possibilita que múltiplos inquilinos (empresas/clientes) compartilhem os mesmos meios físicos, mas continuem logicamente isolados (TAURION, 2010). A versão mais recente da definição do NIST exige que as redes de computação em nuvem usem a virtualização e ofereçam suporte a multi-inquilino.

Figura 1 – Definições de computação em nuvem do NIST.



O modelo de nuvem desenvolvido pela NIST não aborda os muitos serviços intermediários, como transações ou Agente de Serviços , provisionamento, integração e serviços de interoperabilidade que constituem a base de muitas discussões sobre computação em nuvem (FONSECA, 2011).

2.1.2 Modelos de Implantação

Um modelo de implantação define o propósito de uma núvem e a natureza de como a nuvem está localizada.

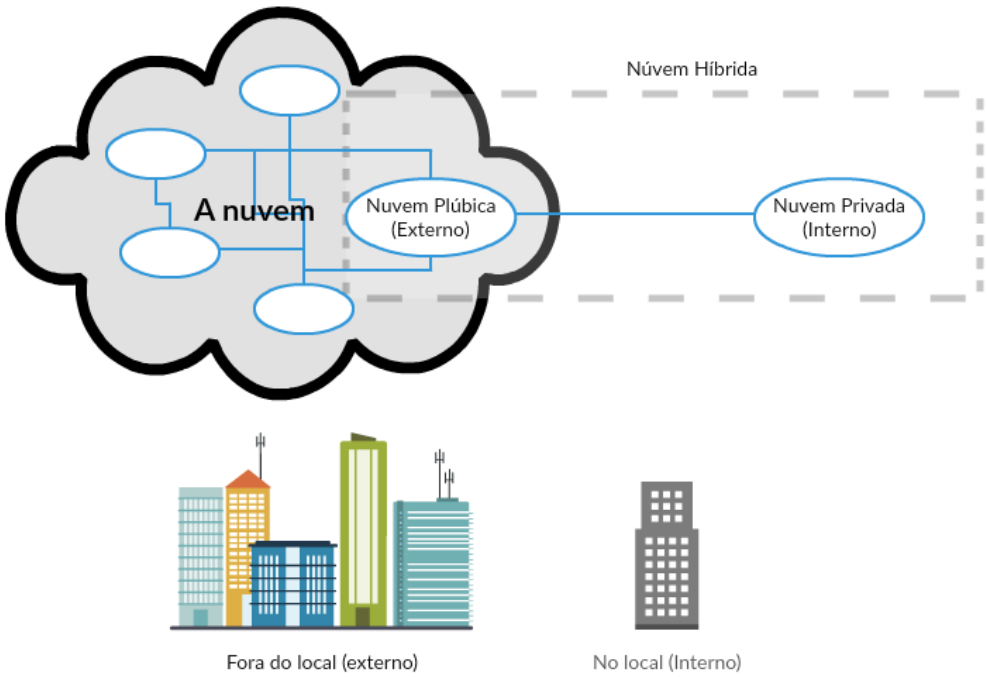
As definições do NIST para as quatro modelos de implantação são as seguintes:

- **Nuvem Pública**: A infraestrutura de nuvem pública está disponível para uso público, alternativamente, para um grande grupo da indústria e é de propriedade de uma organização que vende serviços em nuvem.

- **Nuvem Privada**: A infraestrutura de nuvem privada é operada para uso exclusivo de uma organização. A nuvem pode ser gerenciada por essa organização ou por terceiros. Nuvens Privadas podem estar no local ou fora da organização.
- **Nuvem Híbrida**: Uma nuvem híbrida combina várias nuvens (privada, pública), onde essas nuvens mantêm suas identidades únicas, mas são unidas como uma unidade. Uma nuvem híbrida pode oferecer acesso padronizado ou proprietário a dados e aplicativos, bem como portabilidade de aplicativos.
- **Nuvem Comunitária**: Uma nuvem comunitária é aquela em que a nuvem foi organizada para servir uma função ou propósito comum. Pode ser para uma organização ou para várias organizações, mas elas compartilham preocupações comuns, como sua missão, políticas, segurança, necessidades de conformidade normativa e assim por diante. Uma nuvem comunitária pode ser gerenciada pela(s) organização(ões) constituinte(s) ou por um terceiro.

A figura 2 mostra os diferentes locais que as nuvens podem ser utilizadas.

Figura 2 – Locais de implantação para diferentes tipos de nuvens



2.1.3 Modelos de Serviço

No modelo de implantação, diferentes tipos de nuvem são uma expressão da maneira como a infraestrutura é implantada. Pode-se pensar na nuvem como o limite entre o ponto em que a rede, o gerenciamento e as responsabilidades de um cliente terminam e o provedor de serviços em nuvem começa. À medida que a computação em nuvem se desenvolveu, diferentes fornecedores oferecem nuvens com diferentes

serviços associados a elas. O portfólio de serviços oferecidos adiciona outro conjunto de definições chamado modelo de serviço.

Existem muitos modelos de serviços diferentes descritos na literatura, na qual todos possuem a seguinte forma:

Xaas, ou ”<Alguma coisa> como um Servico”

Três tipos de serviços são universalmente aceitos:

- **Infraestrutura como um Serviço**: IaaS fornece máquinas virtuais, armazenamento virtual, infraestrutura virtual e outros ativos de hardware como recursos que os clientes podem provisionar.

  O provedor de serviços IaaS gerencia toda a infraestrutura, enquanto o cliente é responsável por todos os outros aspectos da implantação. Isso pode incluir o sistema operacional, aplicativos e interações do usuário com o sistema.

- **Plataforma como um Serviço**: PaaS fornece máquinas virtuais, sistemas operacionais, aplicativos, serviços, estruturas de desenvolvimento, transações e estruturas de controle.

  O cliente pode implantar seus aplicativos na infraestrutura de nuvem ou usar aplicativos que foram programados usando idiomas e ferramentas que são suportados pelo provedor de serviços de PaaS. O provedor de serviços gerencia a infraestrutura em nuvem, os sistemas operacionais e o software de ativação. O cliente é responsável por instalar e gerenciar o aplicativo que está implementando.

- **Software como um Serviço**: SaaS é um ambiente operacional completo com aplicações, gerenciamento, e a interface do usuário.

  No modelo SaaS, o aplicativo é fornecido ao cliente por meio de uma interface de ”thin client” (geralmente um navegador), e a responsabilidade do cliente começa e termina com a entrada e o gerenciamento de dados e interação do usuário. Tudo, desde o aplicativo até a infraestrutura, é de responsabilidade do fornecedor.

Os três modelos de serviços diferentes tomados em conjunto passaram a ser conhecidos como o modelo SPI de computação em nuvem. Existem muitos outros modelos de serviço como: StaaS, Storage as a Service; Gaas, Jogos como Serviço, IdaaS, Identidade como Serviço; CmaaS, Compliance como um serviço; e assim por diante. No entanto, os serviços da SPI englobam todas as outras possibilidades.

A figura 3 mostra os diferentes modelos de serviços e a figura 4 mostra de provedores dos três tipos de modelos de serviços mais conhecidos.

Figura 3 – Modelos de Serviços

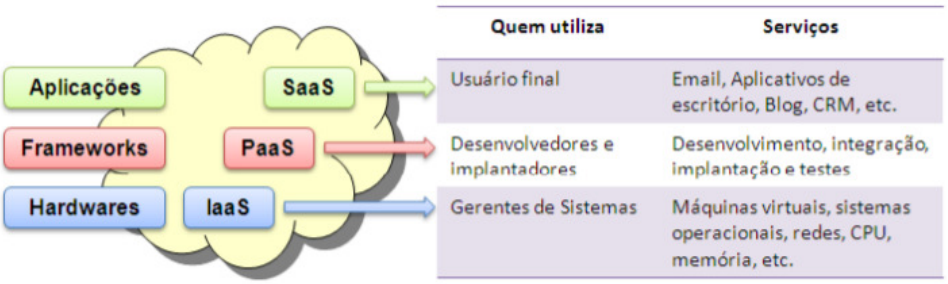
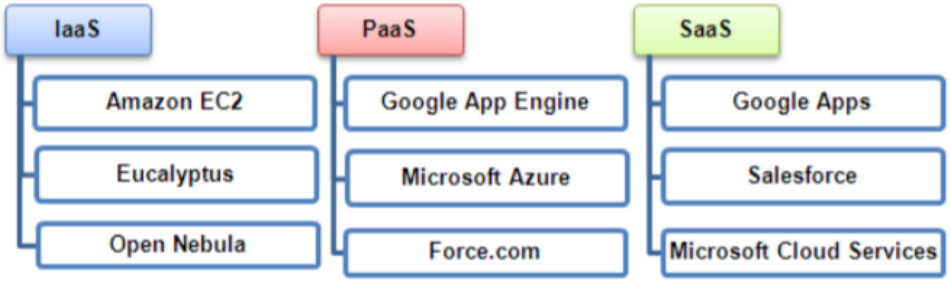


Figura 4 – Exemplo de provedores de serviços em nuvem



2.2 Benefícios da Computação em Nuvem

Alguns dos principais benefícios do uso dos serviços em nuvem são listados a seguir:

- **Baixo Custo**: Como as redes em nuvem operam com maior eficiência e com maior utilização, reduções significativas de custos são frequentemente encontradas.
- **Facilidade de Utilização**: Dependendo do tipo de serviço oferecido, pode-se descobrir que o usuário não precisa de licenças de hardware ou software para implementar seu serviço.
- **Qualidade de Serviço**: A Qualidade de Serviço (QoS) é algo o usuário pode obter sob contrato do seu fornecedor.
- **Confiabilidade**: A escala das redes de computação em nuvem e sua capacidade de fornecer balanceamento de carga e a tolerância a falhas os torna altamente confiáveis, geralmente muito mais confiáveis do que pode alcançar em uma única organização.
- **Gerenciamento de TI terceirizado**: Uma implantação de computação em nuvem permite que outra pessoa gerencie sua infra-estrutura de computação

enquanto o usuário gerencia seus negócios. Na maioria dos casos, o utilizador obtém reduções consideráveis nos custos da equipe de TI.

- **Manutenção e atualização simplificada**: Como o sistema é centralizado, o utilizador pode aplicar facilmente patches e atualizações. Isso significa que seus usuários sempre têm acesso a versão mais recente do software.
- **Baixa barreira à entrada**: Em particular, os gastos iniciais de capital são dramaticamente reduzido. Na computação em nuvem, qualquer um pode ser um gigante a qualquer momento.
- **Segurança**: Em casos como uma enchente, um incêndio, terremoto ou uma falha na energia elétrica, os dados em nuvem são conservados. Isso porque os servidores costumam ficar a milhares de quilômetros de distância entre si. Portanto, se algum servidor ficar indisponível, os sistemas e arquivos armazenados continuarão disponíveis (ABC, 2014).

3 INTERNET DAS COISAS

Após quatro décadas do advento da Internet pela ARPANET (Advanced Research Projects Agency Network), o termo ”Internet” refere-se à vasta categoria de aplicativos e protocolos construídos sobre redes de computadores sofisticados e interconectados, atendendo a bilhões de usuários em todo o mundo. De fato, estamos no começo de uma era emergente onde comunicação e conectividade onipresentes não são mais um sonho nem um desafio. Posteriormente, o foco mudou para uma integração perfeita de pessoas e dispositivos para convergir o reino físico com ambientes virtuais criados pelo homem, criando a chamada utopia da Internet das Coisas (IoT).

Um olhar mais detalhado desse fenômeno revela dois importantes pilares da IoT: “Internet” e “Coisas”. Embora pareça que cada objeto capaz de se conectar à Internet caia na categoria de “Coisas”, essa notação é usada para abranger um conjunto mais genérico de entidades, incluindo dispositivos inteligentes, sensores, seres humanos e qualquer outro objeto que esteja ciente do seu contexto e é capaz de se comunicar com outras entidades, tornando-o acessível a qualquer hora, em qualquer lugar. Isso implica que os objetos precisam estar acessíveis sem restrições de tempo ou local.

A conectividade omnipresente é um requisito crucial da IoT e, para isso, os aplicativos precisam oferecer suporte a um conjunto diversificado de dispositivos e protocolos de comunicação, desde sensores minúsculos capazes de detectar e relatar um fator desejado até servidores back-end poderosos usados para análise de dados e extração de conhecimento. Isso também requer a integração de dispositivos móveis, dispositivos de borda como roteadores e hubs inteligentes, e humanos como controladores.

Decisões de negócios significativas foram tomadas por grandes empresas como Google, Apple e Cisco para se posicionarem no cenário da IoT. Os operadores de telecomunicações consideram que o Machine-to-Machine (M2M) e a Internet das coisas estão se tornando um foco central de negócios, relatando crescimento no número de objetos conectados em suas redes. Fábricas de dispositivos, por exemplo, sobre dispositivos vestíveis, antecipam um novo segmento de negócios para uma adoção mais ampla da IoT.

Em consonância com este desenvolvimento, a maioria dos governos na Europa, na Ásia e nas Américas consideram a Internet das Coisas uma área de inovação e crescimento. Embora grandes empresas em algumas áreas de aplicação ainda não reconheçam o potencial, muitos delas prestam muita atenção ou até mesmo aceleraram o ritmo, cunhando novos termos para a IoT e adicionando mais componentes para ele. Além dos usuários finais no domínio privado e empresarial adquiriram hoje em dia uma competência significativa para lidar com dispositivos e aplicativos em rede.

Inicialmente, a identificação por radiofrequência (RFID) costumava ser a tecnologia dominante por trás do desenvolvimento da IoT, mas com novas conquistas tecnológicas, redes de sensores sem fio (WSN) e dispositivos habilitados para

Bluetooth aumentaram a adoção da tendência da IoT. Essas tecnologias e aplicativos IoT foram extensivamente pesquisados, no entanto, menos atenção tem sido dado às características e requisitos exclusivos da IoT, como escalabilidade, suporte à heterogeneidade, integração total e processamento de consultas em tempo real (BUYYA, 2016).

À medida que a Internet das Coisas continua a se desenvolver, mais potencial é estimado por uma combinação com abordagens e conceitos tecnológicos relacionados como Computação em Nuvem, Big Data, Robótica e tecnologias Semântica. A Internet das Coisas deixou de ser uma visão futurista - às vezes com um certo grau de exagero - para uma realidade de mercado crescente (VERMESAN, 2014).

3.1 Arquiteturas IoT

Os blocos de construção da IoT são dispositivos sensoriais, chamada de serviço remoto, redes de comunicação e processamento de eventos com reconhecimento de contexto; estes existem há muitos anos. No entanto, o que a IoT tenta retratar é uma rede unificada de objetos inteligentes e seres humanos responsáveis por operá-los (se necessário), que são capazes de se comunicar de maneira universal e onipresente uns com os outros.

Quando se fala de um ambiente distribuído, a interconectividade entre entidades é um requisito crítico, e a IoT é um bom exemplo. Uma arquitetura de sistema holística para IoT precisa garantir a operação perfeita de seus componentes (a confiabilidade é considerada o fator de projeto mais importante na IoT) e vincular os reinos físico e virtual. Para conseguir isso, é necessário considerar cuidadosamente a possibilidade de projetar recuperação de falhas e escalabilidade. Além disso, como a mobilidade e a mudança dinâmica de localização se tornaram parte integrante dos sistemas de IoT com o uso disseminado de smartphones, as arquiteturas de ponta precisam ter um certo nível de adaptabilidade para lidar adequadamente com interações dinâmicas em todo o ecossistema.

As arquiteturas e modelos de referência fornecem uma visão geral de todo o sistema subjacente, portanto, sua vantagem sobre outras arquiteturas depende de fornecer um nível de abstração melhor e maior, o que, consequentemente, oculta restrições específicas e detalhes de implementação.

Diversos grupos de pesquisa propuseram arquiteturas de referência para IoT. A IoT-A se concentra no desenvolvimento e validação de uma arquitetura de rede IoT integrada e no apoio a blocos de construção, com o objetivo de ser “the European Lighthouse Integrated Project addressing the Internet-of-Things Architecture.” Projeto IoT-i relacionado ao projeto IoT-A mencionado anteriormente, concentra-se na promoção de soluções de IoT, capturando requisitos e interesses. A IoT-i visa alcançar objetivos estratégicos, tais como: criar uma visão estratégica e técnica conjunta para a IoT na Europa que englobe os setores atualmente fragmentados do domínio da internet das coisas e contribua para a criação de um ambiente economicamente sustentável e socialmente aceitável na Europa para tecnologias IoT.

Figura 5 – Arquitetura de referência para IoT

A figura 5 descreve um esboço de uma arquitetura de referência para IoT (FREMANTLE, 2015). Diferentes camadas de serviço e apresentação são mostradas nessa arquitetura. As camadas de serviço incluem processamento e análise de eventos, gerenciamento de recursos e descoberta de serviços, bem como agregação de mensagens e serviços ESB (Enterprise Service Bus) criados sobre camadas físicas e de comunicação. O gerenciamento de APIs, essencial para definir e compartilhar serviços do sistema, e painéis baseados na Web (ou aplicativos equivalentes de smartphone) para gerenciar e acessar essas APIs, também está incluído na arquitetura. Devido à importância do gerenciamento de dispositivos, da segurança e da imposição de privacidade em diferentes camadas e da capacidade de identificar objetos de maneira exclusiva e controlar seu nível de acesso, esses componentes são estendidos de forma independente nessa arquitetura.

3.1.1 Arquitetura baseada em SOA

Em IoT, a arquitetura orientada a serviços (SOA) pode ser imperativa para os provedores de serviços e usuários. SOA garante a interoperabilidade entre os dispositivos heterogêneos.Por exemplo, pode-se considerar uma SOA genérica consistindo de quatro camadas, com funcionalidades distintas, como segue:

- A camada de detecção é integrada com objetos de hardware disponíveis para detectar o status das coisas.
- A camada de rede é a infra-estrutura para suportar conexões sem fio ou com fio entre as coisas.

- Camada de serviço é para criar e gerenciar serviços exigidos por usuários ou aplicativos.
- Camada Interfaces consiste nos métodos de interação com usuários ou aplicativos.

Geralmente, em tal arquitetura, um sistema complexo é dividido em subsistemas que são fracamente acoplados e podem ser reutilizados posteriormente, proporcionando uma maneira fácil de manter todo o sistema, cuidando de seus componentes individuais (D TRIFA V, 2011). Isso pode garantir que, no caso de uma falha de componente, o restante do sistema (componentes) ainda possa operar normalmente. Isso é de imenso valor para o design eficiente de uma arquitetura de aplicativo IoT.

A SOA tem sido intensamente usada na rede de sensores sem fio (WSN), devido ao seu nível apropriado de abstração e vantagens relativas ao seu design modular. Trazendo esses benefícios para a IoT, a SOA tem o potencial de aumentar o nível de interoperabilidade e escalabilidade entre os objetos na IoT. Além disso, do ponto de vista do usuário, todos os serviços são resumidos em conjuntos comuns, eliminando a complexidade extra para o usuário lidar com diferentes camadas e protocolos. Além disso, a capacidade de construir serviços diversos e complexos compondo diferentes funções do sistema por meio da composição de serviços atende à natureza heterogênea da IoT, onde realizar cada tarefa requer uma série de chamadas de serviço em todas as entidades diferentes espalhadas por várias localizações.

3.1.2 Arquitetura orientada a API

Abordagens convencionais para o desenvolvimento de soluções orientadas a serviços usam o SOAP e o Remote Method Invocation (RMI) como meio de descrever, descobrir e chamar serviços; entretanto, devido à sobrecarga e complexidade impostas por essas técnicas, os métodos baseados em Web APIs e Representational State Transfer (REST) foram apresentados como soluções alternativas promissoras (CLARK, 2015). Os recursos necessários variam de largura de banda de rede para capacidade computacional e de armazenamento, e são acionados por conversões de dados de solicitação-resposta que ocorrem regularmente durante as chamadas de serviço. Formatos leves de troca de dados como o JSON podem reduzir a sobrecarga mencionada anteriormente, especialmente para dispositivos inteligentes e sensores com uma quantidade limitada de recursos, substituindo grandes arquivos XML usados para descrever serviços. Isso ajuda a usar o canal de comunicação e a processar o poder dos dispositivos com mais eficiência.

Da mesma forma, a criação de APIs para aplicativos IoT ajuda o provedor de serviços a atrair mais clientes, enquanto se concentra na funcionalidade de seus produtos, e não na apresentação. Além disso, é mais fácil habilitar a ocupação variada pelos recursos de segurança das APIs da Web modernas, como OAuth, APIs que, de fato, são capazes de impulsionar a exposição e a comercialização de serviços de uma organização (BUYYA, 2016). Ele também fornece ferramentas de

monitoramento e precificação de serviços mais eficientes do que as abordagens orientadas a serviços anteriores.

3.2 IoT e a Nuvem

A computação em nuvem, devido a seus recursos de processamento e armazenamento sob demanda, pode ser usada para analisar dados gerados por objetos IoT em formato de lote ou fluxo. Um modelo pay-as-you-go adotado por todos os provedores de nuvem reduziu o preço da computação, armazenamento de dados e análise de dados, criando um processo simplificado para a criação de aplicativos IoT. Com a elasticidade da nuvem, os Stream Processing Engines (SPEs) podem implementar recursos importantes, como tolerância a falhas e escalonamento automático para cargas de trabalho em rajadas.

Os aplicativos de IoT podem aproveitar os serviços de nuvem e usar os recursos de armazenamento e computação disponíveis para atender às suas demandas de processamento escalável e de computação intensiva, que são aplicações de computador que exigem muito poder de processamento (COMMISSION, 2015). A maioria das abordagens atuais de design para integrar nuvem com IoT é baseada em uma arquitetura de três camadas, onde a camada inferior consiste em dispositivos IoT, a camada intermediária é o provedor de nuvem e a camada superior hospeda aplicativos diferentes e protocolos de alto nível. No entanto, o uso dessa abordagem para projetar e integrar a computação em nuvem com um middleware de IoT limita a praticidade e a plena utilização da computação em nuvem em cenários em que minimizar o atraso de ponta a ponta é a meta. Por exemplo, no streaming de jogos online, onde o atraso é um fator importante para a satisfação do usuário, um middleware de IoT leve e sensível ao contexto que seleciona de maneira inteligente a rede de distribuição de conteúdo (CDN) mais próxima pode reduzir significativamente o atraso na entrega de dados geral (A ZASLAVSKY A, 2014).

3.3 Protocolos para IoT

Dado que as questões principais para o desenvolvimento de programas para a Internet das Coisas é a escolha de qual protocolo de aplicação será utilizado para comunicação entre as “coisas”. Protocolos de comunicação já definidos e utilizados por meio de computadores como por exemplo, o HTTP, não consideram limitações de processamento, rede e bateria do hardware sobre o qual executam. Essa limitação motivou a criação de novos protocolos regulados a este tipo de ambiente. Logo, esses protocolos miram no uso de um conjunto reduzido de mensagens e tamanho, de modo que a sobrecarga de dados seja pequena e, a capacidade computacional reduzido dos dispositivos não seja um impeditivo para a execução do protocolo (GONCALVES, 2016).

Os protocolos para Internet das Coisas se diferenciam em respeito a finalidade de uso e o escopo de operação. Do propósito de uso, pode-se perceber que alguns são indicados para aplicações com ênfase na coleta de dados enquanto outros tem maior uso em situações onde existe a necessidade do controle de dispositivos.

Quanto ao escopo de operação, tem-se protocolos que são mais recomendados para comunicação direta entre dispositivos (Dispositivo-para-Dispositivo), outros para comunicação entre os dispositivos e a Nuvem (Dispositivo-para-Nuvem) e outros ainda para interação entre usuários e dispositivos (Dispositivo-para-Humano). Para o desenvolvimento do protótipo proposto nesse trabalho, foi utilizado o procotolo MQTT (Message Queue Telemetry Transport), que é o protocolo usado pela AWS IoT Device SDK para Python que permite aos desenvolvedores escrever scripts em liguagem Python para acessar a plataforma AWS IoT.

3.3.0.1 MQTT (Message Queue Telemetry Transport)

O MQTT foi criado e desenvolvido pela IBM no final dos anos 90. Sua aplicação inicial era conectar sensores em pipelines de petróleo a satélites. Como seu nome insinua, ele é um protocolo de mensagem com suporte para a comunicação assíncrona entre as partes. No final do ano de 2014, ele tornou-se, de modo oficial, um padrão aberto OASIS, com suporte nas linguagens de programação populares.(YUAN, 2017).

O MQTT é um protocolo de rede leve e flexível que possibilita ao desenvolvedor um ambiente ideal para desenvolvimento de plataformas IoT:

- O protocolo simples permite a execução em hardware de dispositivo altamente restringido e, em redes de de alta latência e largura da banda limitada.
- Sua flexibilidade permite o suporte a vários ambientes de aplicativo para dispositivos e serviços de Internet das Coisas.

3.4 Modelos de Comunicação da IoT

De uma perspectiva operacional, é útil pensar sobre como os dispositivos IoT se conectam e se comunicam em termos de seus modelos de comunicação técnica. Em março de 2015, o Conselho de Arquitetura da Internet (IAB) divulgou um documento arquitetural orientador para redes de objetos inteligentes (RFC 7452), que delineia um quadro de quatro modelos comuns de comunicação usado por dispositivos IoT.

A seguir são conceituados os quatro modelos básicos de comunicação para objetos conectados (ROSE SCOTT ELDRIDGE, 2015):

- O modelo de comunicação de **dispositivo-para-dispositivo** representa dois ou mais dispositivos que se conectam diretamente e se comunicam entre si, em vez de usar um servidor intermediário de aplicativos. Esses dispositivos se comunicam por vários tipos de redes, incluindo redes IP ou a Internet. Muitas vezes, no entanto, esses dispositivos usam protocolos como Bluetooth, Z-Wave, ou ZigBee para estabelecer comunicações diretas entre dispositivos.

  Essas redes de dispositivo-para-dispositivo permitem que dispositivos que aderem a um protocolo de comunicação específico se comuniquem e troquem mensagens para alcançar sua função. Esse modelo de comunicação é comumente usado em aplicativos como sistemas de automação residencial, que

normalmente usam pequenos pacotes de dados de dados para se comunicar entre dispositivos com requisitos de taxa de dados relativamente baixos. Dispositivos residenciais de IoT, como lâmpadas, interruptores de luz, termostatos e fechaduras normalmente enviam pequenas quantidades de informações entre si (por exemplo, mensagem de status de bloqueio de porta ou comando de ativação de luz) em um cenário de automação residencial

- Em um modelo de comunicação de **dispositivo-para-nuvem**, o dispositivo IoT se conecta diretamente a um serviço de nuvem da Internet, como um provedor de serviços de aplicativo, para trocar dados e controlar o tráfego de mensagens. Essa abordagem frequentemente aproveita os mecanismos de comunicação existentes, como conexões com fio tradicionais Ethernet ou Wi-Fi, para estabelecer uma conexão entre o dispositivo e a rede IP, que finalmente se conecta ao serviço em nuvem.

  Esse modelo de comunicação é empregado por alguns dispositivos de IoT populares como o Nest Labs Learning Thermostat e o Samsung SmartTV. No caso do Nest Learning Thermostat, o dispositivo transmite dados para um banco de dados na nuvem onde os dados podem ser usados para analisar a energia doméstica consumo. Além disso, essa conexão na nuvem permite que o usuário obtenha acesso remoto ao termostato por meio de um smartphone ou interface da Web e também oferece suporte a atualizações de software para o termostato. Da mesma forma, com a tecnologia Samsung SmartTV, a televisão usa uma conexão de Internet para transmitir informações de visualização do usuário à Samsung para análise e para ativar os recursos de reconhecimento de voz interativos da TV. Nesses casos, o modelo de dispositivo para nuvem agrega valor ao usuário final estendendo os recursos do dispositivo além de seus recursos nativos.

- No modelo de **dispositivo-para-gateway**, ou mais tipicamente, no modelo de device-to-application-layer gateway (ALG), o dispositivo de IoT se conecta por meio de um serviço ALG como um canal para acessar um serviço de nuvem. Em termos mais simples, isso significa que existe software de aplicação operando em um dispositivo de gateway local, que atua como um intermediário entre o dispositivo e o serviço de nuvem e fornece segurança e outras funcionalidades, como tradução de dados ou protocolos.

  Várias formas desse modelo são encontradas em dispositivos de consumo. Em muitos casos, o dispositivo de gateway é um smartphone que executa um aplicativo para se comunicar com um outro dispositivo e retransmitir dados para um serviço de nuvem. Este é frequentemente o modelo empregado com itens de consumo populares, como rastreadores de fitness pessoais. Esses dispositivos não possuem capacidade nativa de se conectar diretamente a um serviço de nuvem, de modo que eles frequentemente confiam no software de aplicativo de smartphone para servir como um gateway intermediário para conectar o dispositivo de fitness à nuvem.

- O modelo de **compartilhamento de dados de backend** refere-se a uma arquitetura de comunicação que permite aos usuários exportar e analisar dados

de objetos inteligentes de um serviço de nuvem em combinação com dados de outras fontes. Essa abordagem é uma extensão do modelo de comunicação dispositivo para nuvem único, que pode levar a silos de dados em que dispositivos IoT carregam apenas dados para um único provedor de serviços. Uma arquitetura de compartilhamento de backend permite que os dados coletados de fluxos de dados de dispositivos de IoT únicos sejam agregados e analisados.

Para implementação da arquitetura proposta neste trabalho, utilizou-se as abordagens de comunicação **Dispositivo-para-Nuvem**, na qual os dados coletados pelo Raspberry Pi são enviados para o serviço de nuvem da Amazon. E também é utlizado o **Modelo de compartilhamento de dados backend**, em que os dados coletados pelo Raspberry Pi é enviado para a aplicação web.

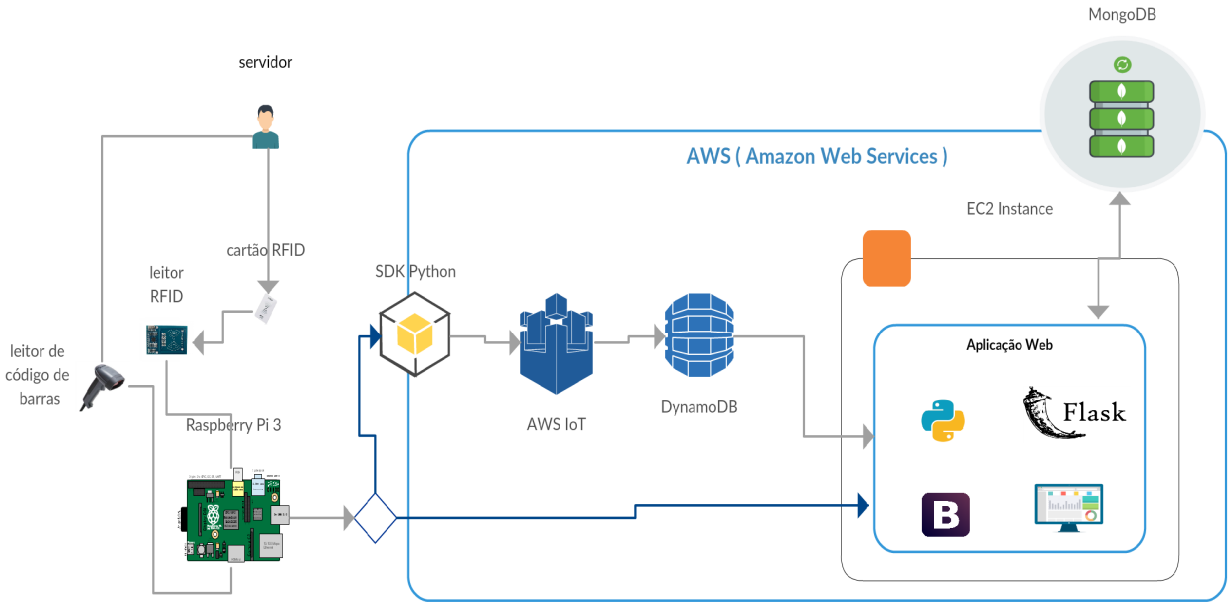
4 ARQUITETURA PROPOSTA

4.1 Introdução

Para o desenvolvimento do ambiente proposto, a construção de um protótipo que automatize o processo de empréstimo feito por um funcionário na Universidade Federal do Maranhão, foi modelado uma arquitetura em que houvesse a comunicação entre todas as partes do sistema, ou seja, o envio de dados coletado pelo módulo do Raspberry Pi para a nuvem da AWS, onde a aplicação se encontra hospedada, para que a mesma processe a informação, e facilite todo o processo de empréstimo. A figura 6 mostra a arquitetura utilizada na construção do protótipo.

É necessário que se possa entender a funcionalidade de cada parte da arquitetura, que se divide com uso da Internet das Coisas, através do Rasbberry Pi e da Computação em Nuvem, por meio dos serviços da Amazon.

Figura 6 – Arquitetura proposta para construção do protótipo.

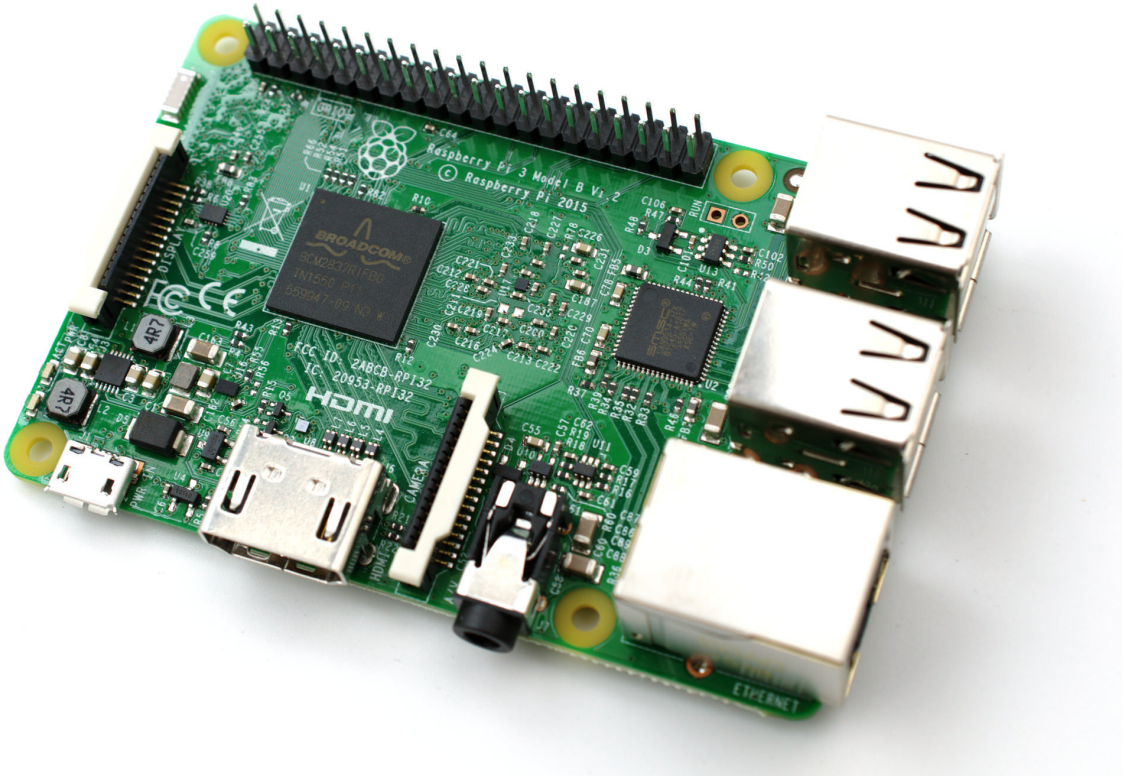


4.2 Raspberry Pi

4.2.1 Introdução

O Raspberry Pi 3, ilustrado na figura 7, é um computador de placa única criado pela Fundação Raspberry Pi, uma instituição de caridade formada com o objetivo principal de reintroduzir habilidades de informática de baixo nível para crianças no Reino Unido. O objetivo era reacender a revolução do microcomputador dos anos 80, que produziu toda uma geração de programadores qualificados (COX, 2014).

Figura 7 – Raspberry pi 3.



Lançado oficialmente em fevereiro de 2012, o computador pessoal Raspberry Pi conquistou o mundo, vendendo milhares de unidades disponíveis imediatamente. Trata-se de uma placa de circuito impresso de baixo custo e do tamanho de um cartão de crédito, um computador pessoal totalmente programável que executa o sistema operacional Linux de código-fonte aberto gratuito (COX, 2014). O Raspberry Pi possui um adaptador Wifi 802.11n integrado, além de um conector Ethernet para conexão de rede.

Originalmente criado para despertar o interesse das crianças em computadores, o Raspberry Pi chamou a atenção de curiosos por tecnologia, empreendedores e educadores em todo o mundo.

A linguagem de programação oficial do Raspberry Pi é Python. Python é uma linguagem de programação flexível que é executada em quase todas as plataformas (UDACITY, 2017). Assim, um programa pode ser criado em um computador com Windows ou Mac e executado no Raspberry Pi e vice-versa. O Python é uma linguagem de programação elegante, confiável, poderosa e muito popular, possuindo uma comunidade altamente ativa por todo o mundo.

4.2.2 Microcontrolador e Microcomputador

O Raspberry Pi trata-se de um microcomputador completo e não apenas um simples microcontrolador. Um microcomputador é um sistema autônomo que executa as seguintes tarefas de processamento de dados:

- **Entrada**: o computador recebe instruções e dados do usuário ou aplicativo.
- **Processamento**: O computador executa ações pré-programadas após sua entrada.
- **Saída**: o computador exibe os resultados do processamento de uma ou várias maneiras para o usuário ou aplicativo.

Por outro lado, um microcontrolador é uma peça de hardware muito mais especializada. Um microcontrolador é um circuito integrado que é semelhante a um computador pessoal, na medida em que recebe dados, executa o processamento nessa entrada e, finalmente, gera uma ou outra saída (BRAGA, 2018).

No entanto, o microcontrolador é separado do microcomputador pelas seguintes três características (WARNER, 2014):

- **A operação de um microcontrolador depende de um tempo preciso**: como o microcontrolador geralmente é um dispositivo de propósito único, não há sobrecarga de driver ou sistema operacional para desacelerar o sistema. Portanto, o microcontrolador pode realizar trabalhos usando ciclos de clock extremamente precisos. Esta operação dependente do tempo é difícil de realizar com o Raspberry Pi porque o Raspberry Pi deve acessar seu hardware através de várias camadas de software.
- **Um microcontrolador dá ao usuário acesso total e direto ao hardware**: a maior parte do hardware do Raspberry Pi (particularmente o sistema Broadcom BCM2835 em um chip) é proprietária. Por outro lado, a maioria dos microcontroladores como o chip RISC (Reduced Instruction Set Computing) são de código aberto e, portanto, são totalmente acessíveis aos usuários. Com o Raspberry Pi, estamos limitados a interagir com os componentes de hardware da placa por meio de interfaces de programação de aplicativos de software (APIs).
- **Um microcontrolador normalmente não tem interface de usuário**: um programador deve usar um sistema externo para enviar dados e receber dados de um microcontrolador.
- **Um microcontrolador é normalmente projetado para uma única finalidade**: um microcontrolador se destina a executar uma única tarefa e, fazer essa tarefa com precisão e muito bem. Por exemplo, considere uma estação meteorológica alimentada por Arduino que detecta o ambiente e informa sobre a temperatura do ar, umidade relativa, pressão barométrica e assim por diante.

4.2.3 Leitura de RFID com Raspberry Pi

Leitores e tags RFID (Radio Frequency Identification, ou Identificação por Radiofrequência) costumam ser usados para controle de acesso, reconhecimento de pessoas e equipamentos, seja por meio de crachás ou etiquetas aplicadas à produtos (VICENZI, 2016).

No Raspbberry Pi a comunicação com o leitor RFID RC522 é feita por meio da interface SPI (Serial Peripheral Interface). O SPI usa comunicação síncrona, full duplex, utilizando o modelo de comunicação mestre-escravo. Na figura 8 é possível observar quais são os pinos correspondentes ao SPI na GPIO do Raspberry Pi 3, eles estão destacados com a cor roxa.

Figura 8 – Pinos correspondentes ao SPI na GPIO do Raspberry Pi 3.

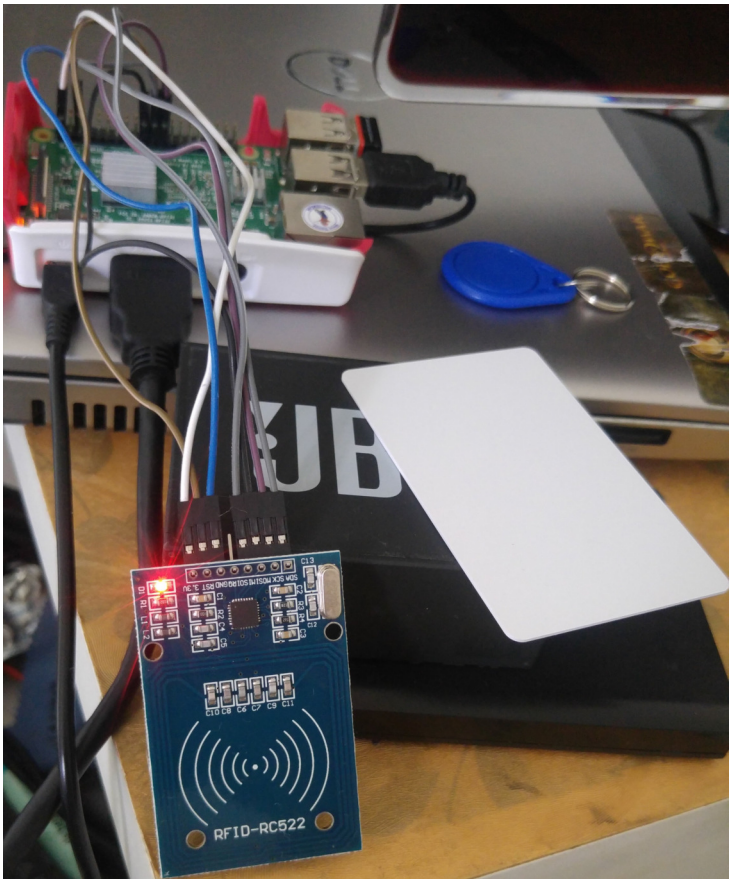
Raspberry Pi 3 GPIO Header

Pin#	NAME		NAME	Pin#
01	3.3v DC Power		DC Power 5v	02
03	GPIO02 (SDA1 , I²C)		DC Power 5v	04
05	GPIO03 (SCL1 , I²C)		Ground	06
07	GPIO04 (GPIO_GCLK)		(TXD0) GPIO14	08
09	Ground		(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)		(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)		Ground	14
15	GPIO22 (GPIO_GEN3)		(GPIO_GEN4) GPIO23	16
17	3.3v DC Power		(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)		Ground	20
21	GPIO09 (SPI_MISO)		(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)		(SPI_CE0_N) GPIO08	24
25	Ground		(SPI_CE1_N) GPIO07	26
27	ID_SD (I²C ID EEPROM)		(I²C ID EEPROM) ID_SC	28
29	GPIO05		Ground	30
31	GPIO06		GPIO12	32
33	GPIO13		Ground	34
35	GPIO19		GPIO16	36
37	GPIO26		GPIO20	38
39	Ground		GPIO21	40

O Raspbian, distribuição linux que pode ser usada no Raspberry Pi, vem com SPI desabilitado por padrão. Para ativar o SPI no Raspbian, é necessário acessar o Menu > Preferences > Raspberry Pi Configuration e na aba Interface, habilitar o SPI. Após habilitado, é preciso a instalação do pacote Python para comunicação SPI, chamado **MFRC522.py.**

A figura 9 mostra o módulo RFID conectado ao Raspberry Pi.

Figura 9 – Módulo RFID conectado ao Raspberry Pi.



4.2.4 Leitura de Código de barras com Raspberry Pi

Código de barras é uma representação gráfica de dados numéricos ou alfanuméricos. A leitura dos dados é feita por um scanner - o leitor de código de barras.

O leitor de código de barras pode ser conectado ao Raspberry Pi através de uma das suas entradas USB. Uma vez conectado, o driver do leitor é instalado automaticamente, e o leitor já estará disponível para leitura.

4.2.5 Utilizando a leitura de RFID e de código de barras

Para poder enviar o ID do cartão lido através do módulo RFID RC522 foi necessário a implementação de um código em Python. Este, tem tem como finalidade, possibilitar ao funcionário que utilizará o Raspberry Pi, a leitura do RFID presente no cartão de um professor, de um funcionário ou apenas a leitura do ID. Cada finalidade é detalhada abaixo:

- **RFID presente no cartão de um professor**: Ao ler o ID presente no cartão do professor, o funcionário da UFMA deverá escolher a opção de enviar "código RFID para a nuvem da Amazon", através de um tópico que será lido pelo serviço AWS IoT.

- **RFID presente no cartão de um funcionário da UFMA**: O funcionário da UFMA deverá escolher a opção de "Autenticar Servidor ", no código Python, no qual tem como objetivo fazer uma requisição para um método da aplicação web, em que esta possibilitará ao funcionário fazer autenticação sem a necessidade de digitar uma senha.
- **leitura do ID**: Com a opção "visualizar ID", o programa mostrará no console o ID presente no cartão.

Vale lembrar que a identificação das coisas no AWS IoT pode ser realizada de diferentes maneiras. Neste protótipo é utilizado um certificado digital, que é uma das formas mais comuns e seguras para a identificação de dispositivos. No apêndice A, é mostrado o código de exemplo utilizado para leitura de RFID:

Para que seja capturado a identificação dos equipamentos, é utilizado um leitor de código de barras da marca Datalogic modelo Touch 65 que é conectado por uma porta USB. Basta o funcionário aproximar o leitor do código de barras do equipamento, que o ID será mostrado em uma entrada padrão, como por exemplo, uma entrada de formulário HTML.

4.3 Arquitetura em nuvem AWS

4.3.1 Introdução

A Amazon Web Services (AWS) é uma plataforma de serviços em nuvem segura, ofertando poder computacional, armazenamento de banco de dados, distribuição de conteúdo e outras funcionalidades para auxiliar as empresas em seu dimensionamento e crescimento.

No momento atual, a Amazon Web Services oferece na nuvem uma plataforma de infraestrutura confiável, escalável e de baixo custo. Com data centers localizados nos EUA, Europa, Brasil, Cingapura, Japão e Austrália, clientes de todos os setores estão tendo proveitos com os seguintes benefícios (AMAZON, 2018):

- **Baixo custo**: A AWS oferece preços baixos e do modelo pague o que usar, sem despesas iniciais nem compromissos a longo prazo.
- **Agilidade e elasticidade instantânea**: A AWS fornece uma ampla infraestrutura de nuvem global que permite que o usuário faça mudanças, experimente e interaja rapidamente. O usuário pode implantar novos aplicativos, expandir à medida que sua carga de trabalho aumenta e reduzir com base na demanda.
- **Aberto e flexível**: A AWS é uma plataforma e um sistema operacional de linguagem independente. O usuário escolhe a plataforma de desenvolvimento ou o modelo de programação que faz mais sentido para o seu negócio. Pode escolher os serviços que deseja usar, um ou vários, e escolher como usá-los.

- **Seguro**: A AWS é uma plataforma de tecnologia segura, durável com auditorias e certificações reconhecidas pelo setor: PCI DSS nível 1, ISO 27001, FISMA Moderado, HIPAA e SSAE 16.

4.3.2 Mongo Atlas

MongoDB Atlas é um banco de dados como um serviço criado pelos especialistas que projetam e desenvolvem o MongoDB. Ele fornece todos os recursos do MongoDB, enquanto remove a maior parte da sobrecarga operacional (DOMINGUES, 2018).

O Atlas suporta a implementação de clusters no Amazon Web Services (AWS), então para o desenvolvimento desta arquitetura foi escolhida instâncias fornecidas pela Amazon, no caso, o nível gratuito, que possui até 512 MB de armazenamento.

4.3.3 Serviços da AWS utilizados

4.3.3.1 AWS IoT

A AWS IoT possibilita comunicação bidirecional segura entre dispositivos conectados à Internet como sensores, acionadores incorporados ou aparelhos inteligentes e a nuvem AWS. Isso permite que se colete dados de telemetria de vários dispositivos, armazene e analise os dados. Possibilita também criar aplicativos que permitem que os usuários controlem esses dispositivos de smartphones ou tablets.

O IoT possui o Device Gateway que funciona no modelo ”publish/subscribe”. Este modelo possibilita que dispositivos realizem a publicação de mensagens em tópicos (publish) e que outros dispositivos ou aplicações recebam as mensagens publicadas em tópicos específicos (subscribe) (PIMENTEL, 2016).

O AWS IoT fornece um mecanismo seguro para dispositivos e aplicativos da AWS IoT a fim de publicar e receber mensagens um do outro. Pode-se usar diretamente o protocolo MQTT ou o MQTT pelo WebSocket para publicar e receber.

4.3.3.2 EC2 (Elastic Compute Cloud)

Amazon Elastic Compute Cloud (Amazon EC2) é um serviço web que fornece capacidadede computação redimensionável na nuvem. Possibilita que as empresas possam obter e configurar servidores virtuais nos data centers da Amazon e aproveitar esses recursos para criar e hospedar sistemas de software. As organizações podem escolher uma variedade de sistemas operacionais e recursos de configuração (memória, CPU, armazenamento, etc.) que sejam os ideais de acordo com o perfil da aplicação e da carga de trabalho.

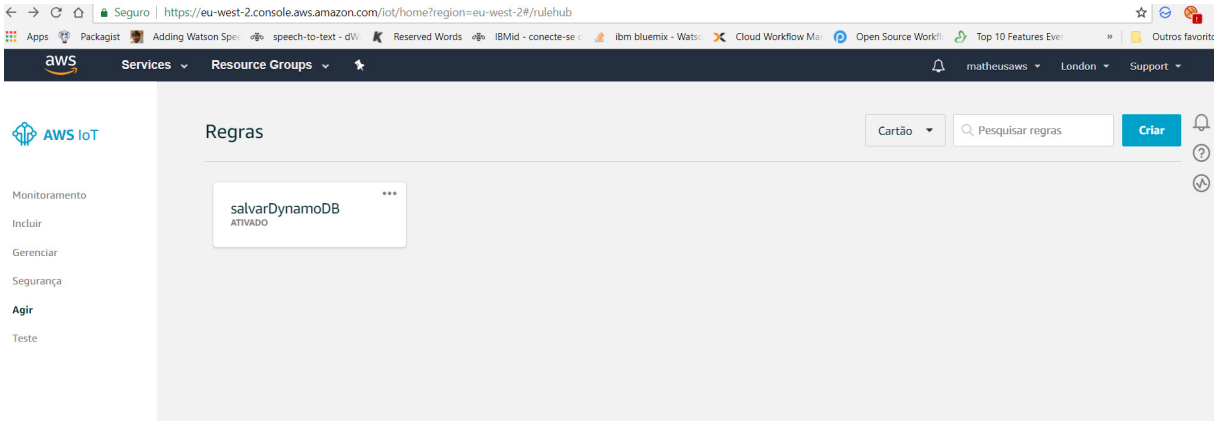
Amazon EC2 apresenta um verdadeiro ambiente de computação virtual, permitindo que as empresas utilizem vários sistemas operacionais, implementem seus aplicativos personalizados e gerenciem permissões de acesso à rede enquanto mantém o controle completo sobre o ambiente.

4.3.3.3 Utilizando o DynamoDB na arquitetura

O Amazon DynamoDB é um serviço de banco de dados não relacional ágil e ajustável para todas as aplicações que necessitem de uma latência de milissegundos seja qual for a escala. É um banco de dados inteiramente gerenciado e suporta modelos de dados de documento e chave/valor. Seu modelo de dados flexível e seu desempenho confiável o tornam ótimo para aplicações mobile, web, jogos, tecnologia de publicidade, Internet das Coisas e muitas outras aplicações.

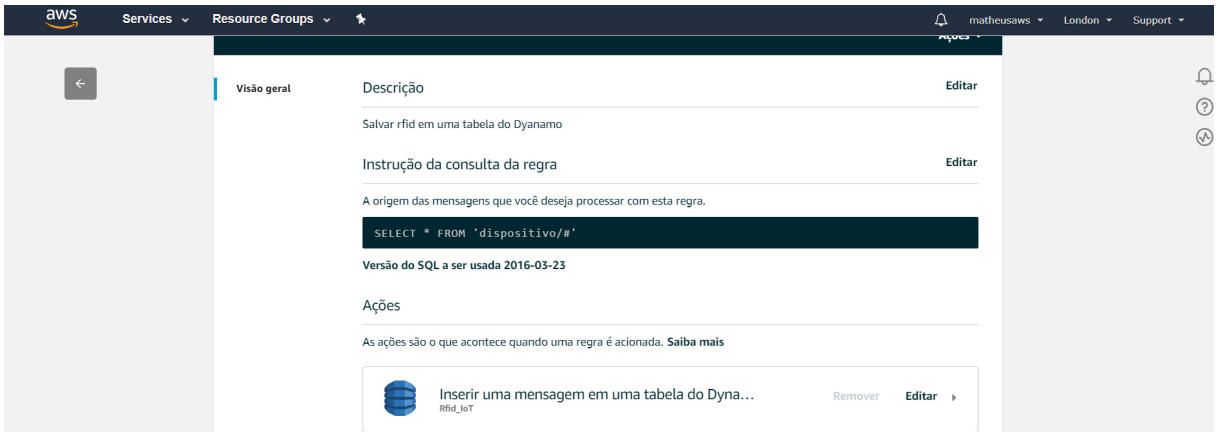
Para armazenar as mensagens enviadas do leitor de RFID para um tópico do AWS IoT em um repositório, de modo que a aplicação web possa acessar esses dados, foi criada uma regra, como visto na figura 10 .

Figura 10 – AWS IoT Regra.



Uma regra para serve para avaliar mensagens enviadas por suas coisas e determinar o que fazer quando uma mensagem é recebida. Neste caso, foi criado uma regra para para armazenar as mensagens recebidas no Amazon DynamoDB, ilustrado na figura 11.

Figura 11 – Regra IoT AWS utilizando o DynamoDB.



4.3.4 Utilizando a AWS na construção do protótipo

Para ter a acesso aos dados coletados pelo Raspberry Pi foi necessário a criação de uma arquitetura em Nuvem. Com uso do serviço AWS IoT, é necessário primeiro registrar seu dispositivo ou coisa, para depois poder fazer a conexão, como ilustrado na figura 12. É publicado no tópico ”dispositivo/raspberry/reportar” os dados do cartão RFID. Após esta etapa, outro serviço da Amazon, o DynamoDB, irá armazenar esse ID em uma tabela de RFID, ilustrado na figura 13. Assim, o funcionário vai conseguir capturar o ID do professor na aplicação web desenvolvida.

Figura 12 – Registro do RaspberryP Pi.

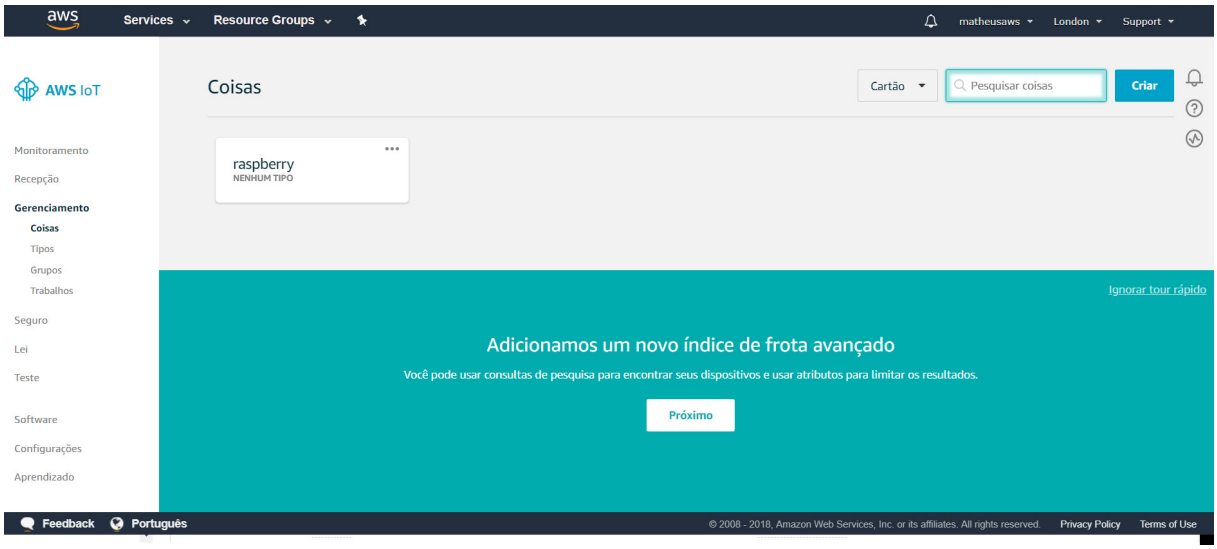
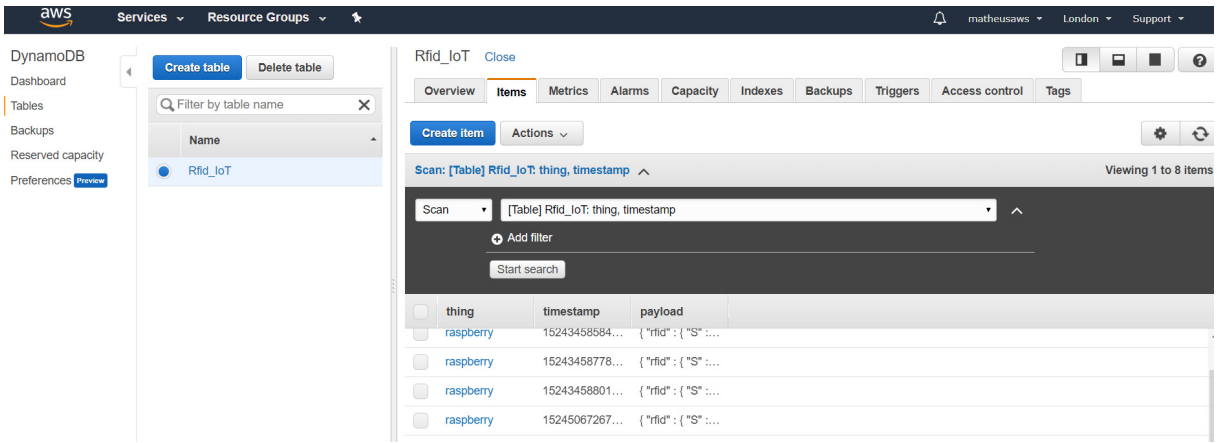


Figura 13 – Tabela do Dynamo



Outra funcionalidade dos serviços da AWS usado para construção do protótipo, é o uso de instâncias EC2 para implantação da aplicação web e banco de dados. Para se acessar as instâncias EC2, é necessário se estabelecer uma conexão SSH. Neste caso, utiliza-se o Putty, que é um cliente SSH muito utilizando em ambiente Windows, em que é necessário informar o nome do host da sua instância EC2 e

carregar a chave de acesso que é gerada no momento de criação de uma instância EC2.

Para poder estabelecer conexão, basta clicar no botão **Open**, como a figura 14 mostra. Após a conexão, é aberto um prompt, visto na figura 15, que vai pedir ao usuário informar o login. O nome do login depende do tipo de instância EC2 selecionada para criar, e para saber qual nome colocar, é preciso acessar a documentação EC2 no próprio site da AWS.

Figura 14 – Conexão com o Putty.

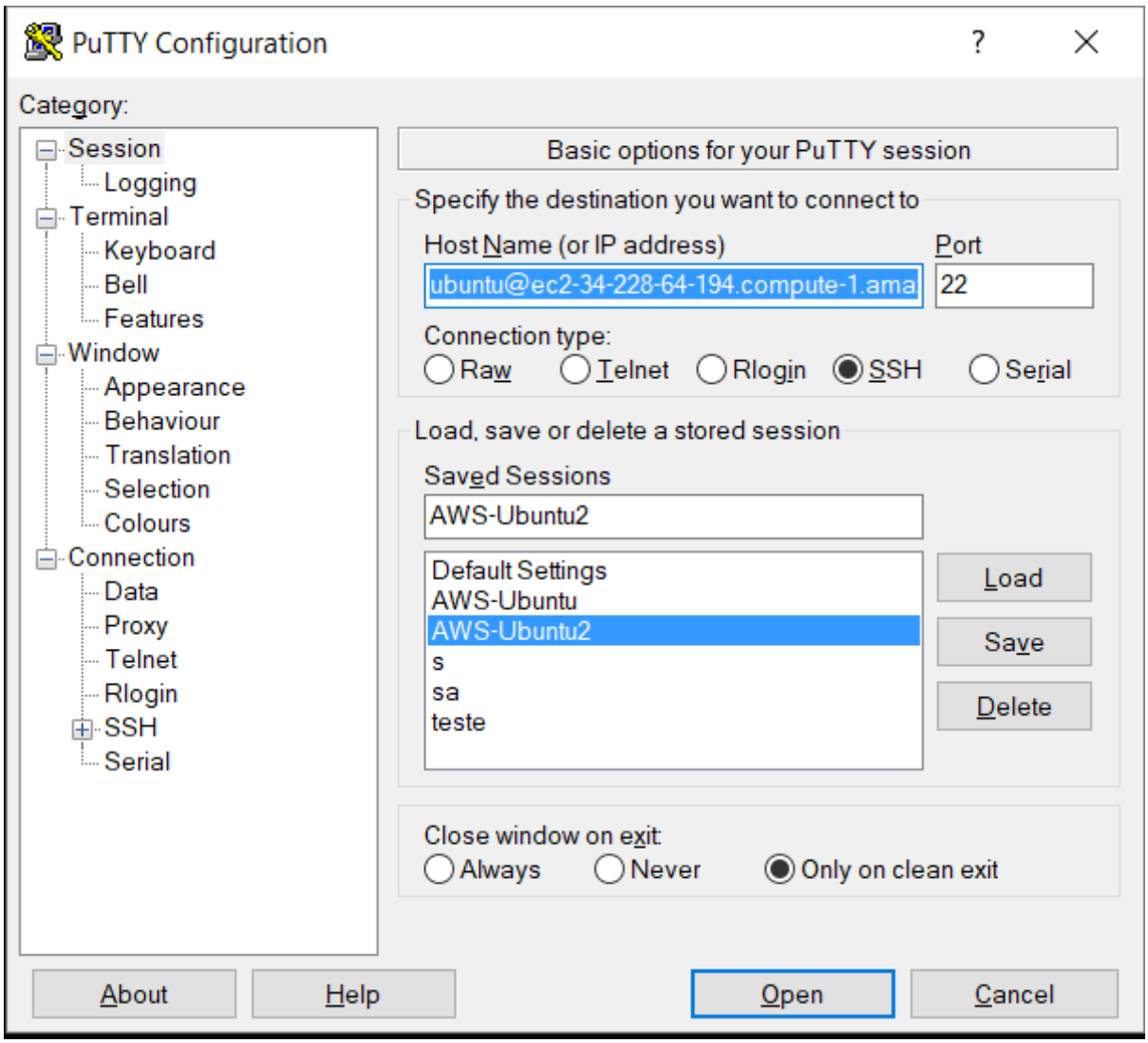
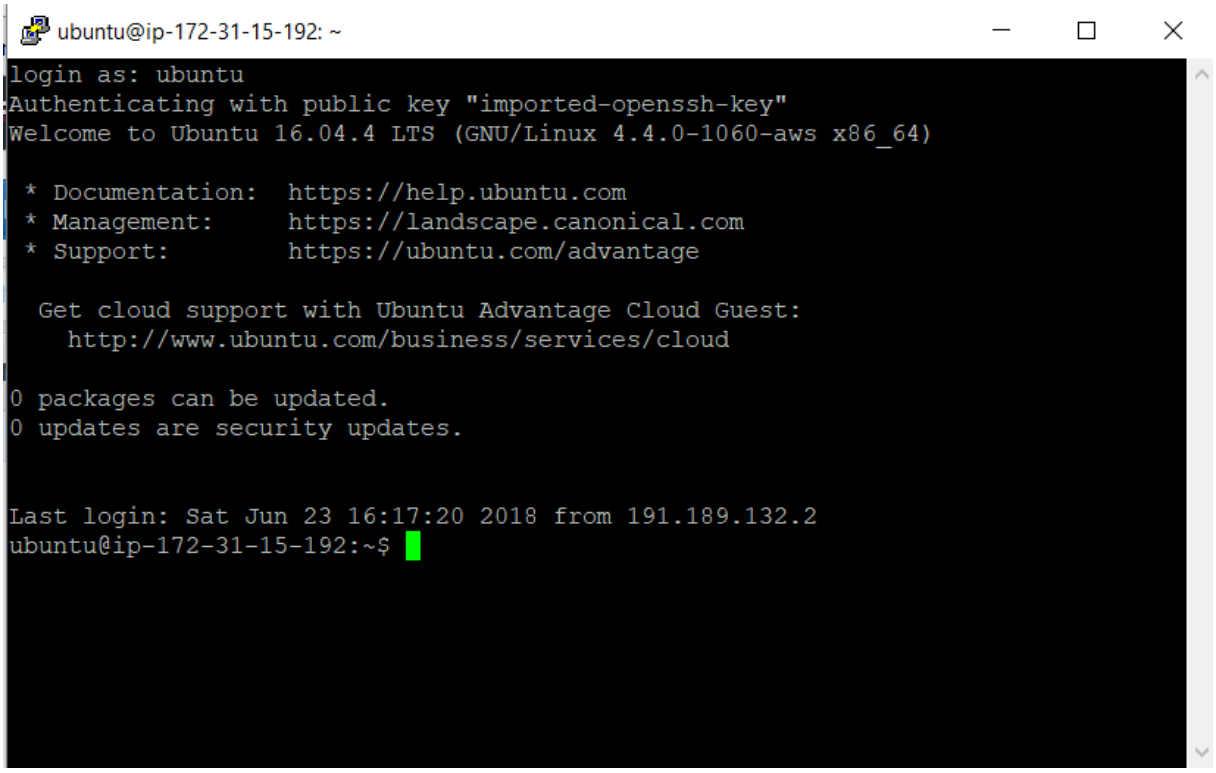
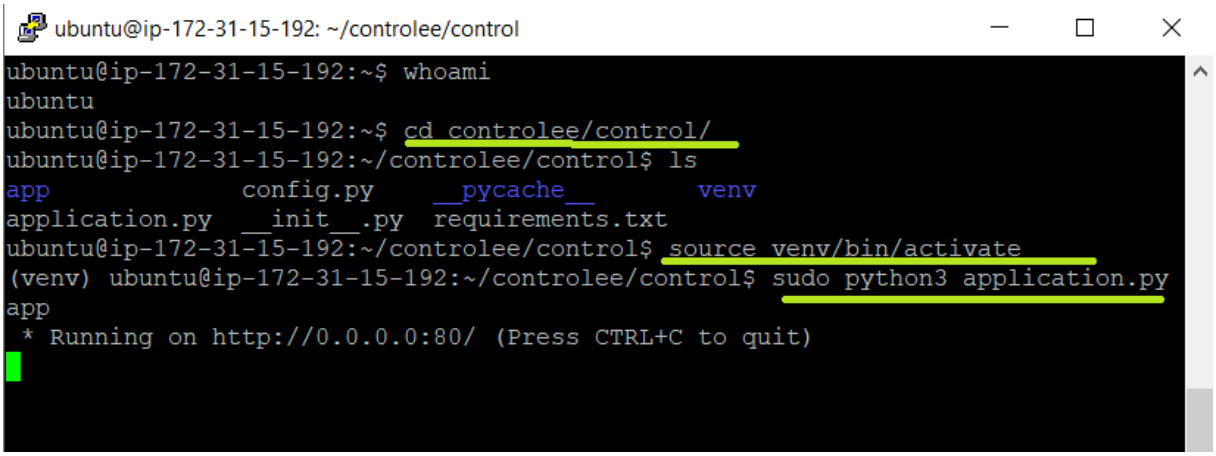


Figura 15 – Login na instância EC2.



Finalmente, é necessário reproduzir o ambiente da aplicação web na maquina local para instância criada. Para isso, basta instalar o Python, instalar um ambiente virtual e transferir a implementação da aplicação web para a instância criada. Feito isso, basta acessar o diretório em que a implementação foi armazenada e ativar o ambiente virtual, assim, a aplicação web estará pronta para executar como ilustra a figura 16.

Figura 16 – Passos para execução da aplicação web na instância EC2.

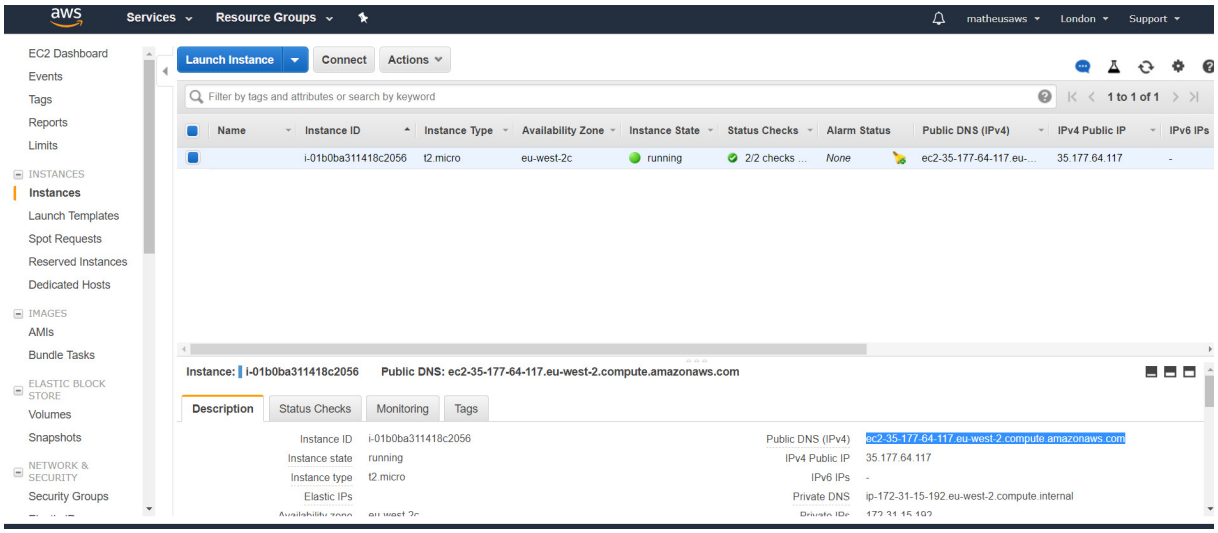


Para acessar a aplicação web, pode-se utilizar o DNS público da instância, que é um endereço normal de internet que a Amazon AWS fornece, como a figura 17

ilustra.

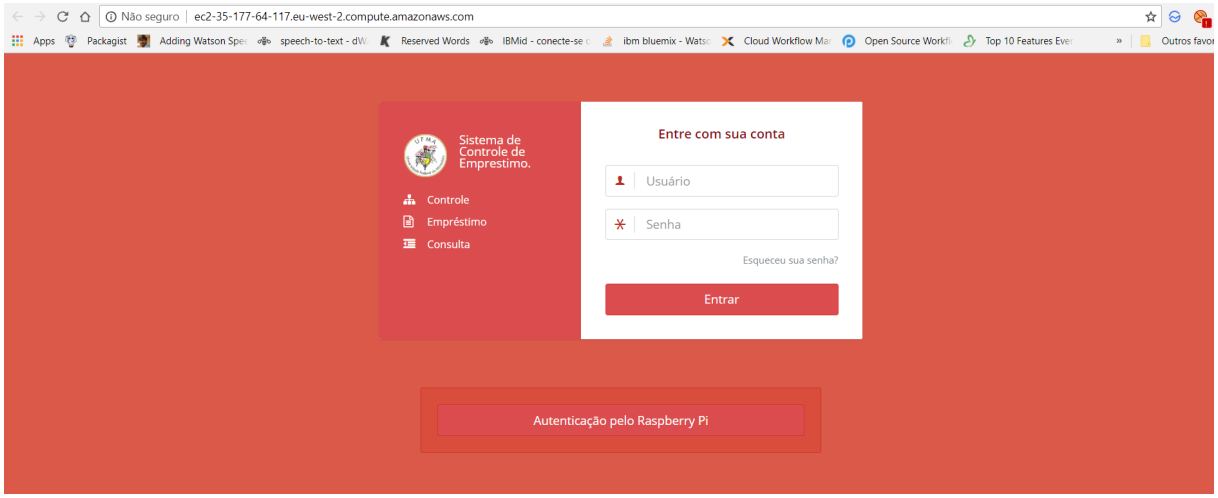
O DNS público é um nome pelo qual o computador (instância) pode ser acessado na Internet. É uma forma simples de obter o endereço do servidor e disponibilizar um site na internet (R.LECHETA, 2014).

Figura 17 – DNS público da instância EC2.



Basta copiar o DNS público e acessar a página da aplicação web pelo browser, conforme a figura 18 mostra.

Figura 18 – Acessando aplicação pelo DNS público da instância EC2.



5 ESTUDO DE CASO: APLICAÇÃO WEB

5.1 Introdução

Para facilitar a iteração do funcionário da UFMA com o protótipo, foi desenvolvido uma aplicação web que possui as principais funcionalidades do sistema. A aplicação foi desenvolvida utilizando o Flask, que é um pequeno framework web escrito em Python e baseado na biblioteca WSGI Werkzeug e na biblioteca de Jinja2 (ROCHA, 2014). Para persistência de dados, foi utilizado o banco de dados não relacional, o mongoDB, que é um software de banco de dados orientado a documentos livre, de código aberto e multiplataforma. Outra tecnologia utilizada foi o estilo de arquitetura REST, que define um conjunto de restrições e propriedades baseados em HTTP.

5.2 Funções da aplicação web

A aplicação web ”Sistema de Controle de Empréstimo” oferece serviços para o controle de equipamentos para fins de empréstimo da Universidade Federal do Maranhão.

É listado abaixo as funções presentes na aplicação web desenvolvida:

a) **Gerenciamento de Usuário:** Para o funcionário que deseja utilizar algum dos serviços do sistema, precisa-se primeiro fazer seu registro, informando nome de usuário, e-mail, uma senha para acesso ao sistema, e o RFID do seu cartão de autenticação, que também serve para acesso ao sistema. Depois do cadastro, o funcionário poderá acessar o ambiente do sistema.

b) **Autenticação:** Para acessar ao ambiente do sistema web, o funcionário terá opção de autenticar pelo nome de usuário e senha, ou pelo RFID. No caso do RFID, ele deverá usar um módulo leitor de RFID que estará conectado a um Raspberry Pi. Este, enviará através de um requisição o ID do funcionário para o sistema web. Com isso o funcionário precisará apenas informar seu nome de usuário e fazer a autenticação.

c) **Permissões:** O sistema web mostra ao funcionário todos os módulos em que abrange, que são:

- Empréstimo: Responsável por todas as operações referente a ação de realizar um empréstimo por parte de um professor, funcionário ou aluno.
- Equipamentos: Responsável por todas as operações referente a ação de cadastrar, excluir ou alterar as informações de um equipamento.
- Estatística: Mostra alguns valores referentes a empréstimos e equipamentos.
- Servidores: Incubido pelo cadastro, consulta e alteração de um funcionário.

- Salas: Incubido pelo cadastro e consulta de salas.
- Professores: Encarregado pelo cadastro de um professor.

Existe apenas um tipo de usuário do sistema que será o funcionário. Todos os funcionários terão o mesmo nível de permissão. Estes usuário são responsáveis por fazer o cadastro de novos equipamentos, novos empréstimo, professores, salas e novos funcionários. Funcionários não poderão alterar os dados cadastrais de outros Funcionários.

d) **Consultas:** O funcionário poderá consultar empréstimos realizados, equipamentos, salas, professores e funcionários.

e) **Cadastro de Equipamentos:** Para cadastrar um novo equipamento, o funcionário precisará informar o nome do equipamento, tipo do equipamento, situação, descrição e o número de identificação, que será pego através de um leitor de código de barras.

f) **Cadastro de Salas:** Para cadastrar novas salas, o funcionário poderá escolher a opção de carregar um arquivo JSON com as informações cadastrais da sala. Ou fazer o cadastro manualmente, informando código, disciplina, departamento, professor, horários, sala, vagas, curso da disciplina e semestre.

g) **Cadastro de Empréstimos:** Com equipamentos cadastrados, o funcionário estará possibilitado de realizar um empréstimo, no qual deverá informar o nome do funcionário responsável pelo empréstimo, usuário, ou seja, o nome de quem solicita o empréstimo, número de usuário, que é o RFID contido no cartão, tipo de usuário, que pode ser aluno ou professor, horários, data inicial e data final. Caso o numero do usuário seja informado o sistema irá procurar na base de dados informações sobre esse usuário e preencher automaticamente o nome do usuário, horários, data inicial e data final. Se o usuário for um professor com dados cadastrados no módulo de sala, informações como horários, data inicial e data final serão preenchidas automaticamente.

h) **Empréstimos:** Após a realização do cadastro de um empréstimo o sistema mostrar na tela de consulta a situação do empréstimo, que podem ser:

- Concluído.
- Em aberto.
- Atrasado.

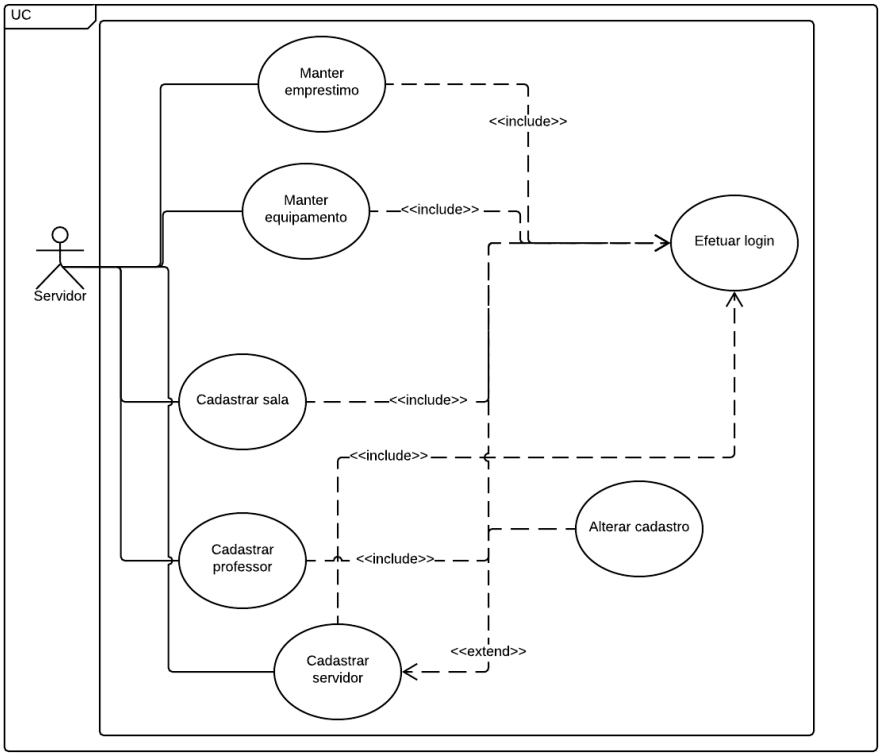
i) **Gráficos e Estatísticas:** A aplicação mostrará algumas estatísticas referentes aos empréstimos e equipamentos.

j) **Cadastro de Professores:** A aplicação web permitirá ao funcionário cadastrar novos professores, na qual se deve informar apenas o nome e o número presente no cartão de identificação que pode ser lido pelo módulo leitor RFID conectado ao Raspberry Pi.

5.3 Modelagem da Aplicação web

Foi utilizado a UML para modelagem. A Linguagem de Modelagem Unificada (UML) foi criada para constituir uma linguagem de modelagem visual comum, semanticamente e sintaticamente rica, para design, arquitetura, e implementação de sistemas de software, tanto estruturalmente quanto para comportamentos (INC., 2018). A figura 19 mostra o diagrama de caso de uso das principais funcionalidades desenvolvidas no protótipo da aplicação web .

Figura 19 – Diagrama de caso de uso da aplicação web.



Abaixo serão listados os casos de uso do sistema, alem de uma breve descrição dos mesmos:

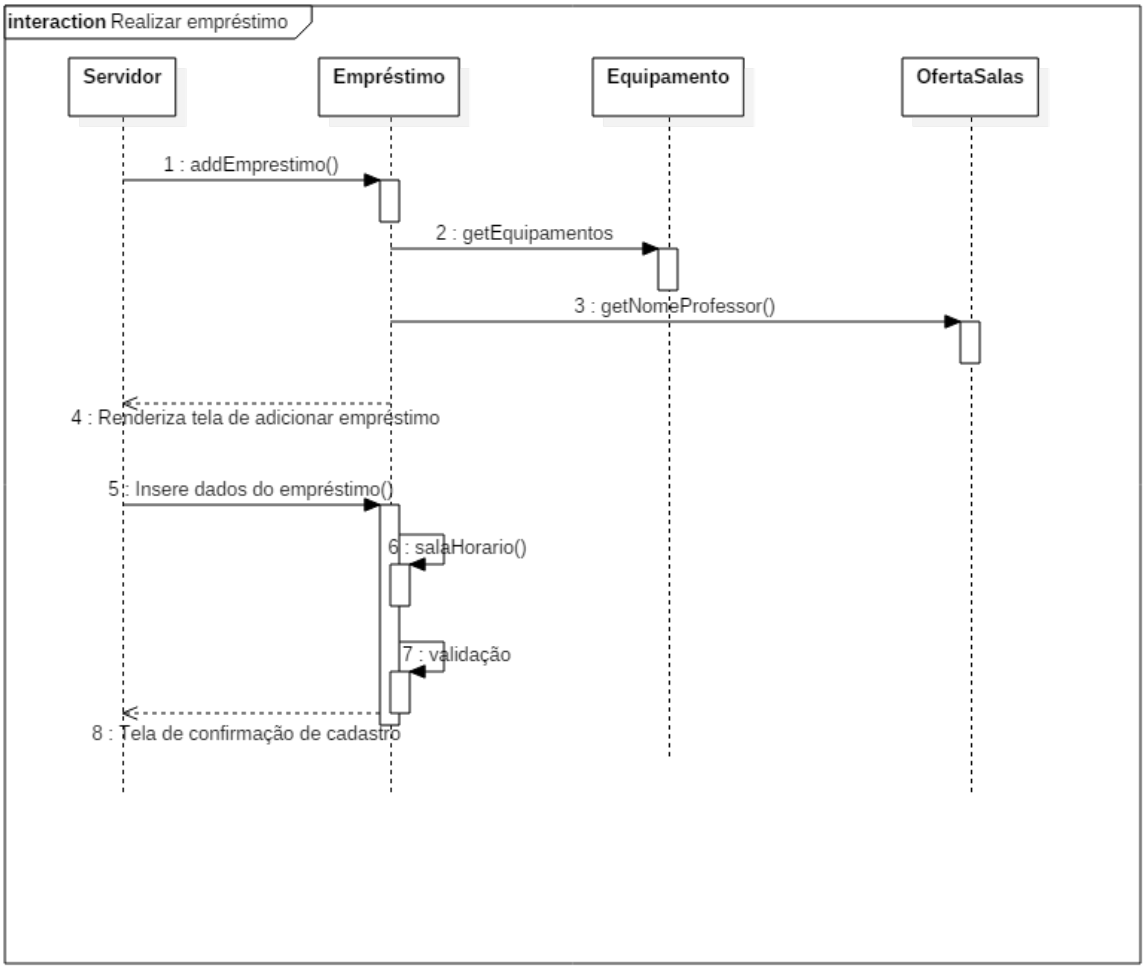
- **UC01 - Manter empréstimo**: Este caso de uso permite ao funcionário inserir um novo empréstimo no sistema, excluir um empréstimo da base de dados do sistema, e fazer uma consulta referente a um empréstimo.
- **UC02 - Manter equipamento**: Este caso de uso permite ao funcionário inserir um novo equipamento no sistema, alterar os dados cadastrais do equipamento, excluir um equipamento da base de dados do sistema, e fazer uma consulta referente a um equipamento.
- **UC03 -Cadastrar sala**: Este caso de uso permite ao funcionário cadastrar informações referente a uma nova sala. Ou seja, informando os professores

e disciplinas que aquela sala terá, assim como os horários das respectivas disciplinas.

- **UC04 -Cadastrar sala**: Este caso de uso permite ao funcionário cadastrar informações referente a uma nova sala. Ou seja, informando os professores e disciplinas que aquela sala terá, assim como os horários das respectivas disciplinas.
- **UC05 -Cadastrar professor**: Permite o cadastro de um professor da UFMA.
- **UC06 -Cadastrar servidor**: Permite o cadastro de um funcionário da UFMA.
- **UC07 -Efetuar Login**: Autenticação do funcionário para acessar o ambiente do sistema web.
- **UC08 -Alterar cadastro**: Alterar os dados cadastrais de um professor.

Após o diagrama de caso de uso, foi modelado um diagrama de sequência, que procuram determinar a sequência de eventos que ocorrem em um determinado processo (SOUZA, 2013).

Figura 20 – Diagrama de sequência para cadastro de empréstimo.



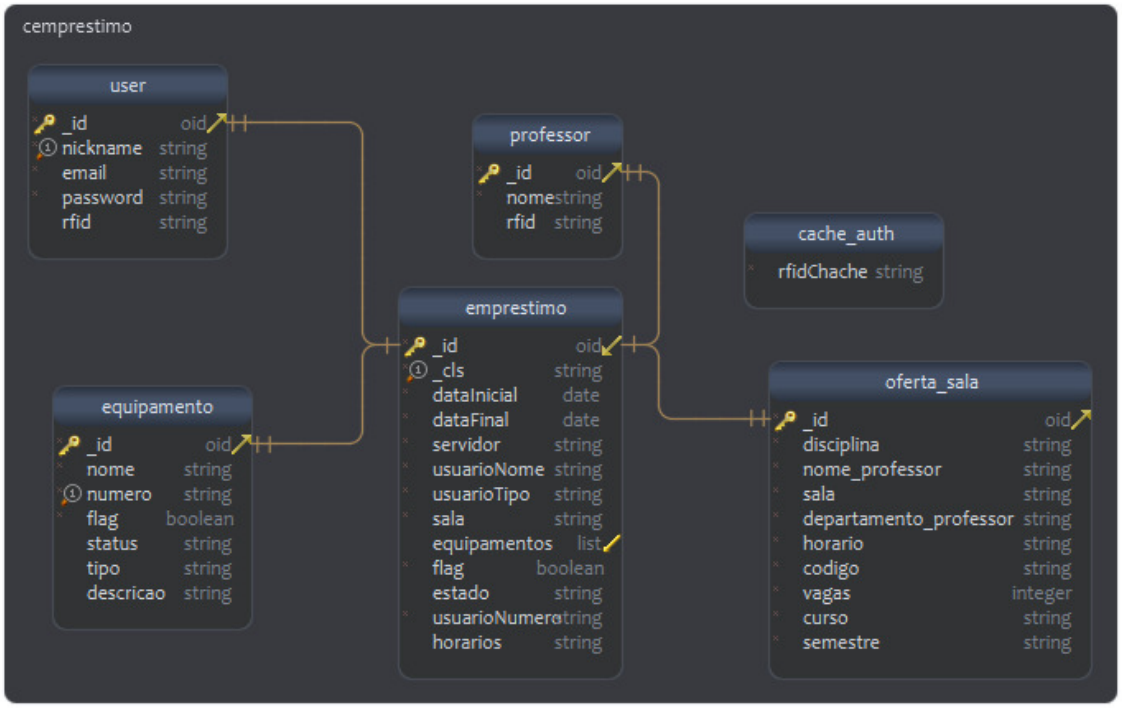
No diagrama, mostra a sequência de eventos para se adicionar um novo empréstimo. O funcionário deve primeiramente chamar o método para adicionar empréstimo. Este, carrega o nome dos equipamentos, presentes na coleção de ”Equipamentos” e, o nome dos professores, presente na coleção ”OfertaSala” nos devidos campos do formulário de cadastro de empréstimo. Com isso, o formulário de cadastro é renderizado para o usuário, e este preenche os campos com os dados corretos. Após, é chamado o método ”salaHorario” para preenchimento automático dos campos do formulário e é feita a validação dados fornecidos pelos usuário. Por fim, é retornado uma tela de confirmação de cadastro.

5.3.1 Modelagem do banco de dados

O banco de dados utilizado na construção da aplicação web foi o Mongodb. O MongoDB é um banco de dados de código fonte aberto, gratuito, de alto desempenho, sem esquemas e orientado a documentos lançado em fevereiro de 2009 pela empresa 10gen (DUARTE, 2017). MongoDB utiliza documentos semelhantes a JSON com esquemas.

Foi criado um esquema chamado ”cemprestimo”, na qual os coleções são guardadas. Foram criadas um total de sete coleções como mostra a modelagem na figura 21.

Figura 21 – Modelagem do banco de dados.



Oposto das tecnologias relacionais que têm uma visão horizontal da tabela, pode-se dizer que a visão do banco de dados orientado a documentos é vertical (PEREIRA, 2017), como pode ser visto nas listagens abaixo:

Código 5.1 – COLECAO EMPRESTIMO

```
{
        "_id": {
                "$oid": "5aca915a3a05e64f00290703"
        },
        "_cls": "Emprestimo",
        "dataInicial": {
                "$date": "2018-04-08T19:01:30.000Z"
        },
        "dataFinal": {
                "$date": "2018-04-08T19:03:00.000Z"
        },
        "servidor": "gestor",
        "usuarioNome": "CARLOS DE SALLES SOARES NETO",
        "horarios": "1",
        "usuarioNumero": "12133131",
        "usuarioTipo": "professor",
        "sala": "sala 106n bloco P, sala 205s bloco P",
        "equipamentos": [
                "1121212121212131"
        ],
        "flag": false,
        "estado": "atraso"
}
```

Código 5.2 – COLECAO EQUIPAMENTO

```
{
        "_id": "5aa827953a05e62aa8479309",
        "nome": "Monitor Samsung",
        "numero": "1121212121212131",
        "flag": true,
        "status": "pronto",
        "tipo": "monitor",
        "descricao": "monitor"
}
```

Código 5.3 – COLECAO PROFESSOR

```
{
        "_id": "5ad9259e3a05e62fcc003043",
        "nome": "WELLINGTON CONCEICAO CARVALHO",
        "rfid": "112345"
```

```
}
```

Código 5.4 – COLECAO OFERTA_SALA

```
{
        "_id": "5abcd5e93a05e603d0f36990",
        "disciplina": "CALCULO DIFERENCIAL E INTEGRAL I  (CP)",
        "nome_professor": "WELLINGTON CONCEICAO CARVALHO",
        "sala": "sala 106n bloco P",
        "departamento_professor": "DEMAT/CCET",
        "horario": "6T12",
        "codigo": "DEMA0164",
        "vagas": 60,
        "curso": "CIENCIA DA COMPUTACAO",
        "semestre": "2018.1"
}
```

Código 5.5 – COLECAO USER

```
{
        "_id": "5ade256a3a05e65fccca9e76",
        "nickname": "gestor",
        "email": "email@mail.com",
        "password": "$2b$12$brzccWkJsbfiPyNfwGfR8euygFSwpL8uASxnT4
           RVqfs1d9NwZFDkO",
        "rfid": "577D34839D"
}
```

Código 5.6 – COLECAO CACH_AUTH

```
{
        "_id": "5b115c259cca164e7c7f7c2b",
        "rfidChache": "577D34839D"
}
```

5.4 Interfaces externas

Esta sessão especifica algumas interfaces de usuário. As páginas são mostradas abaixo.

A tela de login da aplicação web é ilustrada na figura 22. Observa-se que existe o modo de autenticação convencional, ou seja, informando nome de usuário e senha para autenticação e, a autenticação pelo Raspberry Pi, na qual é preciso informar o nome de usuário apenas, como ilustrado na figura 23 .

Figura 22 – Tela de Login da aplicação web.

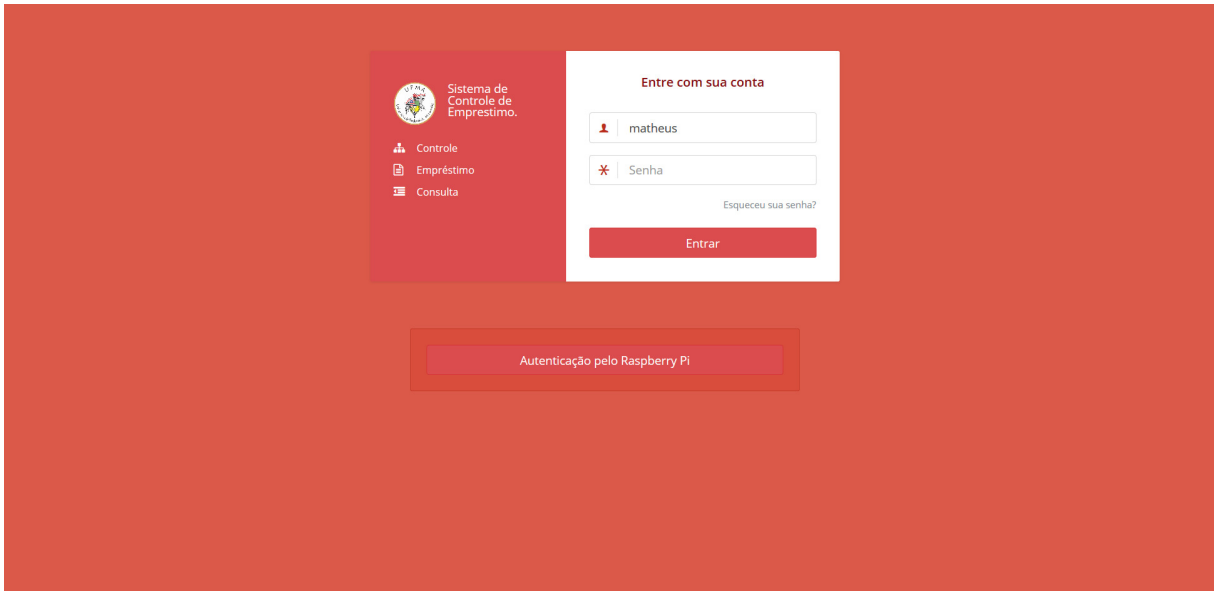
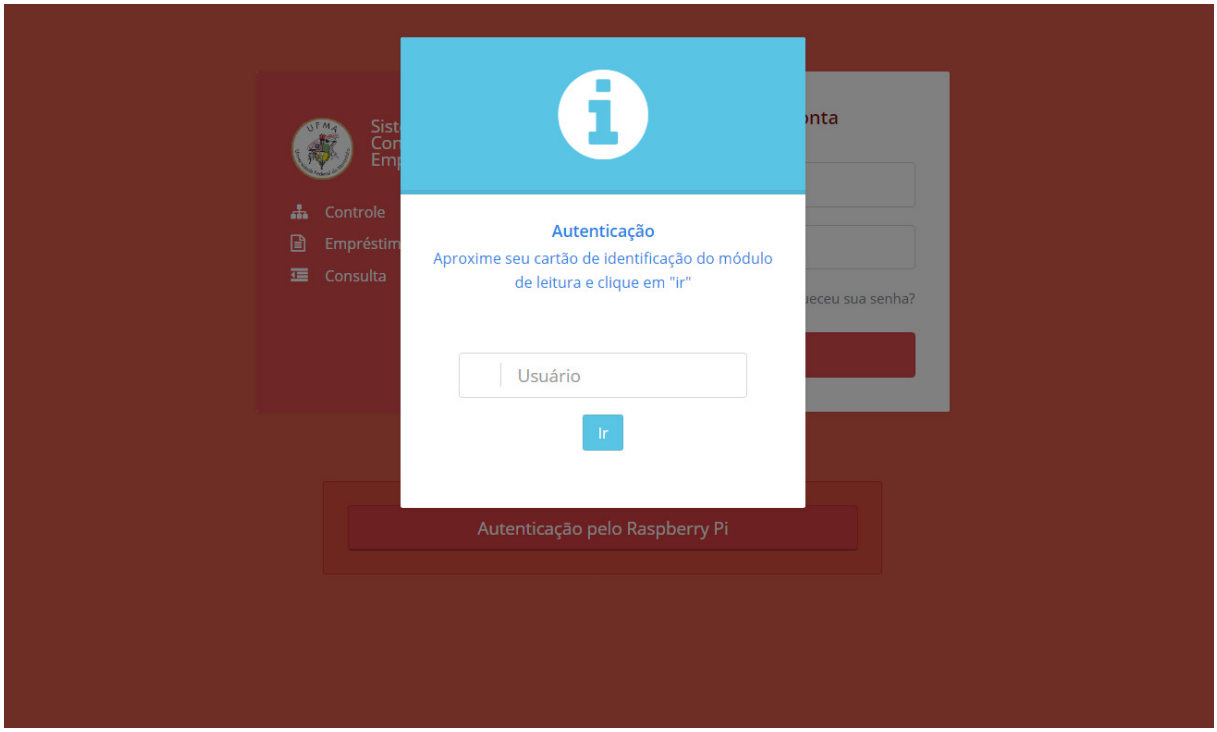
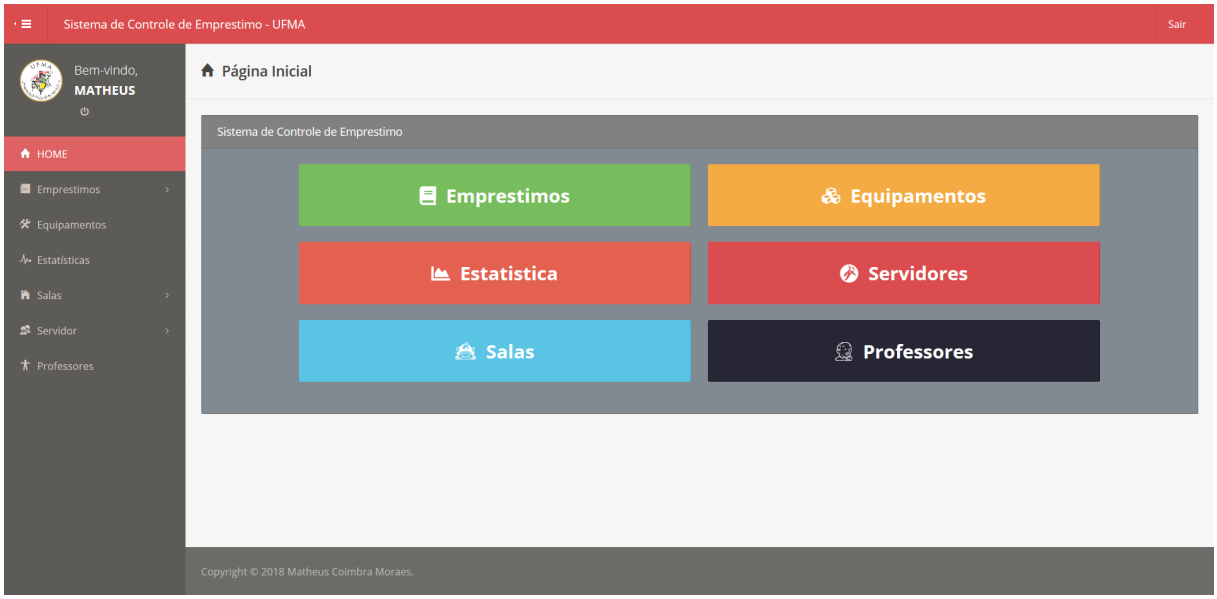


Figura 23 – Tela de Login da aplicação web.



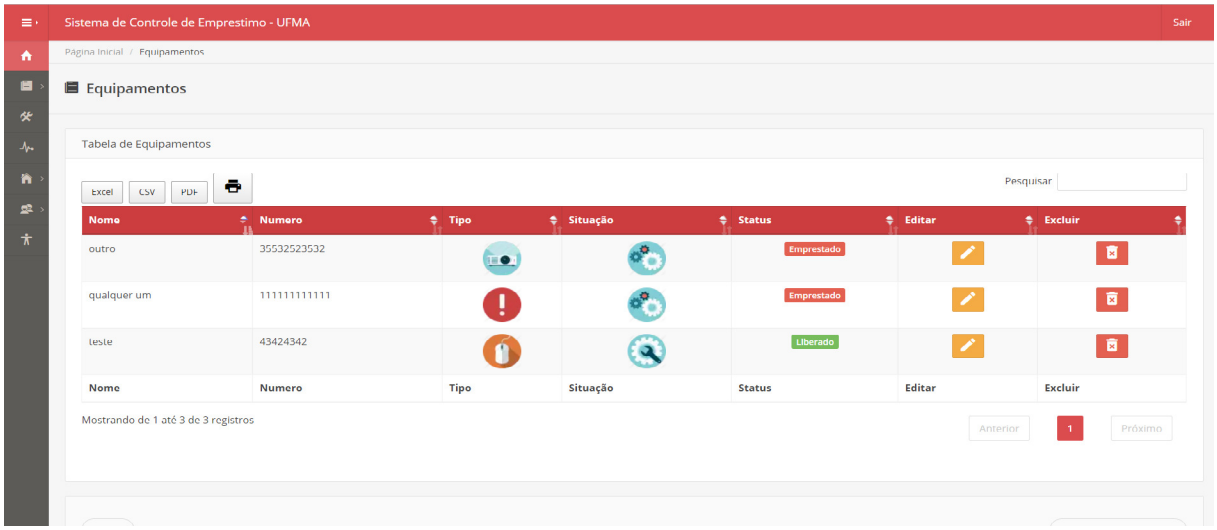
A figura 24 exibe a tela inicial, onde mostra todos os módulos da aplicação no painel principal. Observa-se ainda que no painel lateral esquerdo, é mostrado o usuário autenticado na aplicação e também as principais funcionalidades ao qual o usuário tem permissão.

Figura 24 – Tela inicial da aplicação web.



Na figura 25 são exibidos os equipamentos cadastrados pelo usuário do sistema. Observa-se que dependendo do equipamento cadastrado, o sistema mostra uma figura representativa do equipamento. Também é exibida a situação do equipamento através de figuras, na qual pode representar se um equipamento está disponível para ser utilizado ou está em manutenção. Mostra, ainda, o status do equipamento. Além disso, é possível editar e deletar um equipamento. Possibilita ainda a impressão e exportação da listagem em formato xlsx, csv e pdf.

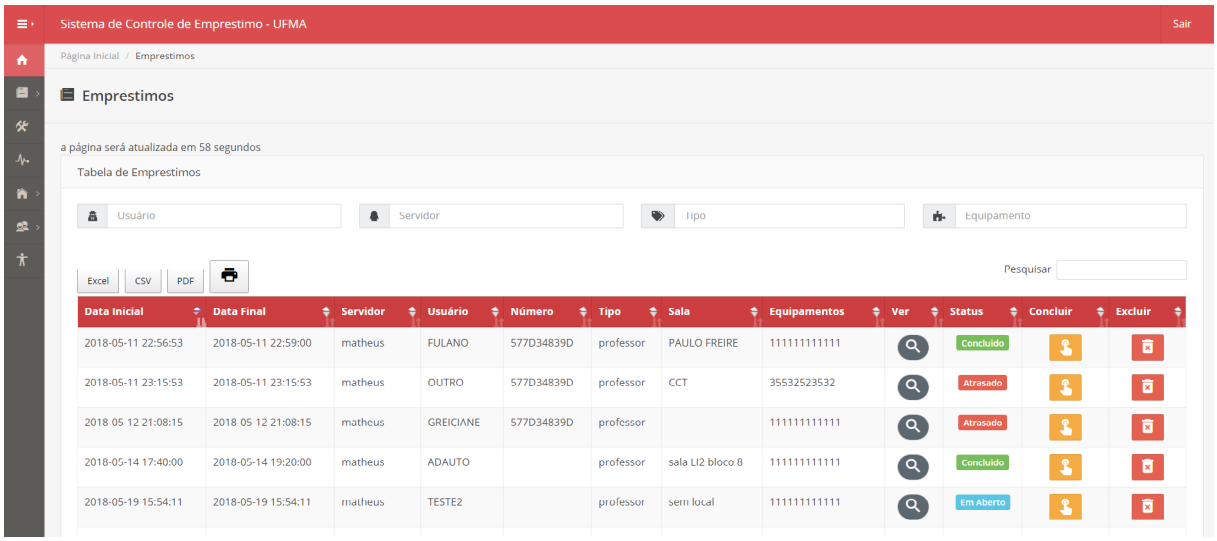
Figura 25 – Tela de equipamentos.



A figura 26 exibe um conjunto de campos de pesquisa para filtrar um empréstimo cadastrado. Exibe também a listagem dos empréstimos cadastrados pelo usuário do sistema, onde é exibida a situação do empréstimo e as informação pertinentes do empréstimo. Além disso, é possível concluir, deletar e detalhar um

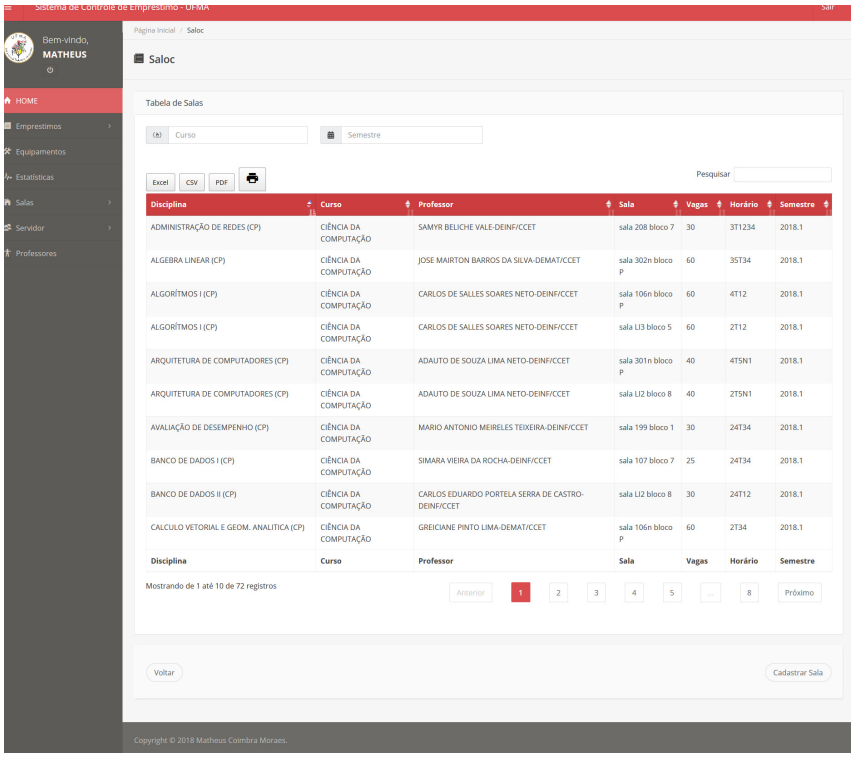
empréstimo. Possibilita ainda a impressão e exportação da listagem em formato xlsx, csv e pdf.

Figura 26 – Tela de empréstimos.



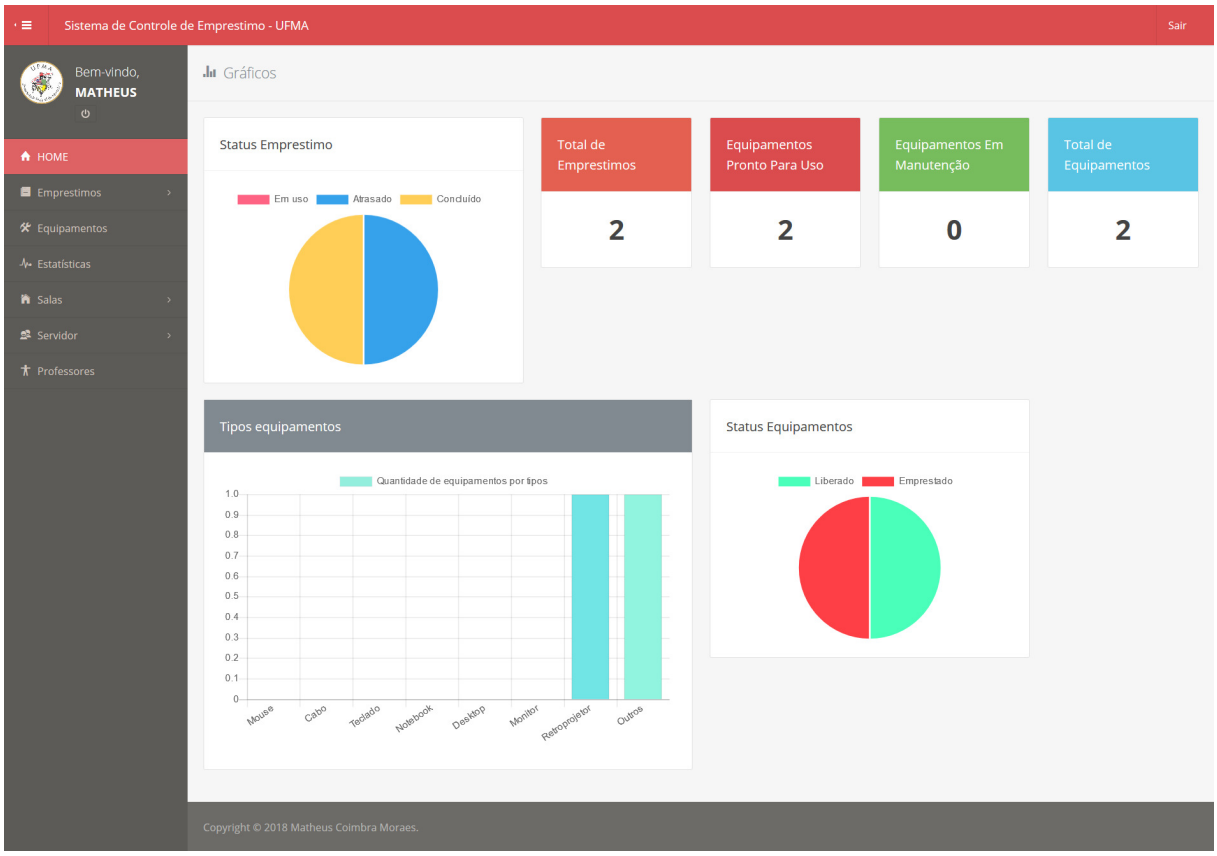
Na figura 27 exibe um conjunto de campos de pesquisa para filtrar uma sala. Exibe também a listagem das salas cadastradas. Além de possibilitar a impressão e exportação da listagem em formato xlsx, csv e pdf.

Figura 27 – Tela de salas.



Já a figura 28 mostra um conjunto gráficos e estatísticas pertinentes aos equipamentos e empréstimos.

Figura 28 – Tela de estatísticas.



6 CONCLUSÃO

O desenvolvimento do presente trabalho abordou os conceitos de computação em nuvem e internet das coisas, e como esses dois conceitos podem ser utilizados em conjunto para construção de uma arquitetura que tem como foco o controle dos bens físicos da Universidade Federal do Maranhão.

A comunicação com os módulos IoT e a aplicação web desenvolvida, que foi implantada na Nuvem da Amazon, mostrou desempenho satisfatório para o que foi proposto neste trabalho, podendo ser utilizado como uma ferramenta de controle de bens físicos de uma instituição. Toda arquitetura pode, sem grande esforço, ser implantada na Universidade Federal do Maranhão.

Toda a arquitetura desenvolvida serve para abordar os inúmeros benefícios que a computação em nuvem e internet das coisas podem trazer para o mercado da tecnologia. O uso dessas tecnologias proporciona um controle maior a respeito dos custos do desenvolvimento de uma aplicação, assim como uma maior portabilidade, maior segurança, facilidade de implantação de um projeto, maior interação entre os dispositivos utilizados na construção de um projeto, delegação das características de infraestrutura de computação para o fornecedor de serviços em nuvem. Todas essas vantagens e benefícios foram utilizados para construção do protótipo proposto.

É importante notar que a demanda por profissionais com competências em arquitetura em nuvem e internet das coisas tem aumentado consideravelmente nos últimos anos, deixando de lado a desconfiança de empresas ao adotar novas abordagens de desenvolvimento.

Vale ressaltar que os serviços prestados pelos provedores de nuvem possibilitou ainda mais o crescimento de projetos envolvendo internet das coisas. Com isso, o crescimento de uma dessas tecnologias, influencia diretamente no crescimento da outra. Espera-se que cada vez mais existam dispositivos comunicando entre si, coletando dados e, utilizando a computação em nuvem para processamento e inferência sobre os dados.

REFERÊNCIAS

A ZASLAVSKY A, G. V. K. S. C. M. *Internet of Things, Smart Spaces, and Next Generation Networks and Systems*. 1. ed. New York City: Springer-Verlag Berlin Heidelberg, 2014. 464 p.

ABC, E. **5 benefícios da computação em nuvem**. [S.l.], 2014. Disponível em: <https://exame.abril.com.br/geral/5-beneficios-da-computacao-em-nuvem/>.

AMAZON. **Computação em nuvem com a Amazon Web Services**. [S.l.], 2018. Disponível em: <https://aws.amazon.com/pt/what-is-aws/>.

BRAGA, N. C. **O básico sobre os Microcontroladores – parte 1 (MIC139)**. [S.l.], 2018. Disponível em: <http://newtoncbraga.com.br/index.php/eletronica/52-artigos-diversos/13263-o-basico-sobre-os-microcontroladores-parte-1-mic139>.

BUYYA, A. V. D. R. **Internet of Things Principles and Paradigms**. 1. ed. Cambridge, USA: ELSEVIER Inc, 2016. 354 p.

CLARK, K. J. **Integration architecture: Comparing web APIs with service-oriented architecture and enterprise application integration**. [S.l.], 2015. Disponível em: <https://www.ibm.com/developerworks/websphere/library/techarticles/1503_clark/1305_clark.html>.

COMMISSION, E. **A Platform for compute intensive applications in the cloud**. [S.l.], 2015. Disponível em: <https://ec.europa.eu/digital-single-market/en/blog/platform-compute-intensive-applications-cloud>.

COX, T. **Raspberry Pi Cookbook for Python Programmers**. 1. ed. BIRMINGHAM - MUMBAI: PACKT PUBLISHING, 2014.

D TRIFA V, M. F. W. E. G. **From the internet of things to the web of things: resource-oriented architecture and best practices.** 1. ed. Berlin Heidelberg: Springer, 2011. 97-129 p.

DOMINGUES, L. **Conhecendo o MongoDB Atlas – o DBaaS da MongoDB**. [S.l.], 2018. Disponível em: <https://codigosimples.net/2018/04/23/conhecendo-o-mongodb-atlas-o-dbaas-da-mongodb/>.

DUARTE, L. **Tutorial MongoDB para iniciantes em NoSQL**. [S.l.], 2017. Disponível em: <http://www.luiztools.com.br/post/tutorial-mongodb-para-iniciantes-em-nosql/>.

FONSECA, R. B. Nelson L.S. da. **Cloud Computing Bible**. 1. ed. Indianapolis: Wiley Publishing, Inc, 2011. 467 p.

FREMANTLE, P. **A reference architecture for the Internet of Things.** [S.l.], 2015. Disponível em: <https://wso2.com/wso2_resources/wso2_whitepaper_a-reference-architecture-for-the-internet-of-things.pdf>.

GONCALVES, G. **Protocolos para Internet das Coisas**. [S.l.], 2016. Disponível em: <https://jualabs.wordpress.com/2016/04/13/protocolos-para-internet-das-coisas/>.

INC., L. S. **O que é um diagrama UML?** [S.l.], 2018. Disponível em: <https://www.lucidchart.com/pages/pt/o-que-e-uml>.

MATIAS, A. C. **Gestão Patrimonial: Contribuição para o controle de bens móveis na Universidade Federal do Rio Grande do Norte**. 1. ed. João Pessoa, RN: [s.n.], 2015.

PEREIRA, G. A. N. **Modelagem de dados com MongoDB**. [S.l.], 2017. Disponível em: <https://www.devmedia.com.br/modelagem-de-dados-com-mongodb/38125>.

PIMENTEL, P. **Começando com o AWS IoT Parte 1**. [S.l.], 2016. Disponível em: <https://aws.amazon.com/pt/blogs/aws-brasil/comecando-com-o-aws-iot-parte-1/>.

R.LECHETA, R. **AWS para desenvolvedores.** 1. ed. São Paulo, SP: novatec, 2014. 97-129 p.

ROCHA, B. C. **What the Flask? Pt-1 Introdução ao desenvolvimento web com Python**. [S.l.], 2014. Disponível em: <http://pythonclub.com.br/what-the-flask-pt-1-introducao-ao-desenvolvimento-web-com-python.html>.

ROSE SCOTT ELDRIDGE, L. C. K. **THE INTERNET OF THINGS: AN OVERVIEW**. 1. ed. Virgínia, EUA: The Internet Society, 2015. 18-22 p.

SOSINSKY, B. **CLOUD SERVICES, NETWORKING, AND MANAGEMENT**. 1. ed. Indianapolis: Wiley Publishing, Inc, 2015. 408 p.

SOUZA, G. R. de. **Diagrama de Sequência**. [S.l.], 2013. Disponível em: <https://docente.ifrn.edu.br/givanaldorocha/disciplinas/engenharia-de-software-licenciatura-em-informatica/diagrama-de-sequencia>.

TAURION, C. **Entendendo o modelo Multi-tenancy**. [S.l.], 2010. Disponível em: <https://imasters.com.br/artigo/19067/cloud/entendendo-o-modelo-multi-tenancy/>.

UDACITY. **8 motivos para aprender a programar em Python**. [S.l.], 2017. Disponível em: <https://br.udacity.com/blog/post/motivos-aprender-programar-python>.

VERMESAN, D. P. F. D. O. **Internet of Things-From Research and Innovation to Makert Deployment**. 1. ed. Denmark: River Publishers, 2014. 351 p.

VICENZI, A. **Controle de acesso RFID com Raspberry Pi**. [S.l.], 2016. Disponível em: <https://www.filipeflop.com/blog/controle-de-acesso-rfid-com-raspberry-pi/>.

VIECELLI, A. M. M. E. **A IMPORTÂNCIA DO CONTROLE PATRIMONIAL PARA AS ENTIDADES PÚBLICAS: UM ESTUDO DE CASO NO CENTRO DE EDUCAÇÃO SUPERIOR DO NORTE DO RIO GRANDE DO SUL (CESNORS)**. 1. ed. RS-Brasil: revistas uri, 2014. 19 p.

WARNER, T. L. **Hacking Raspberry Pi**. 1. ed. Indianapolis, Indiana 46240 USA: Que Publishing, 2014. 30-40 p.

YUAN, M. **Conhecendo o MQTT**. [S.l.], 2017. Disponível em: <https://www.ibm.com/developerworks/br/library/iot-mqtt-why-good-for-iot/index.html>.

Apêndices

APÊNDICE A – IMPLEMENTAÇÃO DO LEITOR DE RFID

Código para enviar RFID para o sistema web:

```
try:
    # Inicia o modulo RC522 e a conexao com o AWS IoT
    LeitorRFID = MFRC522.MFRC522()
    myAWSIoTMQTTClient.connect()

    print('Aproxime seu cartao RFID.')

    while True:

        status, tag_type = LeitorRFID.MFRC522_Request(LeitorRFID.PICC_REQIDL)

        if status == LeitorRFID.MI_OK:
            print('Cartao detectado!')

          # Efetua leitura do UID do cartao.
            status, uid = LeitorRFID.MFRC522_Anticoll()

            if status == LeitorRFID.MI_OK:
                uid = ''.join(['%X' % x for x in uid])
                rfid=uid
                print('UID do cartao: %s' % uid)
                data = {'rfid':uid}

                op = int(input("1-enviar rfid para AWS\n2Autenticar Servidor\n3-
                    visualizar ID\n4-SAIR(CRTL + C) "))
                if op == 1:

                    thing = "raspberry"
                    topic = "dispositivo/raspberry/reportar"
                    print(thing)
                    rfid = str(rfid)
                    data = { "thing": thing,
                        "thing-type": "rfidreader",
                        "rfid": str(rfid)
                    }
                    myAWSIoTMQTTClient.publish(topic, json.dumps(data), 1)

                    time.sleep(2)
                if op == 2:
                    response = requests.post('http://34.224.28.134/pi',json=data)
                if op == 3:
                      print('UID do cartao: %s' % uid)
                print('nAproxime seu cartao RFID')

        time.sleep(.25)
except KeyboardInterrupt:
```

```
    GPIO.cleanup()
    print('nPrograma encerrado.')
```

Código A.1 – Leitor RFID