JOÃO FRANCISCO FERREIRA LUCINDO

SISTEMA DE MONITORAMENTO BASEADO EM ARQUITETURA DE INTERNET OF THINGS

VIÇOSA – MG
JULHO – 2017

JOÃO FRANCISCO FERREIRA LUCINDO

SISTEMA DE MONITORAMENTO BASEADO EM ARQUITETURA DE INTERNET OF THINGS

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Orientador: Prof. Dr. André Gomes Torres.
Coorientadora: Prof.ª Dr.ª Kétia Soares Moreira

VIÇOSA – MG
JULHO – 2017

JOÃO FRANCISCO FERREIRA LUCINDO

SISTEMA DE MONITORAMENTO BASEADO EM ARQUITETURA DE INTERNET OF THINGS

Monografia apresentada ao Departamento de Engenharia Elétrica do Centro de Ciências Exatas e Tecnológicas da Universidade Federal de Viçosa, para a obtenção dos créditos da disciplina ELT 490 – Monografia e Seminário e cumprimento do requisito parcial para obtenção do grau de Bacharel em Engenharia Elétrica.

Aprovada em 11 de Julho de 2017

COMISSÃO EXAMINADORA

Professor Dr. André Gomes Tôrres - Orientador
Universidade Federal de Viçosa

Professora Dr.ª Kétia Soares Moreira – Coorientadora
Universidade Federal de Viçosa

BSc. Ítalo Nógueira Soares
Universidade Federal de Viçosa

VIÇOSA – MG
JULHO – 2017

“Data is not information, information is not knowledge, knowledge is not understanding, understanding is not wisdom.”

Clifford Stoll

Dedico esse trabalho aos meus pais
e ao meu Avô José Lucindo Ribeiro

Agradecimentos

Agradeço aos meus familiares, especialmente meus pais João e Lúcia Lucindo por sempre me apoiarem e estarem juntos sempre.

A minha namorada Lívia, pelo companheirismo.

A todos os amigos que de alguma forma me ajudaram.

Resumo

Sendo parte da Quarta Revolução Industrial [40] sistemas baseados em arquitetura de Internet of Things (IoT) serão, em um curto espaço de tempo, parte cotidiano da população e indústria abrindo um mercado multibilionário. Todavia, IoT ainda é uma área da tecnologia com muitos desafios a serem e ainda há muito espaço para o estudo e desenvolvimento de novas propostas. O objetivo desse trabalho é levantar e propor uma arquitetura baseada em recursos computacionais em nuvem que seja capaz de monitorar e coletar dados de um ambiente com latência baixa, e em paralelo armazenar todos os dados coletados em um banco de dados. A fim de ilustração da capacidade da arquitetura desenvolvida, propôs-se um cenário de monitoração de prevenção de incêndios.

Um Arduino Uno R3 conectado a diversos sensores foi o responsável por originar os dados. Um módulo Wifi ESP8266-01 fazia a interconexão com a nuvem. Utilizou-se a plataforma ThingSpeak para o tratamento dos dados em tempo real (praticamente) e também para o envio de alertas e notificações, interconectado com o Twitter. Já para armazenar os dados enviados pelo módulo Wifi, utilizou-se dois serviços de nuvem: WebApp Azure e SQL Azure Database.

Então conseguiu-se, como resultado, duas análises: A primeira em praticamente tempo real, onde apresentou-se uma análise dos últimos dados coletados do ambiente; a segunda com uma abordagem histórica, onde analisa-se toda a massa de dados coletados. Também fez-se um sistema de alertas e notificação baseado no Rede Social Twitter, por onde é possível notificar ou alertar qualquer mudança crítica no ambiente.

Palavras-chave: IoT; Computação em nuvem; Arduino; Dados; ESP8266-01.

Abstract

As part of the Fourth Industrial Revolution, systems based on Internet of Things (IoT) will be a daily part of both the population and the industry, opening up a multibillion-dollar market opportunity. However, IoT is still a technology area with many challenges to be addressed, there is still much to study and develop. The main objective of this work is to raise and propose a cloud computing based architecture that is capable of monitoring and collecting data from an environment with low latency, storing all the data collected in a database. To illustrate the capacity of the developed architecture, a fire prevention monitoring scenario was proposed.

An Arduino Uno R3 connected to several sensors was responsible for originating the data. An ESP8266-01 Wifi module was interfacing with the cloud. The ThingSpeak platform was used for both data treatment in real time (practically) and sending alerts and notifications, interconnected with Twitter. To store the data sent by the Wifi module, two cloud services were used: WebApp Azure and SQL Azure Database.

Two reviews were then obtained as a result: The first one practically in real time, where an analysis of the last data collected from the environment was presented; the second one with a historical approach, which analyzes the entire mass of data collected. It was also made a system of alerts and notification based on Social Network Twitter, where you can notify or alert any critical changes in the environment.

Keywords: IoT; Cloud computing; Arduino; Data; ESP8266-01.

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

Nas últimas duas décadas, pesquisadores e o mercado vem levantando discussões sobre sistemas inteligentes que podem mudar a maneira em que as pessoas vivem, trazendo muito mais conforto e comodidade e também podem mudar a maneira em que as empresas enxergam os seus negócios, tendo muito mais visibilidade dos processos e baseando as tomadas de decisões de negócios em análise de dados. [5]

Segundo Kevin Ashton [8], tais sistemas inteligentes seriam capazes de entender o ambiente em que estão inseridos através de sensores poderiam coletar dados, armazenar e monitorar vários processos possibilitando a redução de custo e otimização na utilização de recursos.

Mas junto com a ambição de tais sistemas inteligentes surgem vários problemas e limitações: como garantir a segurança dos dados sensíveis? Como disponibilizar uma infraestrutura capaz de suportar um enorme tráfego de dados? Como garantir a privacidade? [5].

Grande parte dessas questões foram e estão sendo resolvidas com o advento da disponibilidade de recursos computacionais em nuvem, onde grandes empresas estão fazendo investimentos bilionários para garantir o desenvolvimento de sistemas inteligentes de uma maneira segura e escalável, atendendo os requisitos dos mais diversos países e instituições. [2]

Inserido nesse contexto de sistemas inteligentes surge o conceito IoT (Internet Of Things - Internet das Coisas), introduzido por Kevin Ashton em 1999 [3]. Segundo Dorsemaine et al em [5], IoT pode ser definido como: "Grupo de objetos comunicantes interligados, que coletem dados e permitam a gestão e o acesso aos dados por eles gerados."

Dentro desse conceito de IoT surgem sistemas que tem como objetivo:

- Sistemas com sensores embarcados que sejam capazes de coletar dados como: localização, vibração, temperatura, umidade, intensidade de luz branca, intensidade de luz infravermelha, intensidade de ruídos, presença ou não de gases no ambiente.

- Sistemas que permitem equipamentos e maquinários em diferentes redes se comunicarem. Possibilitando assim o desenvolvimento de sistemas de automação industrial ou residencial.
- Possibilitar que qualquer pessoa e objeto se conecte a internet para a troca de informação de qualquer lugar, usando qualquer dispositivo a qualquer momento.

Exemplos de aplicações de tais sistemas são:

- Dispositivos que detectam quedas e monitoram sinais vitais: pode permitir que pessoas idosas ou com alguma deficiência tenham uma vida mais autônoma e ao mesmo tempo segura
- Dispositivos que monitoram a temperatura: podem ser aplicados em recipientes que armazenem vacinas e medicamentos sensíveis à variação de temperatura, dando a segurança para o paciente que o medicamento a ele ministrado está em perfeitas condições
- Dispositivos que monitoram ambientes críticos: Podem ser empregados a depósitos de mercadorias inflamáveis e com alto valor agregado. Assim pode-se monitorar de qualquer lugar ou horário a situação daquele ambiente sensível.

Tendo em vista o exposto o presente trabalho trará a discussão de uma proposta de arquitetura para um sistema IoT que poderá ser empregado aos mais diversos cenários. A fim de demonstração neste trabalho será abordado o desenvolvimento de um sistema de monitoração de incêndios.

2 Objetivos

2.1 Objetivo Geral

Este trabalho tem como objetivo geral propor um sistema de monitoramento baseado em arquitetura IoT. A arquitetura aqui proposta é aplicada, a fins de demonstração, a um sistema de monitoramento de incêndios. Todavia, com algumas pequenas modificações pode ser aplicada a qualquer outro cenário de monitoramento.

2.2 Objetivos Específicos:

- Disponibilizar uma análise em tempo real dos dados coletados;
- Disponibilizar uma análise histórica dos dados coletados;
- Disponibilizar essas análises não só em um site Web, como também para qualquer dispositivo móvel;
- Desenvolver um sistema de alertas e notificações sobre o ambiente;

3 Revisão de Literatura

3.1 IoT - Internet of Things

Em 1999, Kevin Ashton, visionário, pesquisador e fundador do centro de pesquisa Auto-ID do MIT teve a seguinte ideia em tradução livre:

> "Se tivéssemos computadores capazes de entender e abstrair todo o ambiente ao seu redor - podendo capturar e armazenar dados automaticamente, assim poderíamos monitorar, contabilizar, reduzir custos e evitar desperdícios. Seria possível prever quando as "coisas" precisam de reparo ou substituição, quando estão aptas ou não para desempenhar sua função. Então precisamos capacitar os computadores para coletar dados e interagir com o ambiente, por si só, sem a limitação da necessidade de operação humana. A Tecnologia RFID e sensores permitirão computadores observar, identificar e compreender o mundo." [Lopez Research]

No mesmo ano, durante uma apresentação para a Procter & Gamble (P&G), Ashton cunhou o termo "Internet of Things (IoT)" ao se referir como a tecnologia RFID por ele desenvolvida podia otimizar a logística da multinacional possibilitando conectar "coisas" à Internet [3].

Naquela época ainda existiam muitos desafios para se possibilitar a sua ideia visionária de Kevin Ashton, mas atualmente com avanços da tecnologia nas áreas de computação em nuvem, hardware, protocolos de comunicação sem fio e rede móvel possibilitam a conectividade de bilhões de "coisas" à internet de uma maneira segura e barata.

Desde então vários outros expoentes da literatura vêm trazendo sua própria definição de IoT. Em 2012 o ITU (International Telecommunication Union) [20] definiu IoT como:

> "Uma infraestrutura global disponível para a sociedade, permitindo o desenvolvimento de serviços avançados através da interconexão de elementos (físicos e virtuais) com base em tecnologias de informação e de comunicação interoperáveis existentes e em evolução "

Nota 1 - Através das capacidades de identificação, captura de dados, processamento e comunicação, o IoT faz pleno uso das "coisas" para oferecer serviços para as mais diversas aplicações, assegurando simultaneamente as exigências de segurança e privacidade.

Nota 2 - Em uma perspectiva mais ampla, o IoT pode ser concebido como um conceito que impactará a tecnologia e a sociedade

Segundo o CERP-IoT (European Research Cluster on the Internet of Things) [46], IoT é, em tradução livre,

"Uma rede de infraestrutura dinâmica e global com a capacidade de autoconfiguração baseada em protocolos de comunicação padronizados e interoperáveis onde 'coisas' físicas e virtuais têm identidade, atributo físico, personalidades virtuais utilizando interfaces inteligentes, sendo perfeitamente integrada com a Internet".

Conforme a representação apresentada na Figura 1, feita pelo grupo europeu.

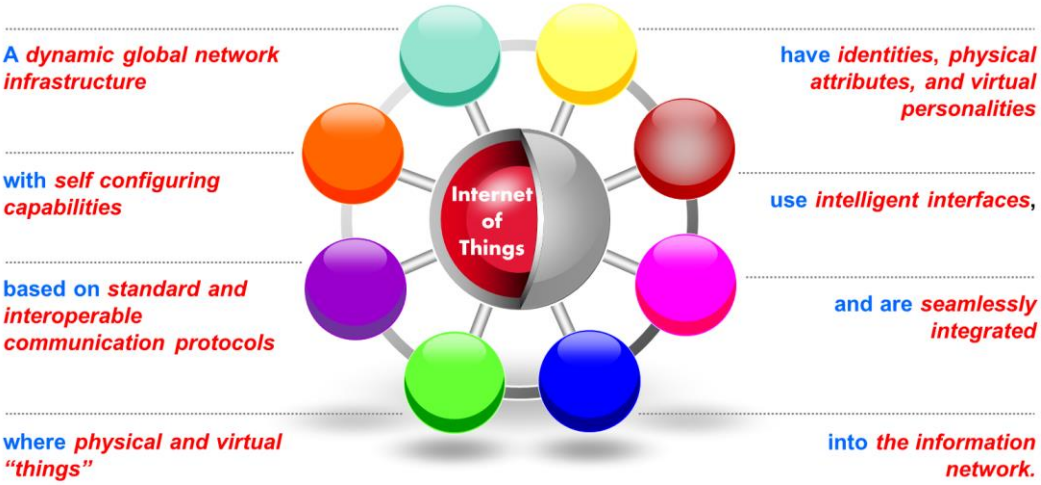

Figura 1- Infográfico definição IOT pelo Cerp [46]

Em acréscimo, o IoT é capaz de integrar e conectar diversas tecnologias através da Internet, permitindo que as "coisas" se interajam, troquem informação e cooperar para atingir um objetivo em comum, podendo assim entender e agir sobre o ambiente em que estão inseridas, muitas vezes sem intervenção humana. O que transforma um simples ambiente em um ambiente inteligente.

Sistemas IoT possibilitam que pessoas e objetos estejam conectados à internet de qualquer lugar, usando qualquer tipo de dispositivo (smartphone, computadores de mesa, microcontroladores, equipamentos industriais, redes sociais, entre outros) conectado a qualquer rede de Internet ou serviço, a qualquer momento ou contexto. Conforme Vermesan em [46], ilustrado na Figura 2. Existem vários artigos, vídeos, conferências para discutir as vantagens e desvantagens da Internet of Things na sociedade e na indústria. Devido à essa nova tecnologia, surgem novos produtos, modelos de negócio e ferramentas de otimização de processos.

Do ponto de vista do consumidor, a promessa é que os ambientes sejam cada vez mais inteligentes: Casa Inteligente, Carro Inteligente, Cidade Inteligente, Hospital Inteligente. O que trará com toda certeza mais conforto, comodidade, capacidade de monitoração, automação e eficiência energética através dos sensores e atuadores inerentes a um sistema IoT.

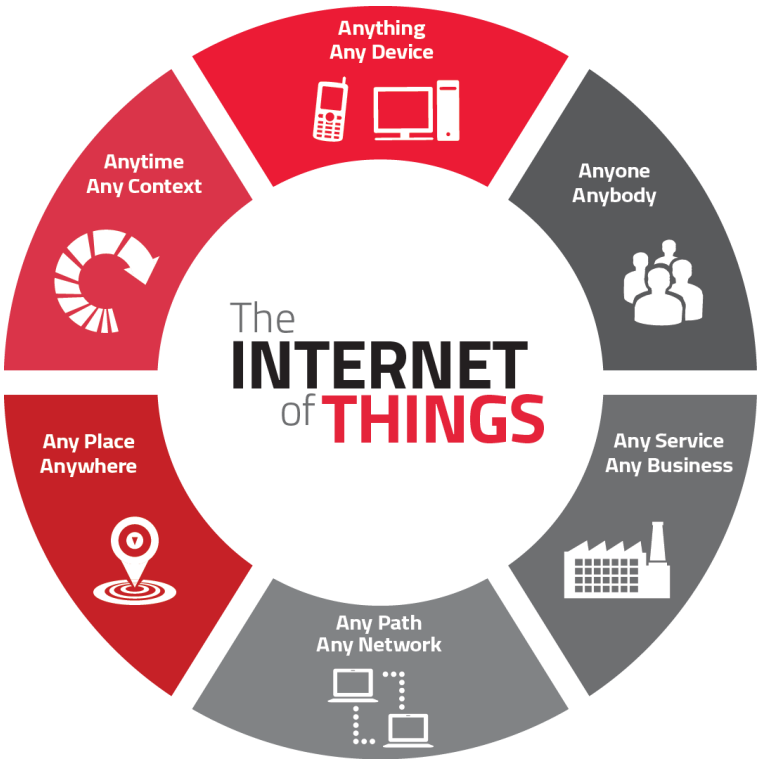

Figura 2 - Infográfico Conceito IoT [46]

Para a indústria, além de todos os benefícios supracitados, abre-se um leque de possibilidades de otimização de recursos através da agregação de valor aos dados coletados pelos sistemas IoT, podendo-se utilizar de ferramentas de análise preditiva e identificação de padrões para otimizar os mais variados processos.

Todavia, alguns pesquisadores alertam para os perigos iminentes do IoT, como a exposição de dados, violação de privacidade e segurança. O que pode ser um problema tanto para consumidores finais, tanto para a indústria. Ainda é preciso desenvolver muito para que sistemas IoT sejam implementados em toda a sua potencialidade.

A consultoria Gartner, mundialmente respeitada, desenvolve periodicamente um infográfico chamado Hype Cycle. O último Hype Cycle publicamente divulgado é apresentado na Figura 3 e está disponível em [15].

Esse gráfico posiciona as tecnologias emergentes em cinco diferentes graus de maturidade, segundo Gartner [15] toda e qualquer tecnologia irá passar por todas essas cinco fases apresentadas a seguir:

- Innovation Trigger (Gatilho de inovação): o primeiro grau onde uma possível tecnologia inovadora começa a ser discutida e divulgada. Nessa etapa são feitas as primeiras experimentações e provas de conceito. Muitas vezes não há disponibilidade do produto no mercado, pois a sua viabilidade econômica ainda não foi comprovada

pelo mercado

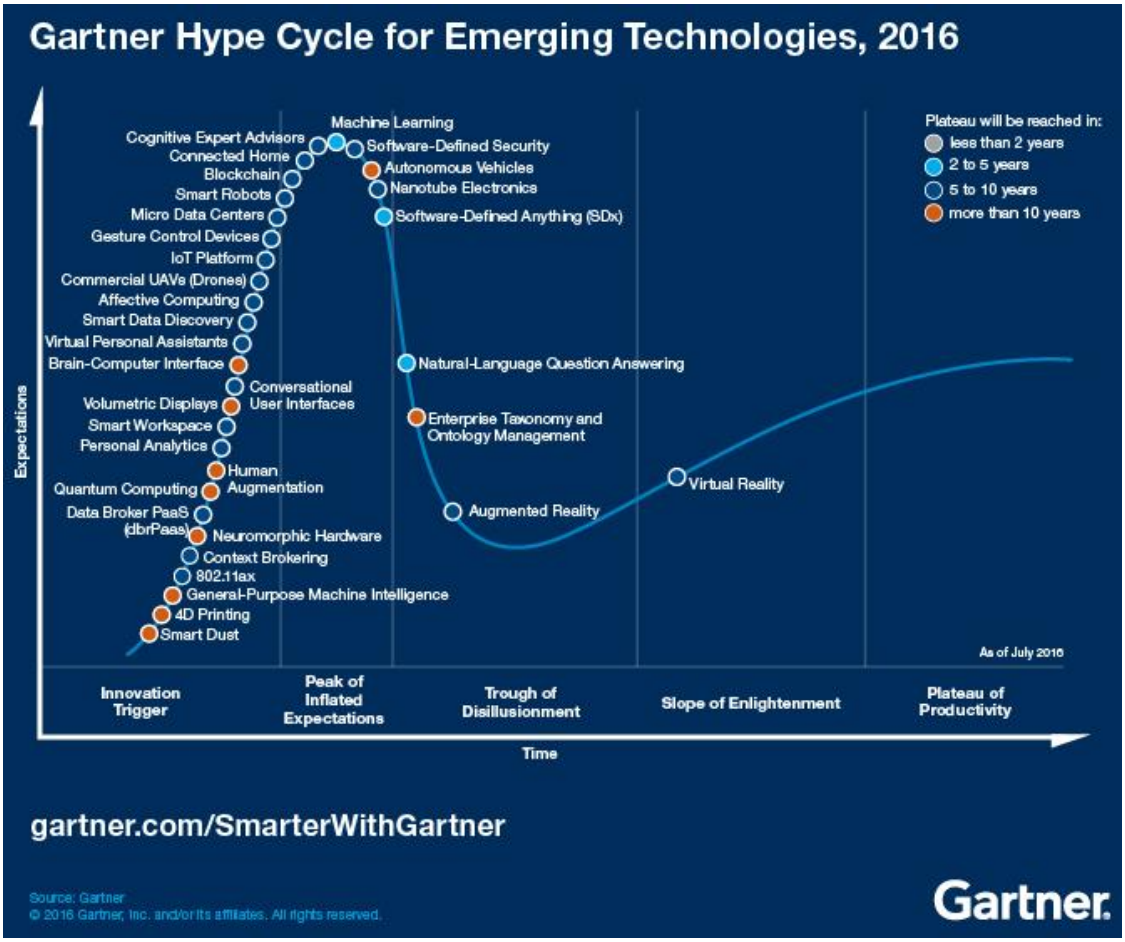

Figura 3 - Hype Cicle 2016 [15]

- Peak of Inflated Expectation (Pico da expectativa): é a etapa onde tem-se o pico máximo das expectativas. A mídia em geral produz uma série de casos de sucesso e de falha. Nesse ponto as empresas começam a se aprofundar na tecnologia, gerando muito conhecimento.
- Trough of Disillusionment (Vale da desilusão): Nessa etapa é onde o interesse é o menor. Muitas empresas cessam os investimentos, pois junto com o aprofundamento na tecnologia surgem muitos problemas, experimentações falham e poucas empresas arriscam em continuar investindo em uma tecnologia repleta de falhas.
- Slope of Enlightenment: Nessa fase, as empresas que continuaram a investir na tecnologia no Vale da desilusão começam a superar os problemas, e o mercado volta a acreditar na tecnologia. Exemplos de aplicação na prática começam a aparecer e consequente o retorno financeiro.

- Plateau of Productivity (Plateau de Produtividade): Nessa fase, a tecnologia já é melhor compreendida e aceita pelo mercado e começa a ser produzida em escala, marcando a sua consolidação.

Dentro dessa classificação, o Gartner posiciona o IoT na fase Innovation Trigger juntamente com várias outras tecnologias, como: Computação Quântica, Interface Cérebro-Computador, Drones, entre outras. Estima-se de 5 a 10 anos para a plataforma IoT evoluir para a próxima etapa [15]. Ou seja, ainda há muito a ser discutir sobre o tema. É uma área ainda aberta para se desenvolver e solucionar muito problemas. Um tema de estudo motivador para muitos pesquisadores que resulta em oportunidades de negócios e desafios globais. Atualmente fóruns e congressos de âmbito internacional e nacional são organizados para reunir a comunidade e discutir sobre os problemas encontrados.

3.1.1 Principais Características de um sistema IoT

Segundo o ITU (International Telecommunication Union) um sistema IoT deve apresentar 5 características essenciais, sendo elas: Interconectividade, serviços relacionados, heterogeneidade, mudança dinâmica e alta escalabilidade [20]. Cada uma dessas características pode ser definida como:

- Interconectividade: toda e qualquer "coisa" deve ser interconectada com alguma infraestrutura de comunicação ou transmissão de dados, como por exemplo: Internet, bluetooth, satélite, infravermelho, microondas, entre outros.
- Serviços relacionados: toda solução IoT deve fornecer algum serviço, deve ter alguma finalidade, como por exemplo: monitoramento, coleta de dados, atuação sobre o ambiente de forma automática ou não, entre outros. Todos esses serviços devem de alguma forma garantir a segurança a privacidade com coerência semântica entre o mundo físico (device) e seu correspondente virtual (dados).
- Heterogeneidade: Existem no mercado uma gama de devices IoT, que foram projetos baseados em diferentes plataformas de hardware e rede. Uma solução IoT deve suportar essa heterogeneidade, possibilitando a interação entre os devices através de diferentes redes.
- Mudanças Dinâmicas: toda solução IoT deve suportar diferentes tipos de mudanças. O estado dos devices podem mudar a todo tempo: passar de modo ativo para de hibernação, conectar e desconectar, mudar de localização geográfica. Também deve ser

capaz de suportar de uma maneira dinâmica a mudança no número de devices que compõem a solução.

- Alta escalabilidade: O número de devices IoT tende a crescer exponencialmente, previsões de que o número de devices conectados chegue a dezenas de bilhões [11]. Então uma solução IoT deve ser capaz de suportar de maneira escalável o crescimento. Devido a essa demanda crescente, a escolha de uma plataforma baseada em computação em nuvem é a melhor escolha, como será discutido com mais detalhes ao decorrer desse trabalho.

3.1.2 Papel na Economia

O papel principal do IoT na transformação dos negócios se dá pela capacidade de conectividade entre diversas áreas. IoT também conhecido como a rede das redes, interconectando áreas como educação, transporte, negócios, energia, construção civil, entre outras de uma maneira segura, analítica e gerenciável [11]. Conforme ilustra a Figura 4.

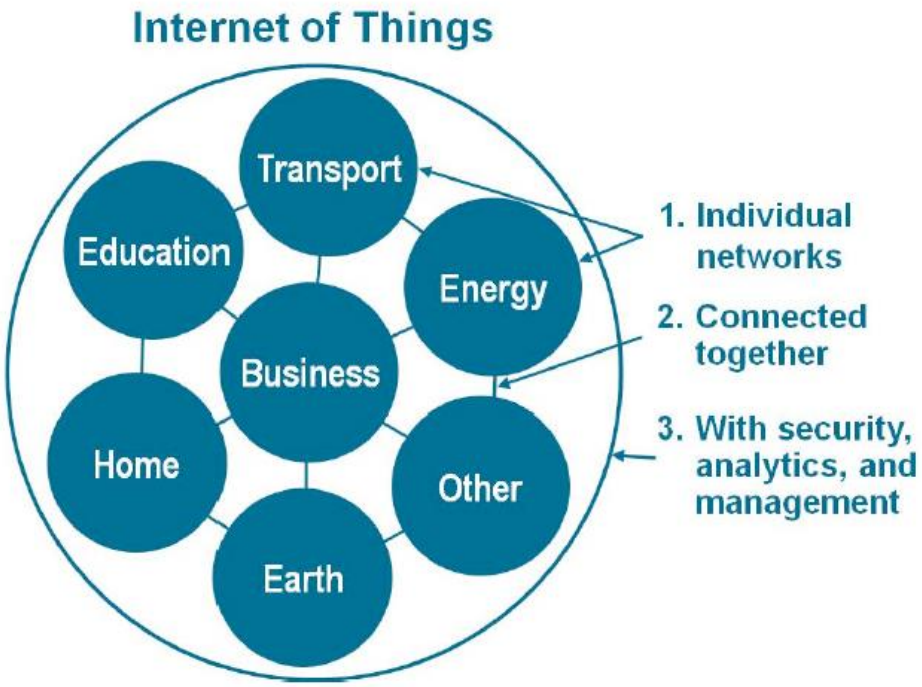

Figura 4 – IoT: Rede de Redes [11]

Essa integração entre diversas áreas traz uma grande massa de dados que combinados com a informação podem ser transformados em conhecimento. O conhecimento, juntamente com a experiência traz a sabedoria [11], que proporciona a otimização de processos e

consequentemente vantagens econômicas e melhores tomadas de decisão [42]. Segundo Mikkers em [37], dado é o petróleo do futuro e dominará o mercado aquele que souber analisar esses dados, transformando-os em sabedoria.

O processo de transformação do dado em sabedoria é ilustrado na Figura 5, disponível em [11], onde evidencia a agregação de valor ao dado bruto, até o transformar em sabedoria.

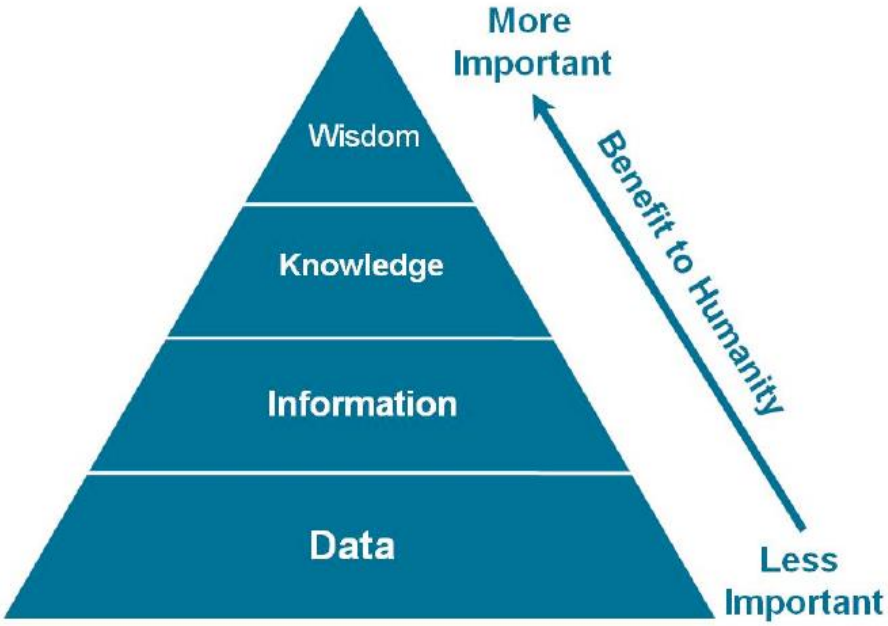

Figura 5 – Processo Mineração de Dados [11]

Vale ressaltar que, nesse modelo, há uma correlação direta entre a entrada (dado bruto) e a saída (conhecimento). Quanto mais dado é criado, mais sabedoria e conhecimento pode-se obter. Logo quanto mais consolidado os sistemas de IoT, mais conhecimento e sabedoria será produzido [11].

Estudos feitos pela Cisco conforme a Figura 6, apontam que, em meados de 2008, o número de devices conectados já era maior que o número de habitantes do planeta. Atualmente já são mais de três devices conectados por pessoa e estima-se que em 2020 existirão 50 bilhões de devices conectados, representando mais de seis devices conectados por pessoa. Salienta-se ainda que essa média está sendo feita considerando que a população inteira do planeta tenha acesso à essas tecnologias, o que se sabe que não é verdade. Levando em consideração que em 2010, somente 2 bilhões de pessoas tinham acesso à internet, tem-se a elevação da média de 1.84 dispositivos/pessoa para assustadores 6.25 dispositivos/pessoa [11].

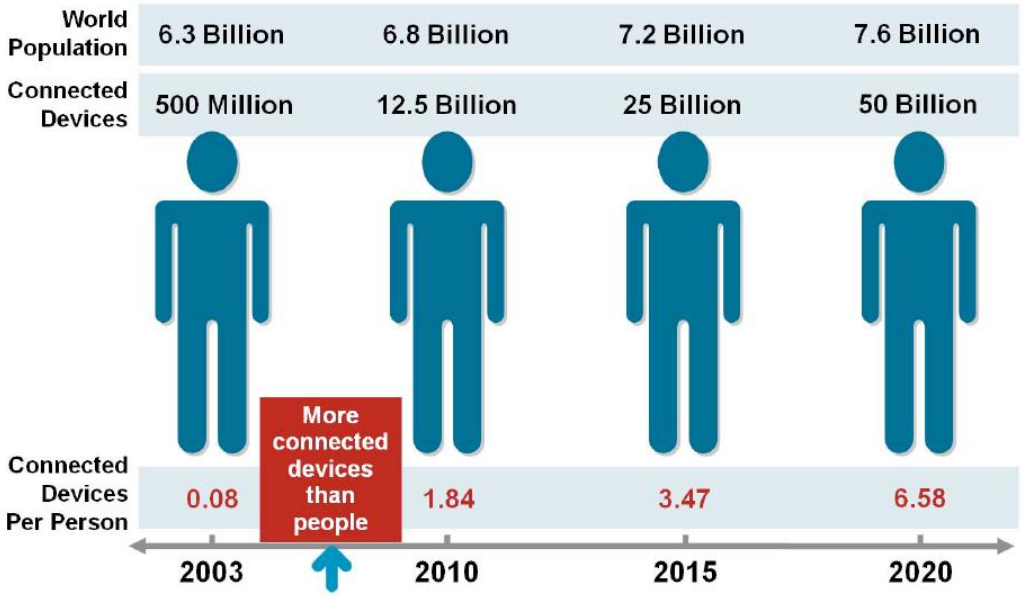

Figura 6 – Proporção Dispositivos/Pessoa ao Longo do Tempo [11]

A adoção de sistemas IoT está sendo acelerada pela computação em nuvem, o que permite escalar até centenas de milhões de devices conectados de maneira segura, sendo assim não só pessoas, mas também grandes empresas estão usando IoT para resolver problemas de negócios. Mais a seguir serão discutidos casos de aplicação de IoT vários segmentos da indústria.

Devido a essa adoção em massa das pessoas e das indústrias, estima-se que, em 2020, 600 Zettabytes serão gerados por pessoas, máquinas e sistemas. A Figura 7 mostra como esses dados serão gerados dentro do conceito de Cidade Inteligente.

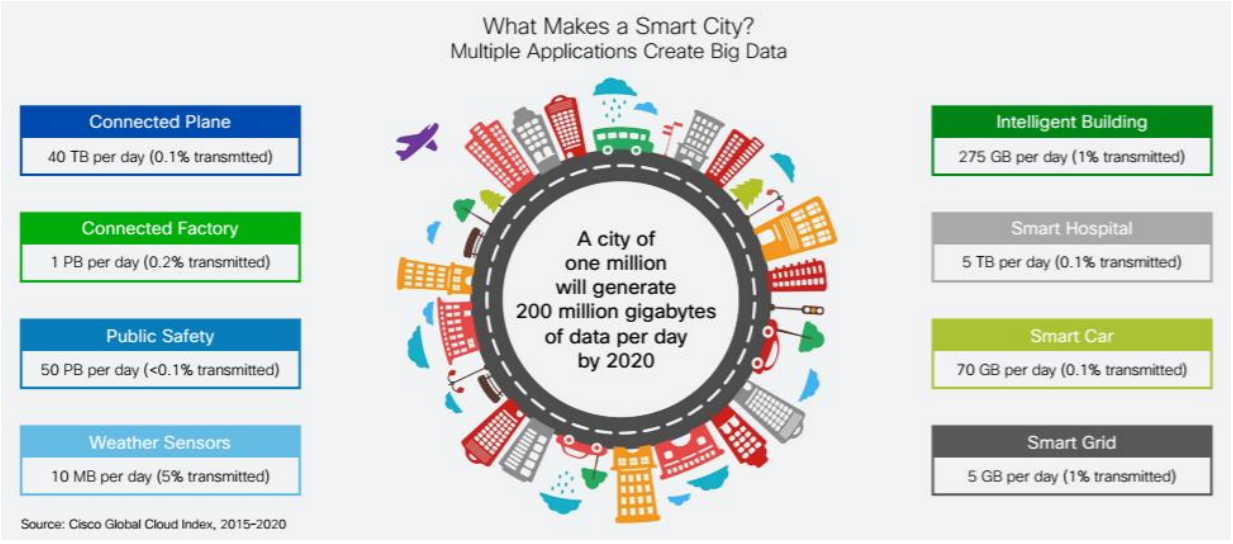

Figura 7 – Cidade Inteligente [11]

A maior parte dos dados, 50 petabytes, serão gerados por sistemas de monitoramento de segurança pública, dados esses vindos de câmeras e sensores dos mais variados tipos. Nota-se também que casas, hospitais, carros, redes de transmissão de energia, sistema aéreo também serão grandes geradores de dados [11]. E como já mencionado anteriormente, dados e sabedoria estão intimamente relacionados.

No cenário de cidade inteligente, com toda a integração de sistemas será possível, por exemplo, que os gestores públicos façam um melhor gerenciamento dos recursos de segurança pública; usando de estratégias como o posicionamento do corpo policial nos locais e horários mais críticos, baseado nos crimes registrados pelas câmeras. O que traz um ganho de segurança para a população.

3.1.3 Aplicações na Indústria

Pode-se explorar diversos setores da indústria utilizando sistemas IoT. Segunda Vermesan [46] os mais impactados são:

- Aviação
- Automotivo
- Telecomunicações
- Medicina
- Industrial
- Logística
- Óleo e Gás
- Transporte
- Segurança e privacidade
- Monitoramento de processos e ambientes
- Agricultura
- Entretenimento
- Construção civil
- Varejo

Dentro de todos esses setores da indústria, nos próximos tópicos serão enunciados alguns casos de sucesso e aplicações.

3.1.3.1 Aviação

Sistemas IoT podem ser aplicados à industrial de aviação coletando dados dos diversos sensores presentes nos aviões e canalizando esses dados para uma fonte única e consolidada. O que possibilitaria uma equipe acompanhar o status de cada parte de cada avião em tempo real. Assim, por meios de mecanismos e algoritmos de inteligência artificial, pode-se fazer a manutenção preditiva, ao invés da preventiva ou corretiva. A equipe de manutenção teria capacidade de saber exatamente qual parte do avião está ou estará com algum tipo de problema antes mesmo do avião pousar.

Correlacionando esses dados com os planos de voo de uma frota inteira, por exemplo, pode-se otimizar toda a cadeia de logística trazendo ganhos para as companhias aéreas e garantindo um nível de segurança nunca imaginado para os clientes.

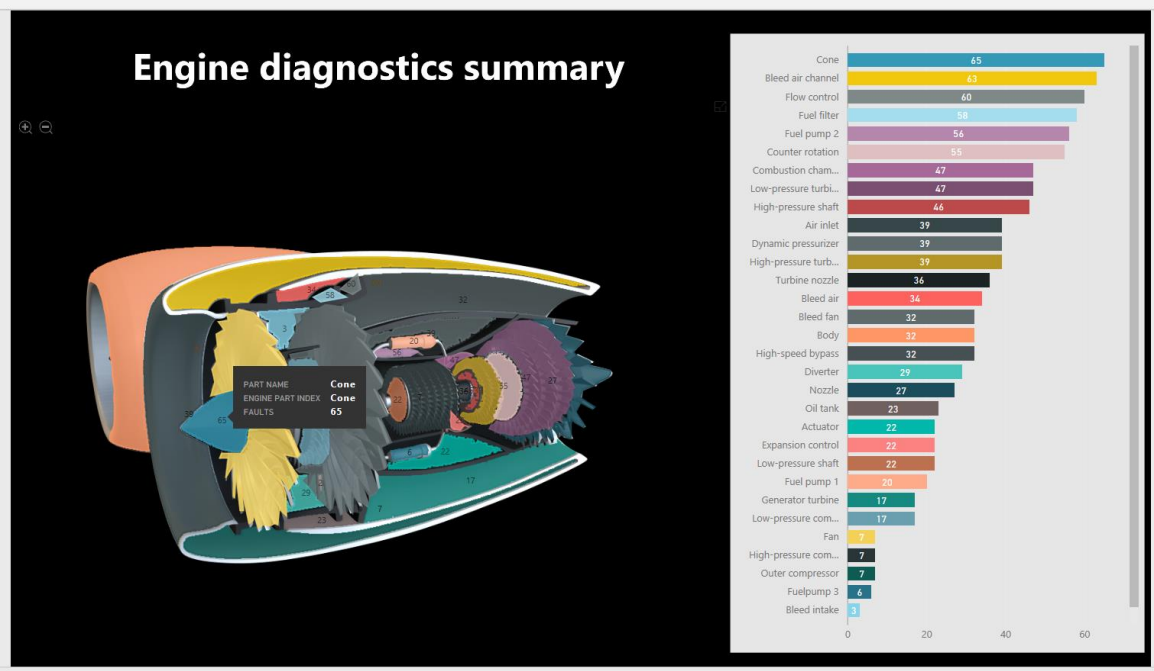

Figura 8 – Monitoramento Turbinas Frota Rolls- Royce [32]

Esse tipo de aplicação já é um caso de sucesso na frota de aviões que utilizam turbinas produzidas pelas Rolls-Royce [31]. Uma parte do sistema de supervisão desenvolvido [32] pode ser observado na Figura 8, onde pode-se observar o monitoramento de uma das turbinas. Nota-se que mais de 30 partes da turbina são monitoradas em tempo real com suas respectivas falhas, como por exemplo, em detalhe o Cone da Turbina apresentou 65 falhas, o que pode representar um alerta para a equipe de manutenção.

3.1.3.2 Automotivo

Sistemas IoT em automóveis podem ser aplicados para o monitoramento de todos os sinais vitais do automóvel, desde de pressão nos pneus à temperatura do motor ou nível do óleo. Sistemas como esse podem representar um ganho muito grande para empresas locadoras, por exemplo. Possibilitando essas empresas a direcionarem seus esforços reduzindo assim custos de operações.

Com o uso de IoT em automóveis também é possível otimizar os recursos utilizados pelo carro. Sabe-se que o desgaste do carro em geral varia com os hábitos do motorista e as condições de uso, sendo assim há veículos que irão precisar fazer manutenção mais precocemente que outros, mesmo com a mesma quilometragem percorrida. Com o monitoramento constante dos sinais do carro é possível saber o momento exato em que a troca de óleo do motor, por exemplo, é necessária.

A Figura 9, compara a manutenção tradicional com a manutenção preditiva [19]. Nota-se que ao longo da vida útil fez-se o dobro de intervenções para manutenção comparando as técnicas tradicionais e preditiva. Muitas vezes essa aplicação pode representar a redução de custos e a diminuição do tempo de inutilidade do automóvel, em que ele estaria parado para a manutenção.

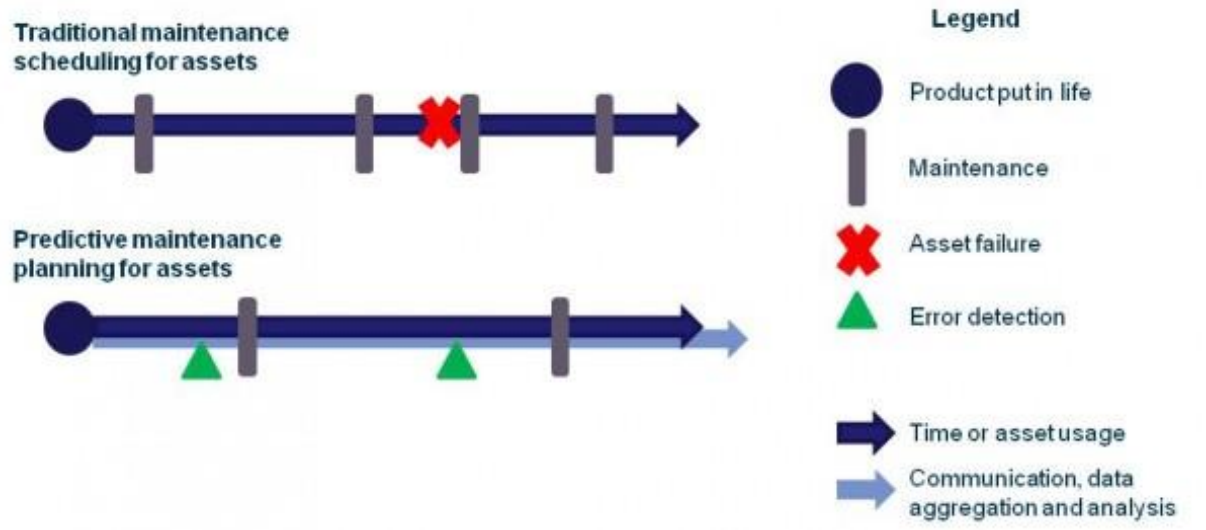

Figura 9 – Manutenção Tradicional Vs Preditiva [19]

3.1.3.3 Telecomunicações

Sistemas IoT permitirão a fusão de diferentes tecnologias de telecomunicações nas mais diversas aplicações. Como por exemplo o uso de Bluetooth, NFC (Near Field Communication), GPS, Internet e rede móvel aplicados em único sistema canalizador de dados.

Como aplicação: uso de NFC para objetos que estão próximos se comunicarem, capturando dados vindos de sensores com comunicação Bluetooth, e enviando esses dados junto com sua localização dada pelo GPS via uma rede móvel ou de Internet. Ou seja, um único sistema IoT interconectando várias tecnologias de Telecomunicação.

3.1.7 Medicina

Dentro do segmento médico, sistemas IoT podem ser empregados para fazer o monitoramento de sinais vitais como: temperatura do corpo, pressão sanguínea, frequência cardíaca, níveis de colesterol e glicose entre outras. Possibilitando assim que médicos ou familiares acompanhem em tempo real pessoas mais sensíveis, como idosos por exemplo. Com essa monitoração em tempo real é possível que decisões sejam tomadas mais rapidamente e proativamente, salvando assim muitas vidas.

Uma aplicação de um cenário como esse foi feita pela empresa Dartmouth-Hitchcock Health System que desenvolveu um sistema IoT chamado "ImagineCare" [29]. Esse sistema prove uma pulseira inteligente que coleta vários sinais vitais, e os envia para uma central onde vários profissionais da saúde monitoram seus clientes. Por essa pulseira o paciente também consegue emitir um sinal de socorro, onde em seguida uma equipe médica seria deslocada para o local exato do paciente. Soluções IoT como essa possibilitam um ganho na qualidade de vida muito grande.

3.2 Computação em Nuvem

Segundo o NSIT, Instituto Nacional de Padrões - órgão governamental estadunidense responsável pela padronização e metrologia - Computação em Nuvem é um modelo de compartilhamento de recursos computacionais com servidores, redes, aplicações, serviços, entre outros. [27]

O precursor desse modelo foi o Professor estadunidense John McCarthy, um famoso cientista da computação com trabalhos muito significativos na área de compartilhamento de recursos e Inteligência Artificial, que em 1957 fez alterações nos computadores IBM 704 e 7090 possibilitando que esses computadores compartilhassem recursos com vários usuários [26].

McCarthy esperava que algumas corporações iriam adotar o compartilhamento de recursos computacionais como modelo de negócio, vendendo recursos como armazenamento, processamento, aplicações, entre outros [18]. McCarthy não estava errado, segundo a consultoria Gartner o mercado de cloud computing movimentou em 2016 o total de $734 bilhões (dólares) e a previsão é que para 2020 esse mercado ainda cresça $216 bilhões, sendo responsável por movimentar quase $1 trilhão [14]. Atualmente grandes empresas como a Microsoft, Amazon Web Services, Google estão investindo fortemente nessa área para absolver essa demanda do mercado [16].

Nas últimas décadas grande parte do mercado e da população já estava em contato com modelos similares ao da computação em nuvem, como por exemplo:

- Computação Voluntária: baseado na ideia de trabalho voluntário, toda pessoa que queira contribuir com algum projeto de pesquisa pode instalar em seu computador um programa para se comunicar com um servidor central. Esse servidor vai usar os recursos da máquina do voluntário para realizar o processamento. Escalando essa ideia para milhões de voluntários, processos computacionais que demorariam anos para executar em um supercomputador podem ser finalizados em muito menos tempo. Um grande exemple de aplicação desse modelo é o sistema BOINC, que é responsável pelo processamento computacional de diversas pesquisas científicas [23].
- Compartilhamento de arquivos online: esses são os sistemas que possibilitam o armazenamento, acesso e compartilhamento de arquivos através da internet. Exemplos dessa tecnologia são: Dropbox, Onedrive, GoogleDrive, Flicker, entre outros.
- Web hosting: Serviço que permite que pessoas físicas ou jurídicas hospedem seus sites em servidores dos fornecedores. No presente trabalho, serão utilizados 2 serviços de Web Hosting: ThingSpeak, fornecida pela MathWorks e o Azure Mobile Service, fornecido pela Microsoft. Mas há ainda muito serviços consolidas dessa categoria, como por exemplo: 000webhost, GoDaddy, HostGator, Liquid Web, entre outros.
- Redes Sociais: São plataformas que apresentam uma variedade de sites e aplicações onde usuários com interesse em comum podem trocar mensagens, informações e compartilhar e acessar conteúdo de uma maneira extremamente eficiente. São exemplos: YouTube, Facebook, Linkedin, Wikipedia, etc.

O modelo de computação em nuvem é baseado em alguns pilares de características essenciais, modelo de implementação e modelo de serviço [27].

3.2.1 Pilar De Características Essenciais

NSIT definiu cinco grandes características essenciais:

Sob demanda e self-service: O usuário deve ser capaz de provisionar os recursos na nuvem conforme a sua necessidade de maneira automática sem a necessidade de intervenção humana por parte do fornecedor. O usuário deve ser capaz de começar e terminar o provisionamento por si só.

Acesso amplo pela rede: Os recursos devem ser disponibilizados através da rede, atendendo qualquer nível de heterogeneidade dos clientes, possibilitando o acesso através de qualquer plataforma (tablet, celular, laptops, desktops).

Agrupamento de Recursos e Multi-tenant: Os recursos computacionais do fornecer são agrupados, formando um grande conjunto de servidores, serviços e aplicações. Esses recursos são disponibilizados ao consumidor de forma que diversos clientes consigam acessar o grande conjunto de recursos computacionais simultaneamente, mas garantindo o isolamento entre os clientes. Essa maneira de compartilhamento de recursos físicos e virtuais de maneira compartilhada e segura é um modelo chamado Multi-tenant.

Quanto ao local físico onde estão os recursos computacionais, o consumidor padrão não controla nem conhece o local exato. Todavia ele pode designar a região ou datacenter em que deseja que o seu recurso seja alocado.

Elasticidade dinâmica: Os recursos podem ser provisionados e desprovisionados a qualquer momento de maneira ilimitada, normalmente de maneira automática. Essa elasticidade permite ao consumir gerenciar os seus recursos de acordo com a demanda, evitando gastos desnecessários.

Serviço monitorado: Os sistemas da nuvem monitoram de maneira inteligente e otimizada o uso de recursos. A qualquer momento pode-se gerar relatórios e acompanhar o consumo, o que torna o modelo transparente tanto para o fornecedor quanto para o consumidor.

3.2.2 Pilar Modelo de Implementação

Além do modelo tradicional, existem basicamente 3 modelos de implementação em nuvem conforme ilustrado na Figura 10: Infraestrutura como serviço (Iaas - Infrastructure as a Service), Plataforma como serviço (Paas - Platform as a Service), Software como Serviço (Saas - Software as a Service). [27]

A forma tradicional de se hospedar alguma aplicação é mantendo a infraestrutura localmente, sob total responsabilidade do cliente. É o cliente que vai ser o responsável por fazer todas as atualizações, configurações, customizações e implementações. Como pode ser observado na figura 10, é o cliente o responsável por gerenciar todas as camadas que uma aplicação requer, desde Rede ao Banco de Dados. Também é o cliente o responsável pela compra de hardware, software e o espaço físico para instalá-los. Ademais, nesse modelo o cliente tem que arcar com as despesas de luz, manutenção de ar condicionado, seguro contra desastres e roubos. De todos, esse é o modelo com maior gasto inicial e com menos elasticidade, pois caso o ambiente precise crescer, é necessário a compra, instalação e configuração, processo que pode demorar meses. [34]

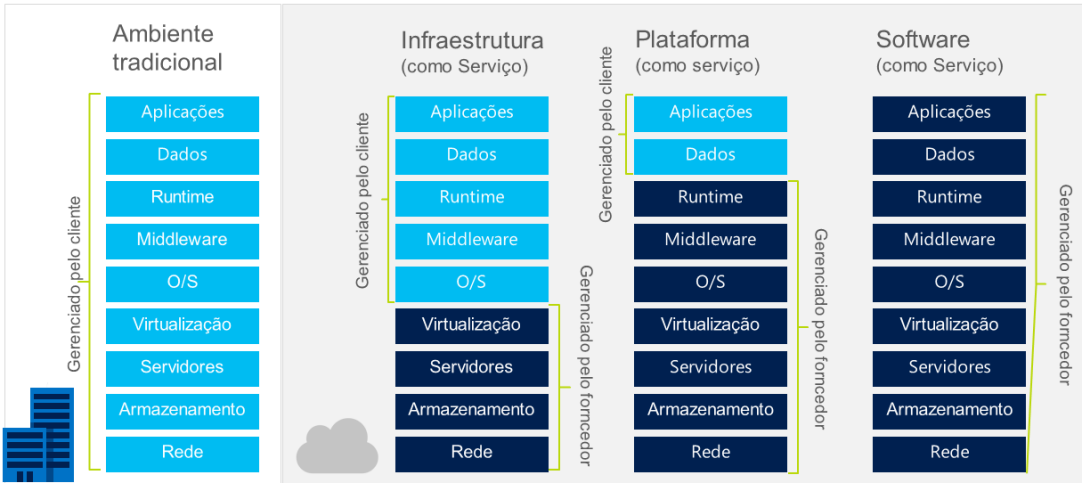

Figura 10 – Modelos de Implementação em Nuvem

Dentro da computação em nuvem a categoria base é a Infraestrutura com Serviço. No modelo IaaS o fornecedor disponibiliza recursos como CPU, memória, rede, máquinas virtuais, disco de armazenamento e rede. Nessa categoria o cliente é capaz de executar qualquer software desejado, incluindo o sistema operacional, sendo responsável pelas customizações e implementações e a gestão de dados e aplicações. Todavia o cliente agora não precisa mais gerenciar as atualizações, configurações e Virtualização, nem mesmo comprar ou manter hardware ou outros gastos de infraestrutura física. Um dos grandes fornecedores de Iaas do mercado são: Microsoft Azure, Amazon, Rackspace e Gogrid. [18]

Na camada intermediária, tem-se a categoria PaaS. Nessa categoria oferece-se ambientes de desenvolvimento e teste de maneira rápida, possibilitando assim os desenvolvedores criem sites, aplicativos, softwares, sem se preocupar com recursos de infraestrutura como rede e servidores. O PaaS veio para tornar o ciclo de desenvolvimento mais

curto, eficaz e dinâmico, uma vez que em curtíssimo tempo o desenvolvedor consegue provisionar uma infraestrutura pronta e funcional para progredir com o seu trabalho sem ter que se preocupar com nenhum aspecto de infraestrutura, sistema operacional ou programas de intermédio entre a aplicação e o sistema operacional (Middleware). Utilizando recursos PaaS o desenvolvedor consegue focar seus esforços em somente na aplicação propriamente dita e nos dados que ela gera. Os dois principais fornecedores são o Google Engine e a Microsoft Azure, que fornece plataformas de banco de dados, web hosting, serviços de fila, entre outros [34].

No topo está a categoria SaaS, que proporciona o usuário a usufruir de aplicações através da internet, normalmente, sob o modelo baseado em subscrição. Aplicações com recursos Saas já são presentes no dia-a-dia de todos através do advento das redes sociais, dos mais diversos aplicativos e serviços de streaming que utilizam recursos da nuvem. Os clientes das redes sociais por exemplo, não gerenciam a manutenção do disco de armazenamento onde as suas fotos, mensagens e vídeos estão armazenadas. Não são responsáveis também pela atualização dos softwares que processam as informações. O mesmo acontece para serviços de Streaming como o Netflix. No modelo SaaS o cliente simplesmente utiliza o software via internet. A responsabilidade de gerenciamento de todo o sistema fica sob responsabilidade do fornecedor do serviço [34].

No ponto de vista de funcionamento o fornecedor provisiona a aplicação na nuvem, permite o acesso dos usuários à aplicação e aos dados, dispensando assim a instalação da aplicação localmente no computador do cliente, que traz a vantagem de poder manter a aplicação sempre atualizada sem a necessidade de ação do usuário, pois a aplicação está centralizada em um único serviço, e não mais instalada no computador do usuário.

Exemplos de provedores de SaaS para consumidor final são: Facebook, Uber, Google Maps, Google Tradutor, Whatsapp, Netflix, Spotify, entre outros.

Para melhor entender a diferença entre os modelos de implementação vale uma analogia muito simples com as maneiras de se comer uma Pizza [47] conforme ilustra a Figura 11.

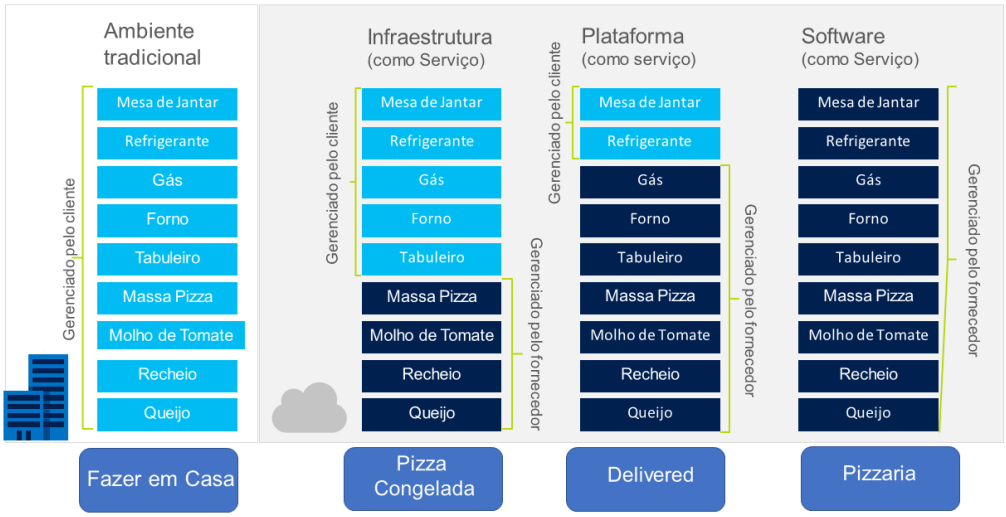

Figura 11 – Analogia Modelos de Implementação em Nuvem

A maneira mais tradicional de se comer uma pizza é faze-la em casa, o cozinheiro tem que dispor de tudo que é necessário para fazer uma pizza, dos ingredientes, recursos enérgicos, bebida e eletrodomésticos. O que é exatamente análogo ao desenvolvimento de aplicações em Ambiente tradicional. Todo o gerenciamento é feito pelo cozinheiro/cliente.

Outra maneira de se comer uma pizza, seria comprá-la congelada e assar em casa. Nessa maneira o cozinheiro precisa somente dos recursos enérgicos, bebida e eletrodomésticos. O que equivale à Infraestrutura como serviço.

A terceira maneira de comer uma pizza seria fazendo uma ligação para uma pizzaria e pedindo para que entregasse em casa. Dessa forma o cozinheiro só precisaria se dispor de uma mesa e a bebida, do o resto dos itens necessários ficam por conta da pizzaria responsável pela entrega, analogamente à Plataforma como Serviço.

A última maneira, análoga ao Software como serviço, seria comer a pizza em algum restaurante ou pizzaria. Assim como no SaaS o cliente não gerencia nenhum recurso e sua única tarefa é usufruir do serviço prestado.

Como um resumo, pode-se recorrer à Figura 12 disponível em [35], onde é possível entender o que cada modelo de implementação fornece.

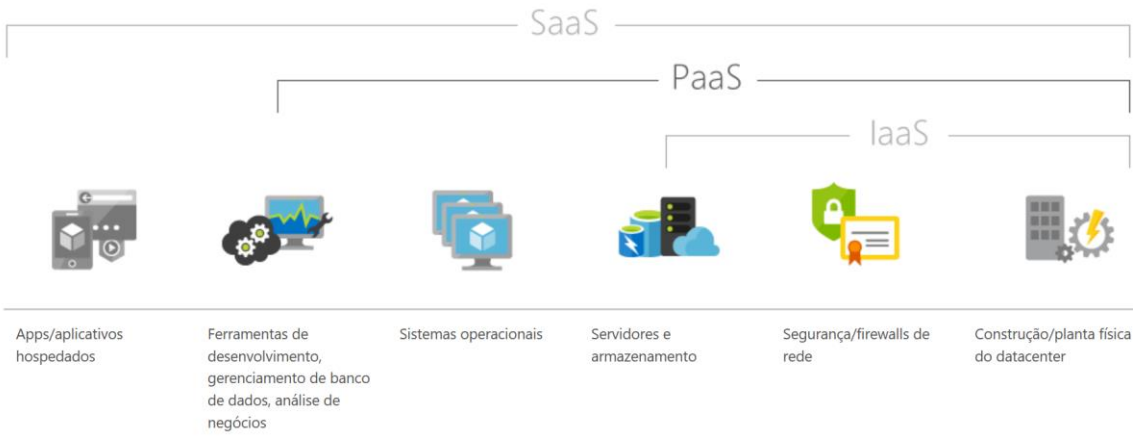

Figura 12 – Resumo Modelos de Implementação e Exemplos [35]

O modelo IaaS é capaz de prover recursos de servidores, armazenamento, segurança da rede, além dos recursos de infraestrutura física. O modelo PaaS contempla o modelo IaaS com o acrescimento de recurso de sistema operacionais e ferramenta de desenvolvimento, gerenciamento e análise. Finalmente o modelo SaaS, o mais completo, consegue toda a gama de recursos necessários para o funcionamento de uma aplicação.

3.2.3 Pilar Modelo de Serviço

O NSIT definiu quatro tipos de modelo de Serviço, sendo eles: Nuvem privada, Nuvem comunitária, Nuvem pública e Nuvem Híbrida. Um esquema funcional desses modelos pode ser visto na Figura 13, onde observa-se os quatros tipos de modelos e que a Nuvem Híbrida é uma combinação da nuvem Pública e Privada. Detalhes serão discutidos a seguir.

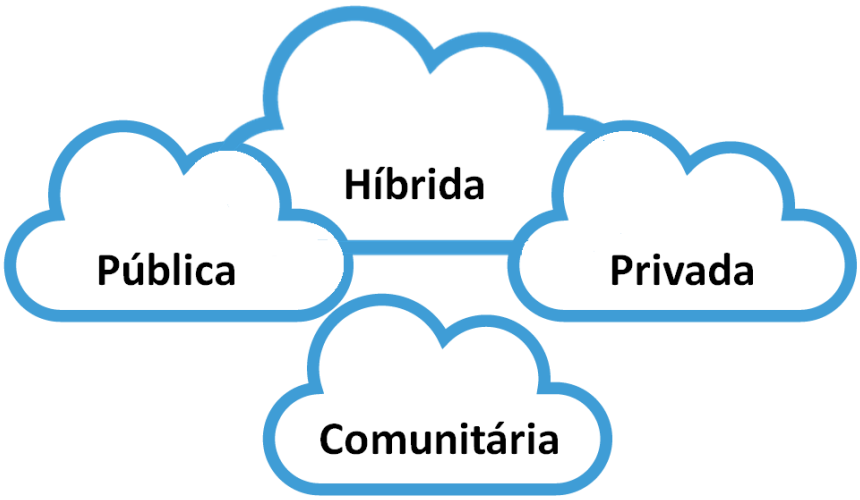

Figura 13 – Modelos de Serviços

O conceito de nuvem privada é atribuído quando a infraestrutura da nuvem é provisionada e dedicada a um único cliente. Essa infraestrutura pode pertencer, ser monitorada e operada pelo próprio cliente ou fornecedores terceiros [27]. Todavia só deve ser utilizada por um único cliente. Dentre todos os modelos esse é aquele em que o cliente tem o nível de isolamento de recursos maior.

Quando a infraestrutura requerida pela nuvem é dedicada à um grupo de clientes, uma comunidade onde todos têm ciência com quem está compartilhando o recurso tem-se a nuvem comunitária. Esse modelo deve pertencer, ser monitorado e operado pela própria comunidade ou fornecedores terceiros [27]. Esse modelo é adotado por muitas corporações formada por diversas empresas que compartilham os mesmos requisitos de segurança e políticas internas.

A definição de nuvem pública é dada quando a infraestrutura provisionada é utilizada pelo público em geral. A nuvem pública deve pertencer, ser operada e gerenciada por uma empresa, instituição de ensino ou governamental [27]. Nesse modelo o cliente não tem controle de quais outros clientes estão acessando o mesmo datacenter, todavia a segurança dos dados é totalmente garantida.

Nuvem híbrida é quando o cliente utiliza qualquer combinação dos modelos de Nuvem privada, comunitária e pública [27]. Normalmente é usado por clientes que tem políticas internas que permitem que somente uma parte dos seus dados ou aplicações sejam expostos para a internet.

O modelo de computação em nuvem apresenta vantagens e desvantagens que serão discutidas a seguir:

3.2.4 Vantagens

A nuvem consegue fornecer tanto para o cliente quando para o provedor em termos técnicos e econômicos.

3.2.4.1 Benefícios sob a ótica do fornecedor:

- Melhor gestão de atualização: Serviços SaaS possibilitam o fornecedor a manter a aplicação sempre atualizada e configurada conforme as melhores práticas [18], pois o serviço está sob o seu total controle apartado da infraestrutura do cliente. Essa vantagem minimiza a chance de erros de aplicação, o que garante a satisfação do cliente.

- Melhor aproveitamento de Hardware: Em modelos como nuvem pública, o fornecedor pode manter a sua infraestrutura funcionando em capacidade máxima [18]. Por exemplo, se o fornecedor tem um cliente utilizando um servidor a 20% de sua capacidade, ele pode vender os 80% restantes para diversos outros clientes. Gestão que garante a otimização de recursos e, consequentemente, lucro.
- Modelo de negócio promissor: Os fornecedores de softwares podem fornecer suas soluções no modelo SaaS, utilizando o licenciamento baseado em uma subscrição mensal ou anual. Esse modelo de licenciamento torna a solução mais atrativa, pois o gasto inicial é muito mais baixo quando comparado aos licenciamentos tradicionais. Isso sem contar com a minimização dos riscos de pirataria, pois agora o cliente não tem mais acesso aos binários dos produtos.
- Monitoramento de hábito de consumo do cliente: devido ao Pilar Essencial de monitoramento supracitado, o consumo do cliente é totalmente transparente para o fornecedor, o que permite o time de estratégia do fornecedor a sugerir produtos e serviços que se encaixam no padrão de consumo do cliente [18], possibilitando assim uma pré-venda mais assertiva.

3.2.4.2 Benefícios sob a Ótica do Consumidor

- Redução de Custos: o modelo de cobrança "pay-as-you-go" (pague conforme o uso), aplicado à computação em nuvem permite que o cliente consiga fazer uma melhor gestão dos seus recursos, como por exemplo, o desligamento dos recursos fora do horário comercial. Isso sem contar com a inibição dos gastos iniciais em hardware e licenciamento, e de infraestrutura física. Devido a esse modelo, a criação de startups ficou muito mais fácil, pois com a nuvem os recursos computacionais não são exclusividade de grandes empresas, com o advento da nuvem houve a democratização de recursos.
- Redução do tempo de configuração: Tradicionalmente disponibilizar recursos computacionais é penoso e demorado para as equipes técnicas do cliente, sendo necessário fazer a compra dos recursos, instalação, configuração e planejamento. Ao contrário, com a nuvem, respeito o Pilar Essencial "Sob demanda e self-service", o cliente com poucos cliques pode provisionar recursos como um servidor, ou banco de

dados em pouquíssimos minutos. Característica que traz agilidade e elasticidade para o negócio.

- Terceirização da gestão: Usufruindo da nuvem, o cliente consegue terceirizar a gerencia em geral diferentes níveis. No modelo IaaS, o cliente terceiriza a gerencia dos recursos de infraestrutura física, rede, virtualização e armazenamento. Todavia ainda é responsável em gerenciar o Sistema operacional, middleware, runtime, dados e a aplicação. Já no modelo PaaS, fica incluído na terceirização de gerencia as camadas de sistema operacional, middleware e runtime. Enquanto que no modelo SaaS todos a gerencia de todos os recursos ficam sob responsabilidade do fornecedor.
- Escalabilidade: Com a computação em nuvem, os clientes podem requisitar e provisionar quantas instancias de hardware ou software forem necessárias conforme a necessidade. Da mesma maneira o cliente também pode fazer o desprovisionamento quando o recurso está ocioso, evitando assim gastos desnecessários. Com uma gestão automática via código é possível manter a infraestrutura sempre trabalhando em capacidade total, otimizada. Cenário muito mais atraente do que o tradicional, em que muitas vezes os recursos são subutilizados.
- Segurança: Para atender as mais diversas políticas e exigências dos clientes, os fornecedores investem muito em segurança. É comum que os datacenters dos fornecedores possuam muitos certificados de segurança de diversos confiáveis e renomadas instituições internacionais, o que garante a privacidade e segurança dos clientes. O data center do Azure (Microsoft) possui diversas certificações de segurança a níveis globais e por segmento de indústrias para atender os mais exigentes mercados. Algumas delas podem ser conferidas na Figura 14 [33].

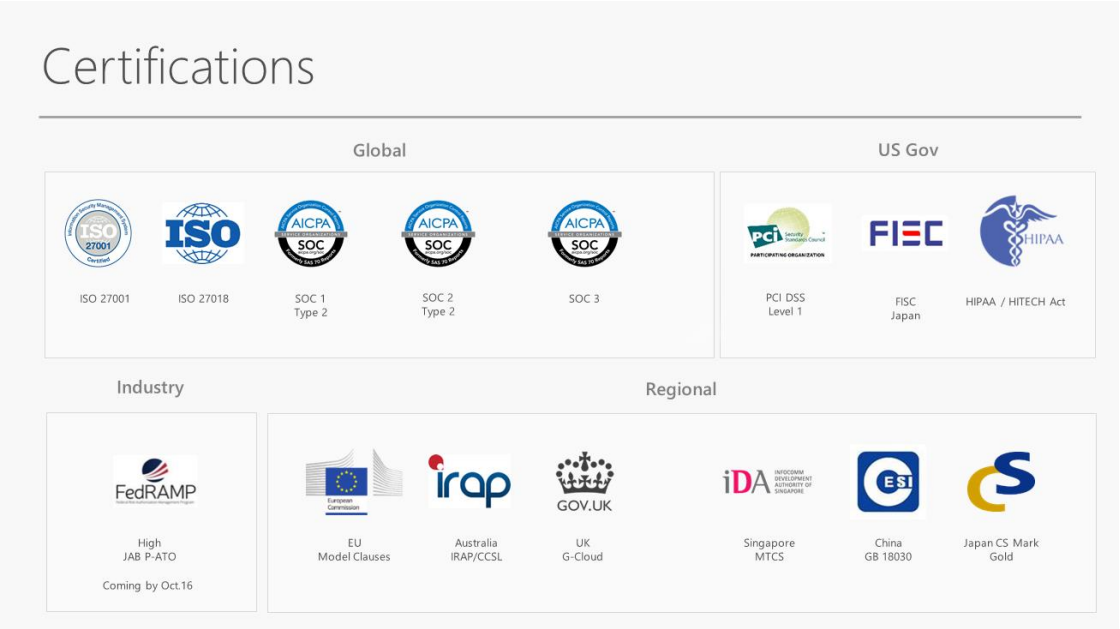

Figura 14 – Certificações de Segurança Datacenter Azure [33]

- Disponibilidade: Com o advento da nuvem, recursos como replicação de ambiente para a recuperação em caso de desastres não são exclusividade das grandes empresas. Fornecedores, como o Azure (Microsoft), replicam os dados três vezes dentro do mesmo data center, fornecendo a opção de também duplicar mais três vezes geograficamente (em outro datacenter muito distante). Esses recursos garantem que mesmo em caso de um desastre muito grande como alagamento, meteoro ou pane elétrica no datacenter do fornecedor as aplicações do cliente continuem funcionando e disponíveis [24].

3.2.5 Desvantagens

A computação em nuvem ainda é uma tecnologia muito recente que apresenta alguns problemas, segundo Quasay em [18] algumas desvantagens são:

- Falta de padronização: Cada fornecedor disponibiliza para os clientes o serviço em um padrão diferente, o que dificulta a migração dos clientes entre os fornecedores, o que pode ser um risco muito grande, pois o cliente fica dependente do fornecedor, o que pode ser uma enorme desvantagem econômica e estratégica.

- Dependência da Internet: É fato que para usar recursos de computação em nuvem, a internet é uma premissa. Todavia em muitas localidades isoladas em países de terceiro mundo, principalmente, o acesso à internet é precário. Fato esse que impossibilita, ou torna muito difícil a utilização dessa tecnologia.

3.2.6 Serviços de computação em nuvem empregados

No presente trabalho, utilizou-se diversos serviços de computação em nuvem. Entre eles tem-se o Banco de Dados SQL Azure Database, o serviço de hospedagem de sites Web App Azure, a plataforma para IoT ThingSpeak, a plataforma de relatórios Power BI e a Rede Social Twitter. Cada um desses serviços serão abordagem com detalhes nos próximos tópicos.

3.2.6.1 SQL Azure Database

SQL Azure Database é um serviço de Banco de Dados relacional na nuvem provido pela Microsoft, é baseado na Engine SQL Server. Dentro da classificação de Modelos de Implementação o SQL Azure Database é um serviço PaaS, onde abstrai-se boa parte do gerenciamento. [28]

Por ser um serviço de nuvem, o SQL Azure Database é escalável automaticamente, se ajustando a carga de trabalho com desempenho otimizado. Por ser baseado na Engine SQL Server, que é altamente consolidada no mercado o SQL Azure Database suporta as mesmas ferramentas, extensões, bibliotecas e API disponíveis para o SQL Server tradicional [28].

Todas essas vantagens possibilitando ao desenvolvedor fazer o seu trabalho de maneira muito rápida, pois ele somente gerencia camada de Dados e Aplicação propriamente dita, que ele está desenvolvendo. As outras camadas são gerenciadas e mantidas pelo provedor, que no caso é a Microsoft.

O Software utilizado para manipular o banco de dados SQL Azure é o Microsoft SQL Server Management Studio, onde é possível tomar várias ações e fazer configurações de acordo com a demanda do desenvolvedor, como por exemplo: fazer consultas, criar tabelas, administrar usuários, excluir ou adicionar linhas, entre outras ações/configurações. A linguagem de programação utilizada no Microsoft SQL Server Management Studio é a Transact-SQL, já consolidada e amplamente utilizada pelo mercado. [28]

Maiores detalhes de como provisionar esse serviço, e como manipulá-lo com o Microsoft SQL Server Management Studio serão discutidos na Metodologia deste trabalho.

3.2.6.2 Web App Azure

Azure App Service é um serviço de nuvem de hosteamento de aplicações classificado como PaaS [17]. Ele consegue prover tudo que é necessário para desenvolver um aplicativo usando qualquer plataforma, por exemplo:

- Permite armazenar dados em diversos banco de dados de mercado como o MongoDB, BlobStorage ou SQL.
- Permite que o desenvolvedor implemente autenticações baseadas em serviços do Facebook, Google, Microsoft, Twitter.
- Suporta serviços de notificação para milhões de usuários simultaneamente.
- Disponibiliza máquinas virtuais para suportar a aplicação desenvolvida globalmente, de maneira escalável e confiável.

Além de todas essas vantagens, o Azure App Service também provê uma maneira simples, rápida e fácil de se armazenar dados. Fazendo a interconexão da aplicação com um banco de dados Azure SQL. Essa funcionalidade é chamada de “Easy Table”, e detalhes de implementação serão discutidos na sessão de Metodologia deste trabalho.

3.2.6.3 Plataforma de IoT ThingSpeak

ThingSpeak é uma plataforma em nuvem provida pela MathWorks. Essa plataforma é dedicada para serviços IoT e dentro da classificação de implementação, o ThingSpeak é um serviço de nuvem SaaS, onde o desenvolvedor apenas utiliza os serviços, sem se preocupar com nenhum tipo de gerenciamento [25].

Essa plataforma prove uma funcionalidade que permite enviar dados pela Internet, esses dados são recebidos e mostrados em forma de gráficos. Ademais o serviço é gratuito, com o limite de oito mil mensagens diárias e até 100 aplicações diferentes enviando dados. A frequência máxima suportada pelo ThingSpeak é de o enviado de dados de 15 segundos. Também só se é possível exibir até oito gráficos simultaneamente. [25]

Outra funcionalidade é a configuração de notificações do usuário integrando com outros serviços de nuvem, como mídias sociais por exemplo.

Maiores de detalhes de configuração da plataforma serão abordados na sessão de Metodologia.

3.2.6.4 Power BI

Power BI é um conjunto de ferramentas analíticas providas pela Microsoft. Com esse conjunto de ferramentas é possível centralizar, agregar e manipular dados de diferentes origens. Depois de manipular os dados é possível fazer diversos gráficos dinâmicos que juntos constituem um relatório. Depois do relatório pronto, é possível compartilha-lo com todos os interessados, o disponibilizando em qualquer plataforma Web ou móvel [36].

O Power BI é capaz de se conectar a mais de 70 fontes de dados, sendo as principais: Excel, Text/CSV, XML, SQL Server Database, Access Database, SQL Server Analysis Services Database, MySQL Database, SAP HANA Database, SAP Business Warehouse server, Azure SQL Database, Azure SQL Data Warehouse, Azure Analysis, Azure Blob Storage, Azure Table Storage, Azure Data Lake Store, Azure HDInsight, Google Analytics, Facebook, GitHub, Web, SharePoint List [21].

Para fazer a manipulação dos dados, utiliza-se o software Power BI Desktop, juntamente com a linguagem DAX (Data Analysis Expressions) [30].

A Figura 15 resume a proposta do Power BI: coletar dados de diversas fontes, trabalha-los de diversas formas para gerar conhecimento, e juntamente com a experiência possibilitar que pessoas, aplicações e processos tomem ações embasadas e seguras. Onde todo o processamento é feito utilizando recursos computacionais na nuvem que possibilitam o desenvolvimento de rápido de um relatório de qualidade e de alta disponibilidade, assim como um serviço da nuvem deve ser.

3.2.6.5 Twitter

O Twitter é uma rede Social O Twitter é uma rede social criada em 2006 pelos americanos Jack Dorsey, Noah Glass, Biz Stone e Evan Willian. Essa rede social é um serviço de nuvem enquadrado no modelo SaaS, onde usuários desse serviço podem interagir trocando mensagens curtas de até 140 caracteres, a essas mensagens dão-se o nome de "Tweets". Atualmente essa rede social é utilizada por centenas de milhões de pessoas que utilizam desse serviço para compartilhar notícias e opiniões, sendo também utilizada por empresas para se fazer campanhas publicitárias. [43]

Figura 15 – Proposta Power BI [36]

Um grande diferencial dessa rede Social quando comparada com as demais, é que ela é aberta para os desenvolvedores de aplicações. Sendo assim é possível desenvolver sistemas que também interajam com o Twitter. E aproveitando dessa capacidade, pode-se desenvolver sistemas de notificação através do Twitter, sendo assim o usuário pode receber alertas em seu dispositivo móvel em qualquer lugar e a qualquer momento. [44]

3.3 ESP8266-01

Conforme [41] o Esp8266 é um microcontrolador produzido pela Espressif, uma empresa Chinesa sediada em Shangay. Esse módulo é uma solução barata e eficaz para a aplicação cenários de diversos cenários de Internet of Things (IoT), pois ele possui um sistema próprio e completo de conexão com a Internet via Wifi. Apesar do ESP8266 possuir um microcontrolador embutido, segundo [10] o módulo ESP8266 também a capaz de se comunicar com outros microcontroladores, empoderando assim uma gama de microcontroladores como o Arduino, por exemplo, a se comunicar com a Internet sem fio. O caso de aplicação no presente trabalho será conectar o módulo no Arduino para que funcione como ponte Serial-Wifi, possibilitando que dados conectados pelo Arduino sejam enviados para a Internet.

Essa capacidade de integração ESP8266/Arduino é possível devido ao suporte do ESP8266 à comunicação via protocolo SPI (Serial Peripheral Interface) o que expande a capacidade de aplicações dos microcontroladores (até então sem acesso à internet) para cenários de IOT aplicações à Automação Industrial ou residencial, monitoramento, instrumentação e redes mesh, por exemplo [10].

Como consta em [10], o módulo ESP8266 apresenta 3 modos de gestão de Energia: Modo Ativo, Modo Sleep e Modo Hibernação. Essas propriedades permitem a redução de gastos energéticos em mais de 90%, viabilizando uma eficiente gestão de energia. Como pode ser verificado em detalhes na Figura 16, disponível em [10] o consumo para envio e recebimento de dados chega em operação normal a cerca de 220mA, o consumo total de até 300 mA [10] enquanto em modo de Hibernação reduz os gastos para 20uA.

Parameters	Min	Typical	Max	Unit
Tx802.11b, CCK 11Mbps, P OUT=+17dBm	-	170	-	mA
Tx 802.11g, OFDM 54Mbps, P OUT =+15dBm	-	140	-	mA
Tx 802.11n, MCS7, P OUT =+13dBm	-	120	-	mA
Rx 802.11b, 1024 bytes packet length , -80dBm	-	50	-	mA
Rx 802.11g, 1024 bytes packet length, -70dBm	-	56	-	mA
Rx 802.11n, 1024 bytes packet length, -65dBm	-	56	-	mA
Modem-sleep①	-	15	-	mA
Light-sleep②	-	0.9	-	mA
Deep-sleep③	-	20	-	μA
Power Off	-	0.5	-	μA

Figura 16 – Consumo do ESP8266-01 [10]

Segundo [41], outra propriedade do ESP8266-01 é a possibilidade de atuar como cliente, possibilitando o módulo a se conectar aos mais diversos serviços da Internet e fazer requisições. Há também um modo de operação como servidor, o que permite ao módulo fornecer funções e serviços requisitados por outros usuários da rede. Existe ainda a possibilidade do módulo em atuar em módulo híbrido, funcionando simultaneamente como servidor e cliente.

Atualmente são produzidos mais de 14 modelos para atender as mais diversas aplicações. Entre todos esses modelos, um dos mais consolidados é o ESP8266-01 [41] que tem como principais especificações:

- Possui 2 pinos GPIO (General-purpose input/output)
- Barramentos I²C,SPI,UART
- CPU com operação na velocidade 80MHz, mas podendo atingir até 160MHz
- 32Kbytes de RAM para instruções
- 96Kbytes de RAM para dados

- 64Kbytes de ROM para boot
- Memória flash SPI de 512KBytes
- Arquitetura RISC de 32Bits

3.3.1 Hardware

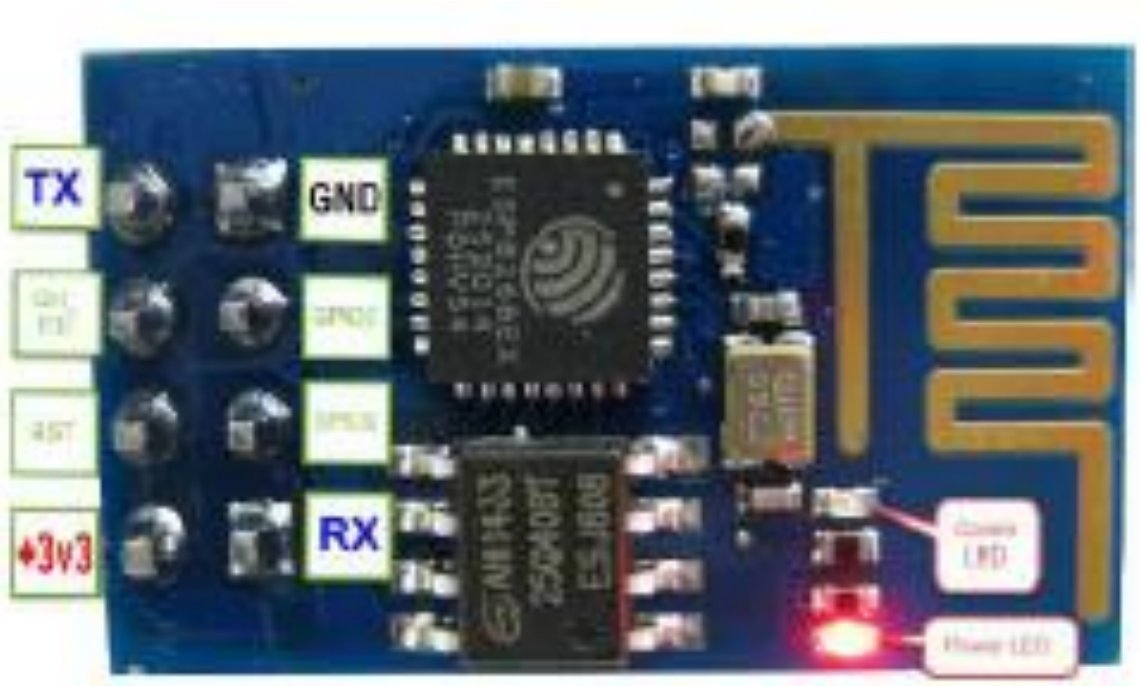

Figura 17 – Pinagem do ESP8266-01 [10]

O modulo Esp8266-01 apresenta um total de 8 pinos, sendo 2 utilizados para a alimentação, 2 como GPIO, 2 para comunicação serial e os demais para controle e atualização de firmware. A Figura 17 mostra a disposição dos referidos pinos, e na Tabela 1, disponível em [10], contém uma breve descrição funcional de cada pino:

Tabela 1 – Pinagem ESP9266-01

Tx	Pino usado para a comunicação serial, responsável pela transmissão do sinal
Rx	Pino usado para a comunicação serial, responsável pela recepção do sinal
Vcc	Pino de alimentação do módulo
GND	Pino ground (terra) do modulo
CH_PD	Pino de controle do módulo, utilizado para atualização ou gravação de firmware. Para operação normal deve ser mantido em nível alto
RST	Pino de Controle utilizado para o hard reset do modulo. É acionado em nível baixo

GPIO 0	Pino de aplicação geral que pode ser controlado pelo firmware
GPIO 2	Pino de aplicação geral que pode ser controlado pelo firmware

Vale ressaltar que, conforme a Tabela 1, para assegurar o funcionamento do módulo em operação normal é requisito que os Pinos RST e CH_PD sejam mantidos em nível alto.

O esquema elétrico recomendado [41] para fazer a programação do módulo é apresentado na Figura 18. O circuito montado nesse trabalho foi baseado nessa referência, respeitando os requisitos da Tabela 1, mas fazendo algumas alterações, como por exemplo substituindo o Conversor de nível lógico FTDI pelo Arduino Uno R3. Mais detalhes sobre o circuito utilizado nesse trabalho serão discutidos na seção Metodologia.

A tensão de operação nominal do módulo é de 3.3V, podendo variar em até 0.3V, conforme a Figura 19 disponível em [10]. Como já mencionado, em plena operação o ESP8266-01 pode consumir até 300 mA. No caso de aplicação do presente trabalho, ponte Serial-Wifi entre o Arduino e o módulo, faz-se necessário utilizar uma fonte dedica, pois se consegue drenar do Arduino somente 50 mA, valor esse muito abaixo do requerido pelo módulo. Não respeitar essa limitação do Arduino pode ocasionar em danos permanentes do mesmo.

3.3.2 Comandos AT

Segundo [4] no início da década de 80, o Americano Denis Hayes desenvolveu o modem Smartmodem 300, dispositivo que possibilita a comunicação entre computadores e a rede telefônica, fazendo a ponte entre sinais digitais, dos computadores, e sinais analógicos, da rede telefônica. Todavia, para viabilizar a programação do Smartmodem 300, Hayes também inventou uma linguagem de comando chamada Hayes command set. Essa linguagem tinha a característica de todos os comandos começarem com os caracteres "AT", abreviação de Attention. Então com o passar do tempo essa linguagem se popularizou como Hayes AT command set, ou simplesmente AT command. Graças a sua simplicidade e baixo de custo de implementação, outros fabricantes de modem aderiram aos comandos AT e, aos poucos, novas funções foram implementadas, se tornando uma linguagem consolidada. Ainda hoje, esses comandos são utilizados em sistemas de voz, SMS e transmissão de dados.

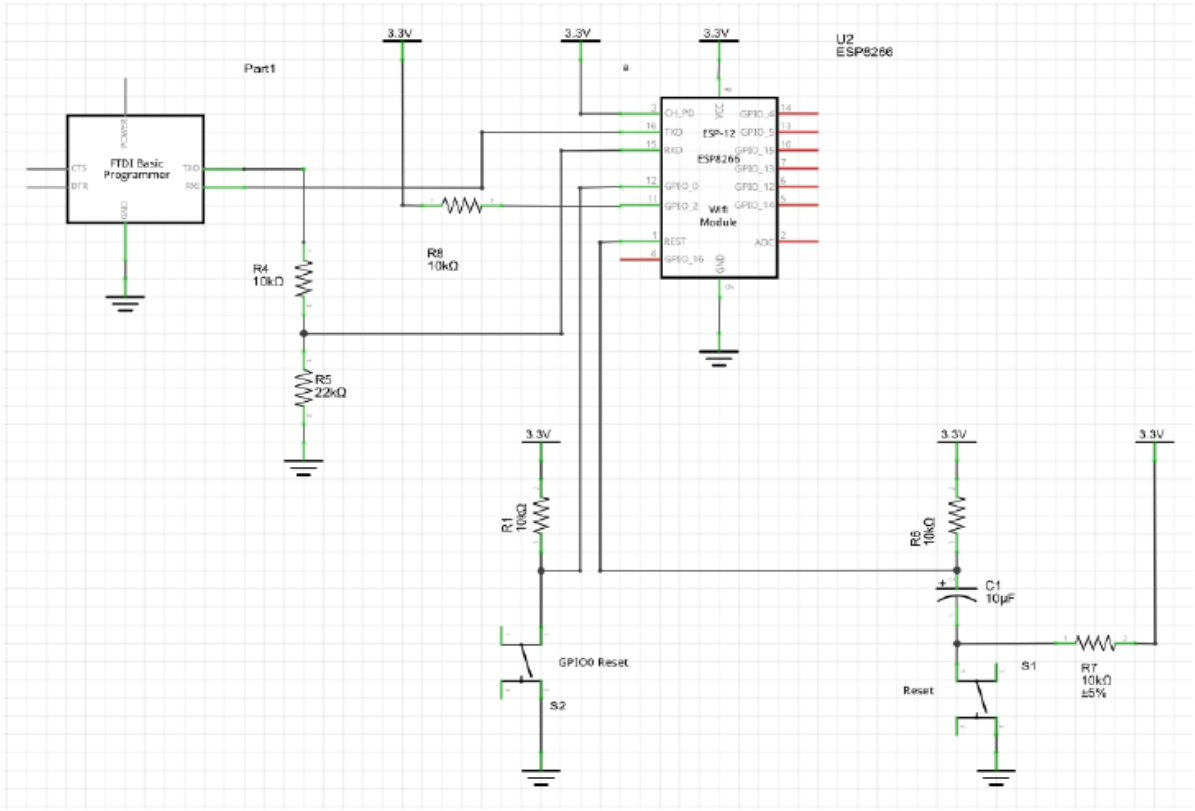

Figura 18 - Esquema Elétrico Sugerido [41]

Parameters		Conditions	Min	Typical	Max	Unit
Storage Temperature Range		-	-40	Normal	125	°C
Maximum Soldering Temperature		IPC/JEDEC J-STD-020	-	-	260	°C
Working Voltage Value		-	3.0	3.3	3.6	V
I/O	V_{IL}/V_{IH}	-	-0.3/0.75V_{IO}	-	0.25V_{IO}/3.6	V
	V_{OL}/V_{OH}	-	N/0.8V_{IO}	-	0.1V_{IO}/N	
	I_{MAX}	-	-	-	12	mA
Electrostatic Discharge (HBM)		TAMB=25°C	-	-	2	KV
Electrostatic Discharge (CDM)		TAMB=25°C	-	-	0.5	KV

Figura 19 – Especificações Elétricas do Esp8266-01

Para o módulo ESP8266 não é diferente, os comandos AT são suportados e utilizados para a configuração de parâmetros, consulta de status e execução de tarefas do módulo. Através dos comandos AT pode-se configurar e executar ações como por exemplo: conectar ou desconectar ao Wifi, listar as redes disponíveis, mudar a velocidade de conexão, enviar dados, abrir e fechar comunicação, entre outras. Possibilitando assim a configuração do modulo para atender aos mais diversos projetos [39].

Os comandos AT suportados podem ser divididos em três grandes grupos: Básicos, Camada de Wifi e Camada TCP-IP. Os comandos básicos são aqueles utilizados para verificar o funcionamento do módulo, reiniciar, verificar a versão do firmware instalada, colocar o módulo em modo de hibernação, entre outras.

Os comandos para a camada de Wifi são suportados os comandos para se conectar ou desconectar a uma rede especifica, verificar todas as redes disponíveis, alterar entre os modos de operação de host e/ou client, entre outros. Para o terceiro e último grupo, Comandos para a Camada TCP-IP, são suportados comandos para abrir e fechar conexão com servidores, verificar o status da conexão, enviar dados, e até mesmo estabelecer múltiplas conexões. Os detalhes dos comandos podem ser verificados na Tabela 2 [39]

Tabela 2 – Comandos AT [39]

Tipo	Comando	Resposta Esperada	Descricão	Grupo
Execução	AT	OK	Faz a verificação se a comunicação entre o Microcontrolador e o módulo está funcionando corretamente	Básico
Execução	AT+RST	OK	Faz o reset do módulo	Básico
Execução	AT+GRM	<versão do firmware>, OK	Retorna a versão do firmware instalado no módulo	Básico
Configuração	AT+GSLP=<tempo em ms>	<tempo em ms>, OK	Configura o módulo para entrar em modo de Hibernação pelo tempo informado	Básico
Execução	ATE0	OK	Desabilita as repostas do módulo	Básico
Execução	ATE1	OK	Habilita as respostas do módulo	Básico
Teste	AT+CWMODE=?	+CWMODE: <mode> , OK	Lista os modos de operação disponíveis. Retorna um valor inteiro entre 1 e 3, inclusive. Onde: • Modo 1: Cliente • Modo 2: Servidor • Modo 3: Cliente e Servidor	Camada Wifi
Consulta	AT+CWMODE?	+CWMODE: <mode>, OK	Informa em qual modo de operação (cliente, servidor ou ambos) o módulo está configurado	Camada Wifi
Execução	AT+CWMODE=<mode>	OK	Configura o módulo para operar no modo desejado	Camada Wifi
Consulta	AT+CWJAP?	+CWJAP:<SSID>, OK	Informa em qual rede (SSID) o módulo está conectado	Camada Wifi
Execução	AT+CWJAP=<ssid>, <pwd>	OK	Conecta o módulo na rede <SSID> utilizando a senha <pwd>	
Execução	AT+CWQAP	OK	Desconecta o módulo da rede conectada	Camada Wifi
Consulta	AT+CIPSTA?	+CIPSTA: <ip> , OK	Informa o endereço IP do módulo	Camada Wifi

Tipo	Comando	Resposta Esperada	Descrição	Grupo
Consulta	AT+CPISTATUS=?	OK	Informa o Status da conexão TCP-IP do módulo	Camada TCP-IP
Configuração	AT+CIPSTART= <id>, <type>, <addr>, <port>	OK	Conecta o módulo como cliente usando uma conexão número <id> do tipo <type>, no servidor de endereço <addr> utilizando a porta <port>	Camada TCP-IP

Cont…

Consulta	AT+CIPSTART=?	AT+CIPSTART= (id), (type), (addr), (port)	Informa em quais conexões remota o modulo está operando como cliente, retornando o ID da conexão, o tipo, o endereço e a porta remota utilizada	Camada TCP-IP
Configuração	AT+CIPSEND=<id> ,<length>	SEND OK	Configura em qual conexão <id> se deseja enviar dados, bem como o número de caracteres que se deseja enviar	Camada TCP-IP
Execução	AT+CIPSEND		Abre a conexão com o servidor e espera os envios dos dados	Camada TCP-IP
Execução	AT+CIPCLOSE=<id>	OK	Fecha a conexão número <id>	Camada TCP-IP
Configuração	AT+CIPMUX=<mode>	OK	Habilita ou desabilita a conexão com múltiplos servidores. Onde <mode>=0 implica em conexão simples e <mode>=4 implica em conexões múltiplas (máximo 4)	Camada TCP-IP
Consulta	AT+CIPMUX?	+CIPMUX: <mode>, OK	Retorna se o módulo está operando com múltiplas (<mode>=1) conexoes ou conexao simples (<mode>=0)	Camada TCP-IP
Configuração	AT+CIPSERVER=<mode>,<port>	OK	Configura o módulo para operar como servidor, quando <mode>=1 ou deleta o servidor quando <mode>=0	Camada TCP-IP

3.4 Arduino

Arduino é uma plataforma de eletrônica open-source. Essa plataforma foi originalmente criada em 2005 para fins educacionais no Instituo Ivrea1 em Milão - Itália como continuação de um trabalho do pesquisador Hernando Barragán [45].

Utilizando essa plataforma é possível facilmente manipular e desenvolver softwares e hardwares para as mais diversas aplicações, simples ou avançadas. Com a plataforma Arduino é possível, por exemplo, coletar dados provenientes de sensores, fazer projetos de automação ou até mesmo postar um Tweet [1].

Atualmente a plataforma Arduino é amplamente adotada pelos mais diversos públicos, mas principalmente por estudantes e hobbystas. Há também uma comunidade de usuários da plataforma onde muitos projetos são compartilhados, sendo uma grande fonte de conhecimento.

O desenvolvimento do Software na plataforma pode ser feito de feita de duas maneiras. A primeira utilizando a Linguagem de Programação Arduino, e a segunda utilizando a IDE (Integrated development environment) Arduino. Na IDE Arduino a programação é feita na linguagem C++.

Quanto ao Hardware, atualmente existem mais de 30 modelos com as mais diversas configurações para atender os mais diversos projetos. Os modelos podem ser divididos em quatro grandes grupos, sendo eles: Modelos Básicos, Modelos Avançados, Modelos IoT e Modelos Wearable.

3.4.1 Modelos Básicos

São aqueles modelos com configurações de hardware mais modestas e com menor número de portas de entrada/saída. São recomendados para projetos mais simples e para projetistas iniciantes. A figura 20 traz os principais modelos dessa categoria.

3.4.2 Modelos Avançados

Essa categoria de Arduino apresenta modelos com poder de processamento muito maior e também algumas funcionalidades mais avançadas para atender projetos mais complexos. A Figura 21 apresenta os nove modelos enquadrados nessa categoria.

3.4.3 Modelos IoT

Nessa categoria enquadra-se aqueles modelos capazes de se conectar com a internet nativamente, sem o uso de módulos. A Figura 22 apresenta todos os modelos da categoria. Vale ressaltar que há alguns modelos que tem conectividade cabeada, como o Arduino Ethernet e o Arduino Leonardo ETH. Os demais modelos são capazes de se conectar sem fio.

3.4.4 Modelos Wearable

Nessa categoria estão os modelos destinados ao desenvolvimento de dispositivos como relógios, pulseiras ou qualquer outro objeto junto ao corpo. Tem a característica de serem muito pequenos e apresentar o formato redondo para melhor se acomodar ao corpo. A Figura 23 mostra todos os integrantes dessa categoria.

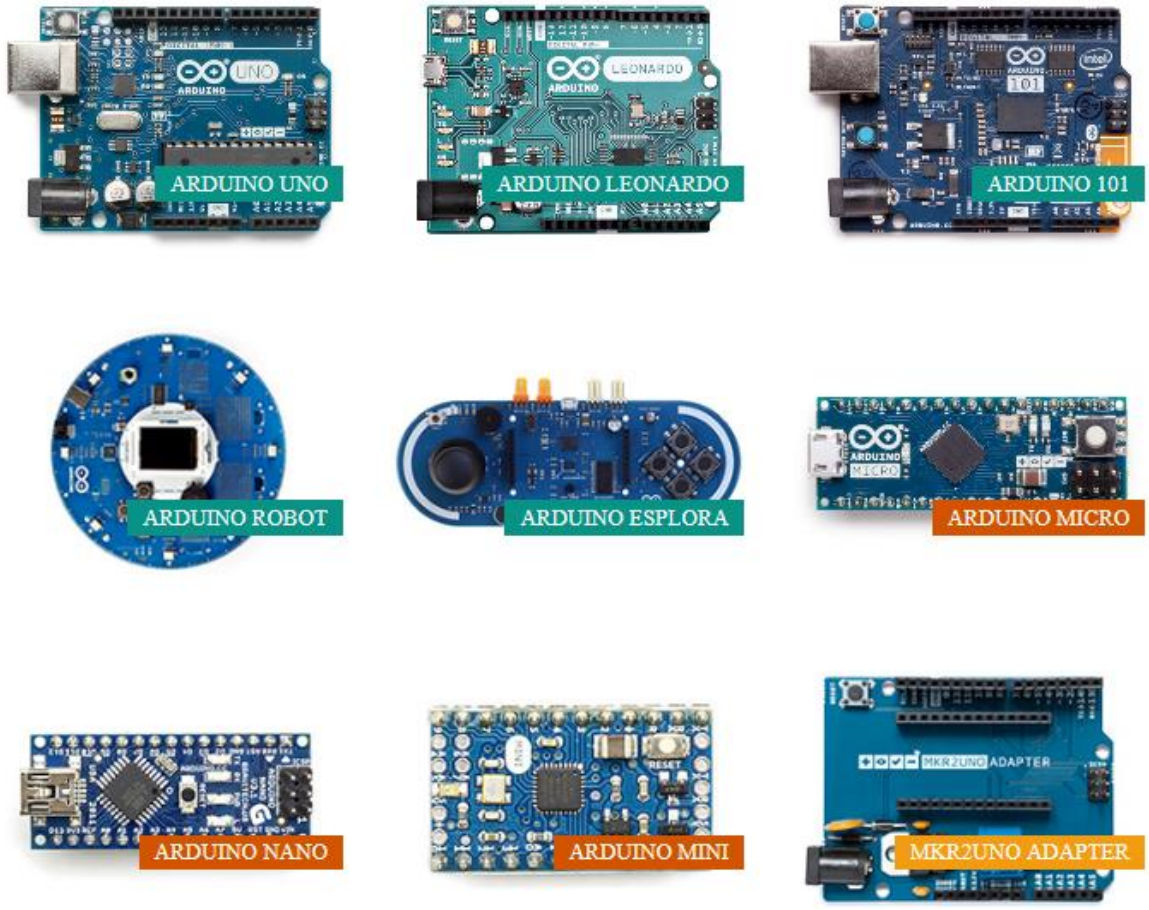

Figura 20 – Modelos Básicos de Arduino [45]

Figura 21 - Modelos Avançados de Arduino [45]

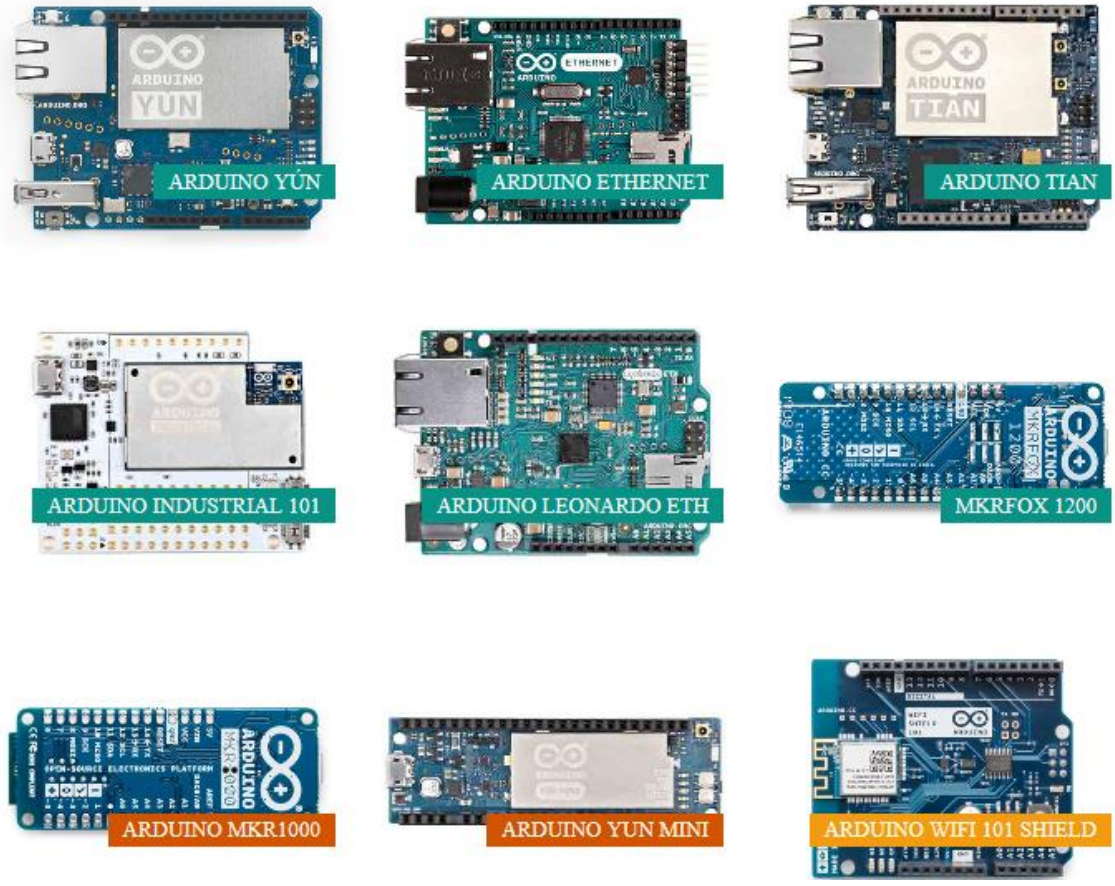

Figura 22 - Modelos IoT de Arduino [45]

Figura 23 - Modelos Wearable de Arduino [45]

3.4.5 Vantagens Arduino

A principal vantagem de utilizar a plataforma Arduino é o preço. Com o Arduino pode-se fazer centenas de projetos com o investimento muito baixo. Por se tratar de uma plataforma open-source há diversos fabricantes que vendem Arduinos a menos de cinco dólares. A maioria dos Shields e Sensores também não ultrapassam os cinco dólares.

Outra vantagem notável é a possibilidade de desenvolvimento em qualquer plataforma: Linux, Windows ou Mac. Sem contar que o desenvolvimento tanto de software quanto do hardware é muito simples quando comparado a plataformas semelhantes. O ambiente de desenvolvimento (IDE Arduino) é muito intuitiva e de fácil utilização. Há também uma comunidade que contribui diariamente com milhares de projetos que podem servir como exemplo para quem está aprendendo.

Essas facilidades permitem que crianças e pessoas com pouca experiência consigam desenvolver soluções sem grande esforço. Por esses motivos a plataforma Arduino é uma das mais consolidadas para prototipação, possuindo milhares de usuários por todo o mundo. A plataforma Arduino conseguiu democratizar a utilização de microcontroladores.

No presente trabalho será utilizado o Arduino Uno R3 então, dentre todos os modelos apresentados, esse será discutido com mais detalhes.

3.4.6 Arduino Uno R3

O Arduino Uno é uma placa de prototipação que se encaixa no grupo dos Arduino Básicos. Seu desenvolvimento é baseado no microcontrolador Atmega328 [12]. Como configuração de pinagem, ele apresentar 14 portas digitais entrada/saída, onde seis dessas podem ser utilizadas como saídas PWM (Pulse width modulation - Modulação por largura de pulso). Apresenta também seis portas de entrada analógica e um cristal de ressonância de 16MHz, suporte à conexão USB e botão de reinicialização. [12]

É uma placa completa, que contém tudo que é necessário para a prototipação de um hardware. Por possuir uma interface USB nativa, pode-se facilmente conecta-lo ao computador para fazer a programação sem a necessidade de um outro componente para fazer a integração Software-Hardware. Pela mesma porta USB é também fornecida a alimentação necessária para o Arduino Uno operar, dispensando assim fontes externar, o que simplifica bastante a fase de prototipação.

As especificações do Arduino Uno R3 segundo o fabricante [12], são apresentadas na Tabela 3.

Tabela 3 – Especificações Elétricas Arduino Uno R3

Microcontrolador	**ATmega328**
Voltagem Operacional	5V
Voltagem recomendada de entrada	7-12V
Voltagens limite de entrada	6-20V
	Cont...
Pinos de entrada/saída digitais	14 (6 podem operar como PWM)
Pinos de entrada analógica	6
Corrente CC por pino entrada/saída	40 mA
Corrente CC para o pino de alimentação 3,3V	50 mA
Memórica Flash	32 KB
SRAM	2 KB
EEPROM	1 KB
Velocidade de Clock	16 MHz

Por ser uma plataforma open-source, o fabricante disponibiliza um modelo esquemático para referência caso qualquer pessoa ou instituição queira fabricar o seu próprio Arduino. O modelo esquemático pode ser visto em detalhes no Apêndice F.

Como já mencionado anteriormente, o Arduino Uno pode ser alimentado pela entrada USB. Todavia há ainda a opção de alimenta-lo por um conversor de corrente alternada para contínua ou até mesmo por baterias. Pode-se também utilizar um conector do Tipo Jack 2.1mm,

conforme a Figura 24, desde que se respeite as especificações da Tabela 3. Caso contrário, o Arduino Uno pode operar de maneira instável ou ser danificado permanentemente.

Figura 24 - Arduino Uno R3 e Conector Jack 2.1mm

Os pinos de alimentação presentes do Arduino Uno são:

• Vin: Entrada de alimentação quando se utiliza uma fonte externa de alimentação. Todavia, caso utilize a entrada USB ou a entrada Jack 2.1mm, esse pino pode funcionar como uma fonte de alimentação regulada de 5 volts.

• GND: Esses pinos são os pinos terra.

• 3V3: Pino que fornece 3.3 volts, podendo ser usado como fonte para algum circuito externo, desde que não ultrapasse o limite de corrente previsto na Tabela 3.

• 5V: Pino que fornece 5 volts, assim como o pino 3V3, pode também ser utilizado como fonte para qualquer outro circuito que opere dentro das especificações do fabricante.

Como já mencionado, ao total, tem-se 14 pinos que podem ser usados como saída ou entrada digital, sendo 6 com a funcionalidade de operar como saídas PWM. Há também 6 pinos de entrada analógica e alguns outros com funções específicas.

Dentre todos esses pinos, aqueles com dupla função ou funções especiais são:

- Pinos 0 e 1: podem operar também como receptor (Pino 0) e transmissor (Pino 1) para como comunicação serial.
- Pinos 2 e 3: podem ser configurados para disparar interrupções, conforme alguma variação sensível pelo circuito.
- Pino 4 e 5: São pinos que podem ser utilizados para habilitar a comunicação I²C ((Inter-Integrated Circuit)
- Pinos 10, 11, 12 e 13: são pinos que podem viabilizar a comunicação SPI (Serial Peripheral Interface Bus)

- Pino 13: é um pino de entrada/saída digital comum, mas com um LED (Light-emitting diode) conectado em paralelo. Quando funcionando como saída digital o LED acende quando o pino 13 está em estado alto, e o LED apaga caso contrário
- Pino AREF: Pino utilizado como referência analógica, quando se utiliza alguma porta de entrada analógica
- Pino Reset: Pino para reinicialização do Arduino Uno. É acionado em nível lógico baixo.
- Pinos A0 a A5: São os pinos de entrada analógica com 10 bits de resolução. O padrão de leitura é de tensões entre 0 e 5 volts, todavia pode-se alterar o limite superior utilizando o já mencionado pino AREF.

3.4.7 Acessórios: Shields e Sensores

Assim como existem vários modelos de Arduino, existem também vários componentes que podem se integrar aos Arduinos para incorporar várias outras finalidades. Podemos dividi-los em duas grandes categorias: Shields e Sensores.

Os Shields são componentes capazes de dar funcionalidades extras, funcionam como um complemento do Arduino para uma função específica. A seguir serão apresentados alguns desses.

- Shield Micro SD

Permite que o Arduino leia e armazene arquivos em um cartão de memória Micro SD. Pode ser visto na Figura 25.

- Shield Relay

Permite o Arduino acionar cargas maiores que ultrapassem as especificações de limite de corrente. É apresentado na Figura 26 [45].

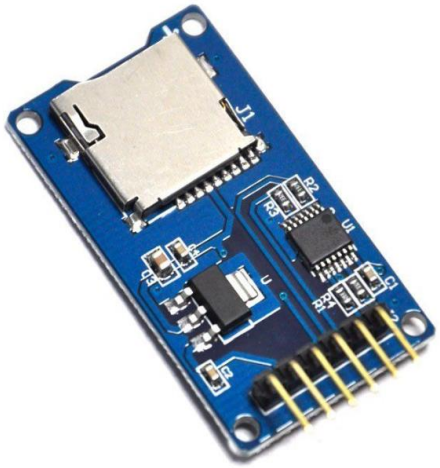

Figura 25 – Shield Micro SD [45]

Figura 26 – Shield Relay [45]

- Shield Wifi ESP-01

Esse Shield tem a capacidade de possibilitar a conexão com a Internet via Wifi por aqueles Arduino que não pertencem ao grupo de modelos IoT, previamente apresentados. Esse Shield é apresentado na Figura 27 e será discutido em detalhes na seção ESP8266-01 inteiramente dedicada a esse módulo.

Já os sensores são dispositivos que tem a capacidade de transformar alguma grandeza física em um sinal elétrico mensurável. Existem inúmeros sensores disponíveis no mercado

para se integrar ao Arduino e desenvolver as mais diversas aplicações. A seguir serão apresentados alguns dos sensores disponíveis no mercado.

Figura 2177 – ESP8266-01

- Sensor DTH11

Este é um sensor capaz de detectar alterações de temperatura e umidade [9]. Apresenta 4 terminais, onde dois são de alimentação, um para comunicação serial responsável pela transmissão dos sinais de temperatura e umidade, e o último pino não é utilizado. O DTH11 e a disposição dos pinos são apresentados na Figura 28.

Figura 28 – Pinagem Sensor Temperatura e Umidade Dth11 [9]

Esse sensor contém um microcontrolador interno responsável pelo envio de um sinal digital de 40 bits através dos pinos de comunicação serial [9]. A composição desse sinal é dada por:

8 bits valor inteiro da temperatura	8 Bits valor decimal da temperatura	8 Bits valor inteiro da humidade	8 Bits valor decimal da humidade	8 Bits de verificação

Vale ressaltar, que o DTH11 é pré-calibrado pelo fabricante e os dados da calibração são salvos no microcontrolador interno do sensor. Sendo assim não é necessário nenhum tipo de calibração. [9]

A seguir, a Tabela 4 resume as principais características de operação [9].

Tabela 4 – Limites Operação Dth11

	Faixa de Operação	Acurácia
Temperatura	0°C a 50°C	±2°C
Humidade Relativa	20% a 90%	±5%

Já as especificações elétricas são apresentadas na Tabela 5 [9].

Tabela 5 – Especificações Elétricas Dth11

	Mínimo	Típico	Máximo
Tensão de alimentação	3 volts	5 volts	5.5 volts
Corrente	0.5 mA		2.5 mA

Um circuito de aplicação da integração do DTH11 é apresentado na Figura 29, onde se fez a alimentação utilizando a fonte interna do Arduino. Entre o pino Vcc e Signal empregou-se um resistor de pull-up de 10KΩ para evitar flutuações de níveis de tensão e garantir que a comunicação seja feita da forma correta. Finalmente conectou-se o pino de sinal do DTH11 a porta digital 7 do Arduino.

- Sensor MQ-05

Esse sensor é capaz de detectar a presença do gás de cozinha no ambiente. Sendo muito empregado em sistemas de prevenção de incêndios. Esse sensor pode ser visto na Figura 30.

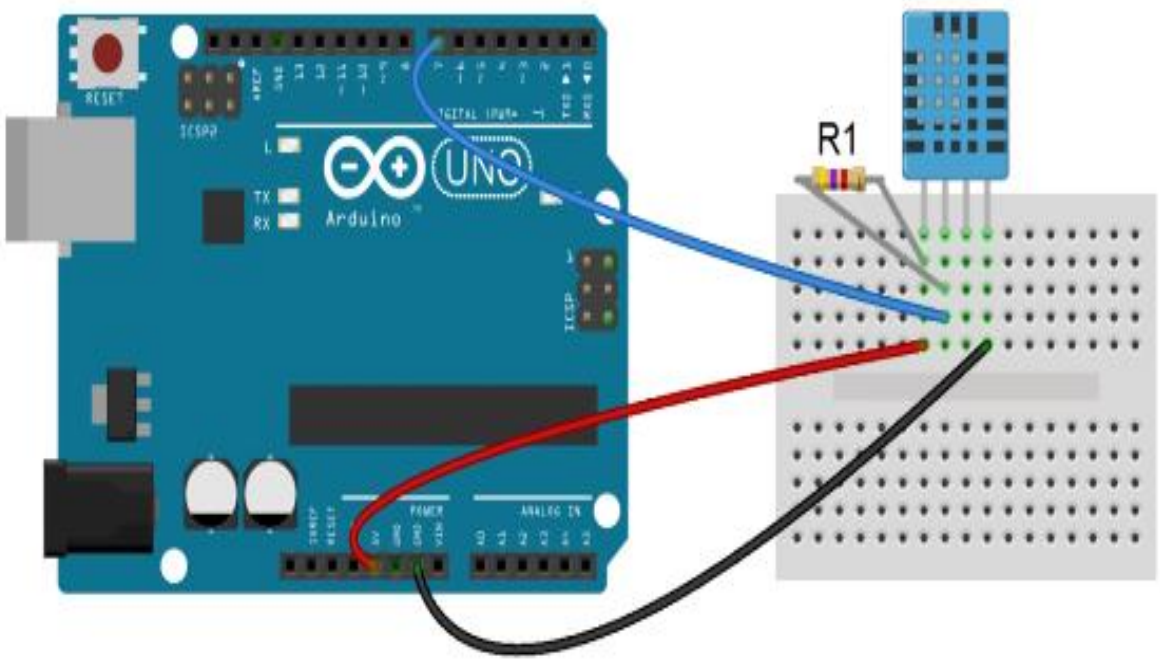

Figura 29 – Integração Dth11 e Arduino Uno R3

Figura 30 – Sensor Gás MQ-05 [38]

Este é um sensor construído a base de Dióxido de Estanho (SnO2), que tem a propriedade de apresentar uma condutividade elétrica mais baixa quando exposto ao ar puro. Todavia, na presença de algum poluente a condutividade do sensor aumenta conforme a Figura 31 [38]. Nota-se na referida figura, que a condutividade (Rs/R0) não varia com a concentração de ar, todavia quando exposto aos gases H2, LPG (Liquefied Petroleum Gas - Gás Liquefeito de Petróleo), CH4, CO, Álcool ou Propano a condutividade varia inversamente proporcional.

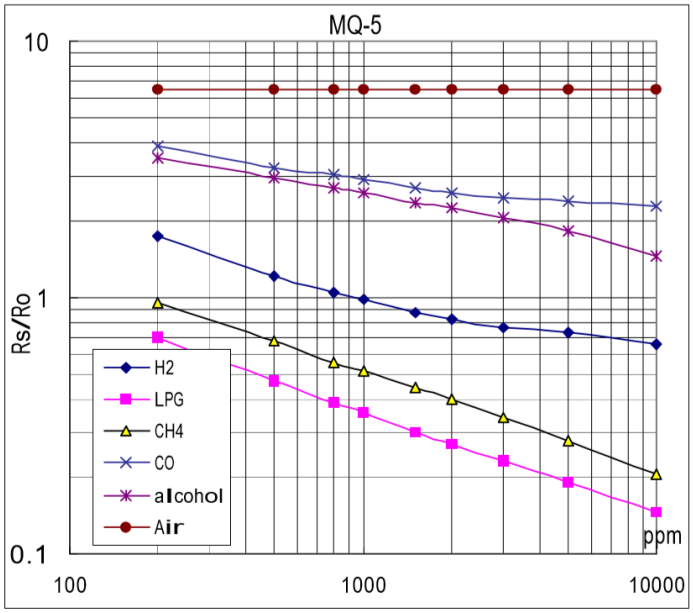

Figura 31 – Variação da Resistencia de Referência pela Concentração de Diversos Gases [38]

O sensor apresenta um total de quatro pinos, onde dois são de alimentação Vcc e GND, e outros dois são os sinais DO e AO. DO é um sinal digital que assume nível lógico alto na presença de algum poluente. Já AO é o sinal analógico que indica a concentração de um determinado gás no ambiente. Detalhes da pinagem podem ser observados na Figura 32, onde da esquerda para a direita então localizados os pinos: AO, DO, GND e Vcc.

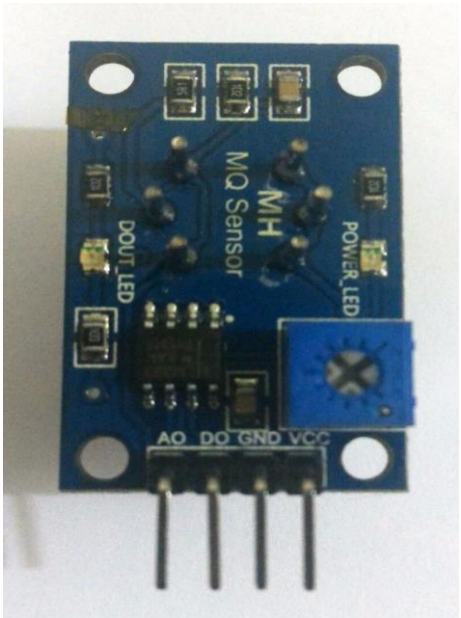

Figura 32 – Pinagem MQ-05

Caso opte-se em utilizar o sinal digital, o fabricante recomenda a calibragem do equipamento, pois como já discutido anteriormente a sua condutividade varia com gases outros

que podem estar presentes no ambiente. Para a calibragem, deve-se ajustar a resistência entre o pino Vcc e GND em 20KΩ variando o potenciômetro existente no sensor com auxílio de um multímetro.

A tensão nominal de operação do sensor é de 5 volts, podendo haver uma variação de 0.1 volts. Em hipótese alguma deve-se ultrapassar esses limites, caso contrário poderá ocorrer danos permanentes no sensor MQ-05.

Um circuito de aplicação de integração do MQ-05 com o Arduino é apresentado na Figura 33, onde se a alimentação do sensor utilizando a fonte interna do Arduino e conectou-se a saída analógica do MQ-05 a uma porta analógica do Arduino. Esse circuito é empregado quando deseja-se calcular a concentração de um dos gases supracitados.

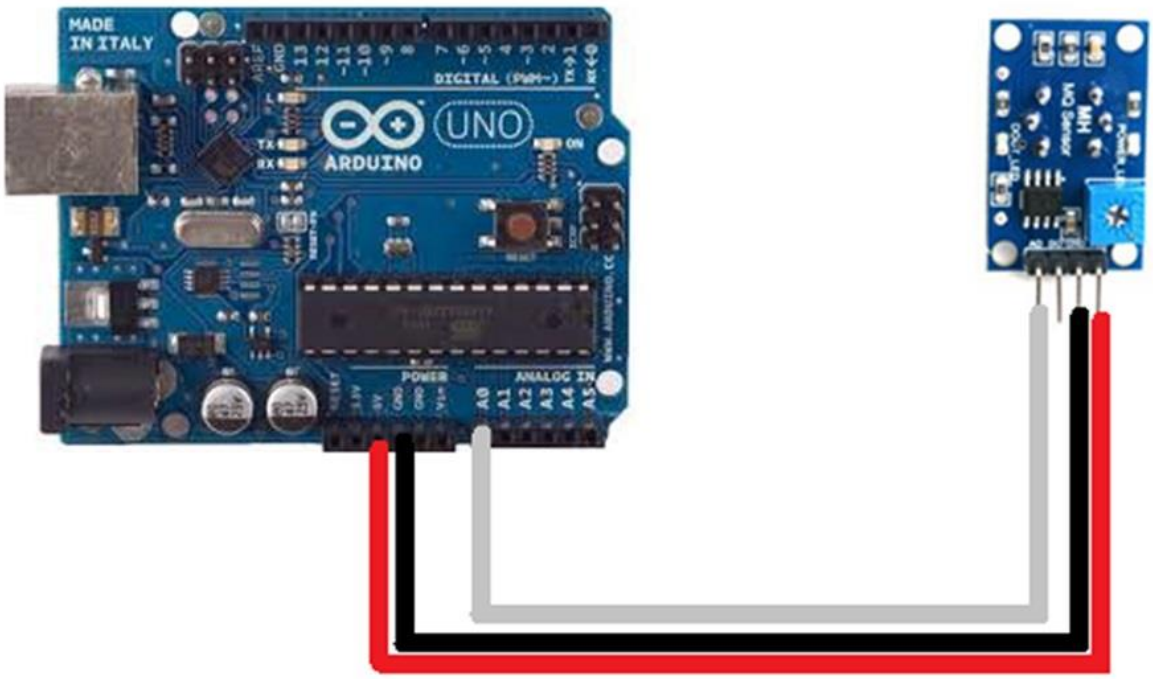

Figura 33 - Integração MQ-05 e Arduino Uno R3

- Sensor de Chamas

Esse sensor é capaz de detectar ondas de luz com comprimento entre 760nm e 1100nm, sendo aplicado para a detecção da presença de chamas no ambiente a até um metro de distância. Esse sensor contém um comparador LM393 que garante uma estabilidade maior na leitura dos dados. Vale ressaltar também a presença de um potenciômetro de calibração. O sensor é apresentado na Figura 34.

Figura 34 – Sensor de Chamas

Este sensor apresenta 4 pinos, que podem ser vistos em detalhes na Figura 35.

Os pinos Vcc e GND são os pinos de alimentação do módulo. Já os pinos DO e AO são os pinos de saída digital e analógica, respectivamente. O pino AO é empregado para indicação do nível de intensidade de emissão de luz dentro do espectro de leitura. Já o pino DO pode ser calibrado com a variação do potenciômetro presente no sensor para que mude de estado lógico na intensidade de emissão desejada.

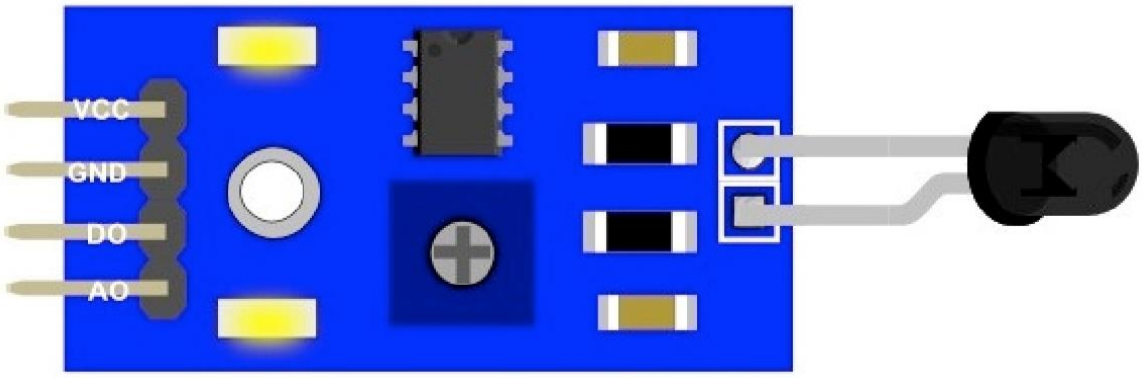

Figura 35 – Pinagem Sensor de Chamas

A faixa de tensão de operação do sensor é de 3.3 volts a 5 volts, não podendo ser ultrapassada de maneira alguma, caso contrário pode ocorrer danos permanentes ou operação instável.

Um circuito de aplicação de integração do sensor com um Arduino é apresentado na Figura 36, onde a alimentação do sensor é feita utilizando a fonte interna do Arduino e o pino de sinal analógico (AO) é conectado a uma porta analógica do Arduino.

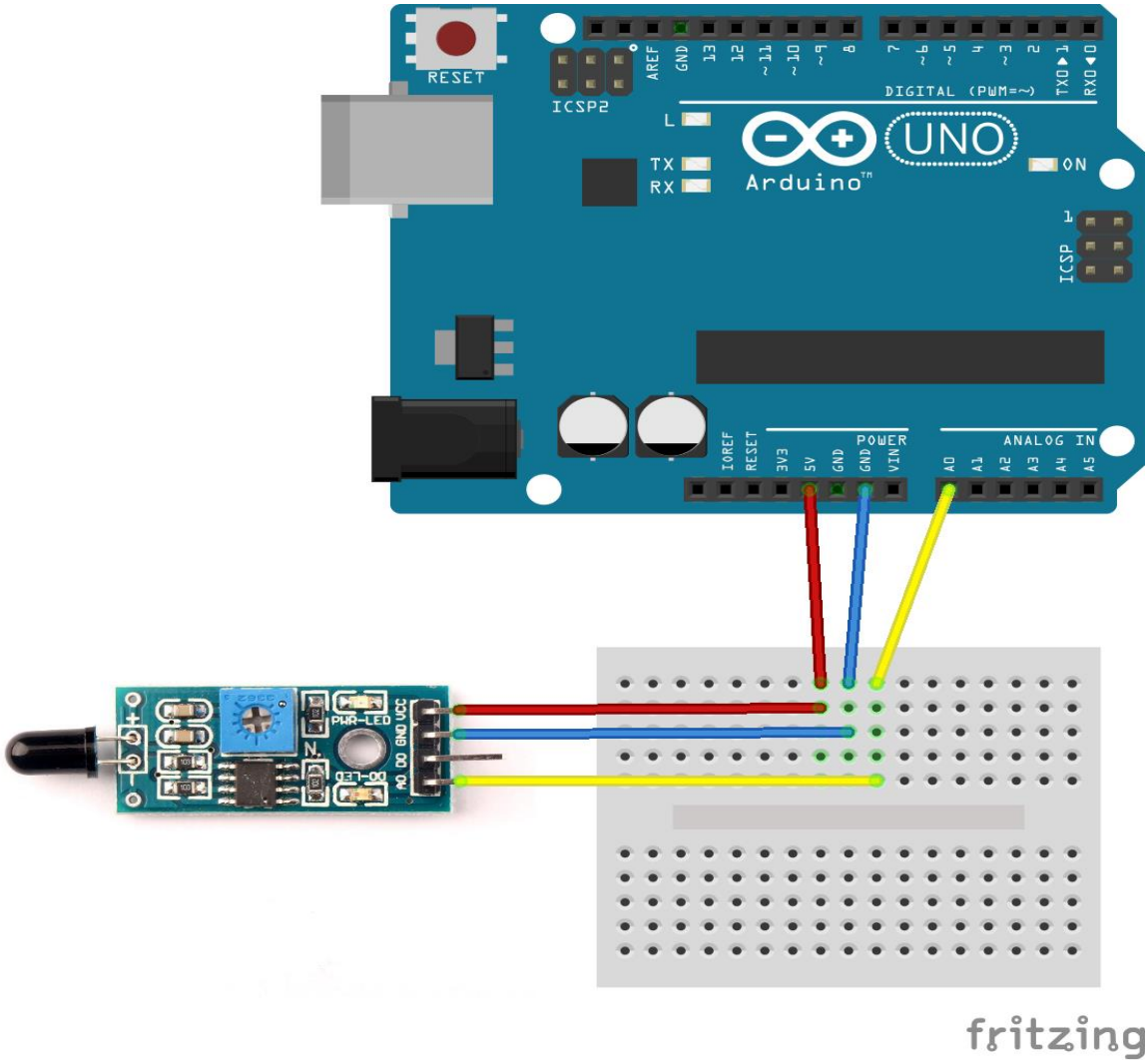

Figura 36 - Integração Sensor de Chamas e Arduino Uno R3

4 Material e Métodos

4.1 Material

Para desenvolver o sistema IoT de monitoramento contra incêndios de depósitos foram necessários recursos de Hardware, Software e alguns serviços de nuvem. Os quais serão apresentados nos próximos parágrafos.

- Hardware

  Utilizou-se:

  - Arduino Uno R3
  - Módulo Wifi ESP8266-01
  - Sensor de temperatura e umidade DTH11
  - Sensor de Gás MQ-2
  - Sensor de Chamas (Infravermelho)
  - Protoboard
  - Fonte externa para protoboard MB-102 (3.3 volts e 5 volts)
  - Resistores 10KΩ
  - Diversos Jumpers

- Softwares

  - IDE Arduino 1.8.2
  - Microsoft SQL Server Management Studio
  - Power BI Desktop

- Serviços de Nuvem
  - Web App Azure
  - SQL Database Azure
  - Power BI
  - ThingSpeak

4.2 Métodos

Nessa seção serão discutidos os métodos empregados no desenvolvimento do sistema de prevenção de incêndios baseado em arquitetura IoT. Essa discussão se dará em quatro grandes tópicos:

1. Preparação da infraestrutura baseada em nuvem
2. Desenvolvimento do Hardware
3. Desenvolvimento Software
4. Desenvolvimento dos relatórios de análises

4.2.1 Preparação da Infraestrutura Baseada em Computação em Nuvem

Os três componentes de serviço de infraestrutura em nuvem utilizados foram o Web App Azure, ThingSpeak e o SQL Database Azure. Os dois primeiros serviços terão o papel de receber os dados enviados pelo Arduino, com a diferença de que o Web App Azure fará o armazenamento dos dados recebidos no banco de dados SQL, enquanto o ThingSpeak irá exibir os valores recebidos em gráficos pré-configurados inerentes a plataforma.

4.2.1.1 Implementação do Web App Azure e SQL Database Azure

Para fazer a implementação do Web App Azure e sua conexão com o SQL Database Azure seguiu-se os passos ilustrados na Figura 37. Cada um desses passos serão comentados nos próximos tópicos, todavia o processo em detalhes pode ser verificado no Apêndice A.

- Login no Portal Azure: Nessa primeira etapa, entrou-se no portal com as credencias da conta Azure tendo acesso a todos recursos da plataforma.
- Provisionamento do serviço Mobile Apps: Fez-se o provisionamento do serviço Mobile Apps
- Configuração dos recursos EasyTable: Nessa etapa fez-se a configuração de um dos recursos disponíveis dentro do Mobile Apps, a EasyTable.
- Conexão com SQL Database Azure: Para armazenar os dados coletados, necessita-se de uma Banco de Dados, então nessa etapa fez-se o provisionamento de um SQL Database Azure

- Configuração do SQL server: Para o SQL Database Azure funcionar propriamente, necessita-se de um servidor dedicado. Então nessa etapa, fez-se o provisionamento de um servidor SQL dedicado.
- Inicialização EasyTable: Com toda a infraestrutura montada, nesse etapa fez-se a inicialização da tabela (EasyTable) que irá receber os dados para o armazenamento do banco de dados.
- Criação das Colunas na EasyTable: Como última etapa do processo, criou-se colunas para receber os dados enviados pelo Arduino.

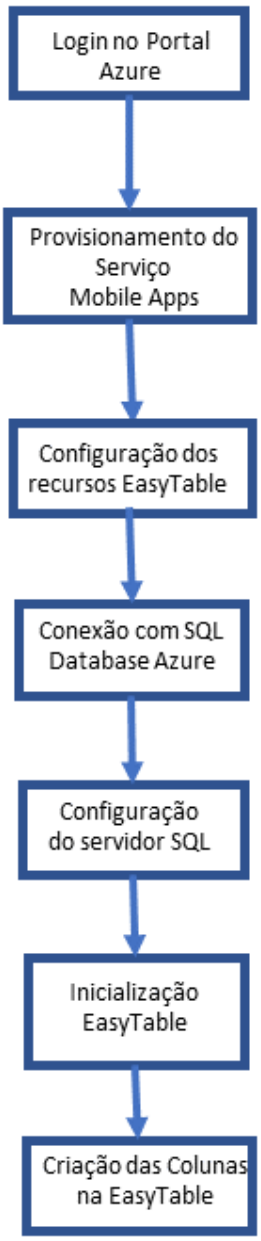

Figura 37 - Fluxograma de Implementação do Web App e SQL Database Azure

Após todos esses passos, obteve-se o Web App Azure e o SQL Database Azure implementados. Os resultados dessas implementações podem são apresentados na Tabela 6.

Tabela 6 – Resultado Implementação Web App Azure

Item	Resultado
URL Web App Azure	http://iotmonografia.azurewebsites.net
Nome SQL Database Azure	iotdatabase
Nome Servidor SQL	Iotservidor.database.windows.net
Nome usuário administrador do Servidor SQL	iotusuario
Nome da tabela "Easy Table"	iottabela

4.2.1.2 Preparação do ambiente ThingSpeak

Esse serviço tem como funcionalidade principal receber os dados do Arduino e exibi-los em tempo real em gráficos pré configurados pela plataforma. Então provisionou-se um canal ThingSpeak se conectar ao Arduino. Para executar essa tarefa fez-se:

- Criação do Canal
- Configuração do Canal
- Obtenção das chaves de acesso

Detalhes desse provisionamento podem ser vistos no Apêndice G.

Os resultados obtidos nessa etapa podem ser resumidos na Tabela 7.

Tabela 7 – Resultados Implementação do Canal Thingspeak

Item	Resultado
Nome do Canal	Streaming Arduino
Chave do Canal	VCAPZPZZJM3DJMO
ID do Canal	284932

4.2.2 Prototipação do Hardware

Como parte da prototipação da solução de monitoramento contra incêndios, montou-se um Hardware com todos os componentes listados na seção de Materiais. O processo de

prototipação pode ser dividido em duas partes. A primeira seria a integração do Arduino Uno R3 com o módulo Wifi ESP8266-01. Já a segunda seria a inserção dos diversos sensores. Cada uma dessas etapas será discutida em detalhes nas próximas seções.

4.2.2.1 Integração Arduino Uno R3 e módulo Wifi ESP8266-01

Baseado no datasheet e nas especificações do fabricante, montou-se o circuito conforme a Figura 38. Nota-se no circuito a utilização de uma fonte externa com saídas de 3.3 volts e 5 volts, a MB-102. Como já discutido anteriormente, a utilização dessa fonte dá-se pela demanda de corrente do módulo wifi ser maior do que o limite máximo de fornecimento da fonte interna do Arduino.

Vale ressaltar também a utilização de um divisor de tensão entre o pino de transmissão do Arduino e o pino de recepção do módulo Wifi. Esse divisor de tensão é formado por dois resistores de 10KΩ e tem como papel limitar a tensão empregada no módulo Wifi em no máximo 3.3 volts, conforme as recomendações do fabricante. Outra conexão de extrema importância foi curto-circuitar os pinos terras de todos os componentes para garantir a mesma tensão de referência. As demais conexões são feitas diretamente entre o Módulo Wifi, o Arduino e a fonte externa.

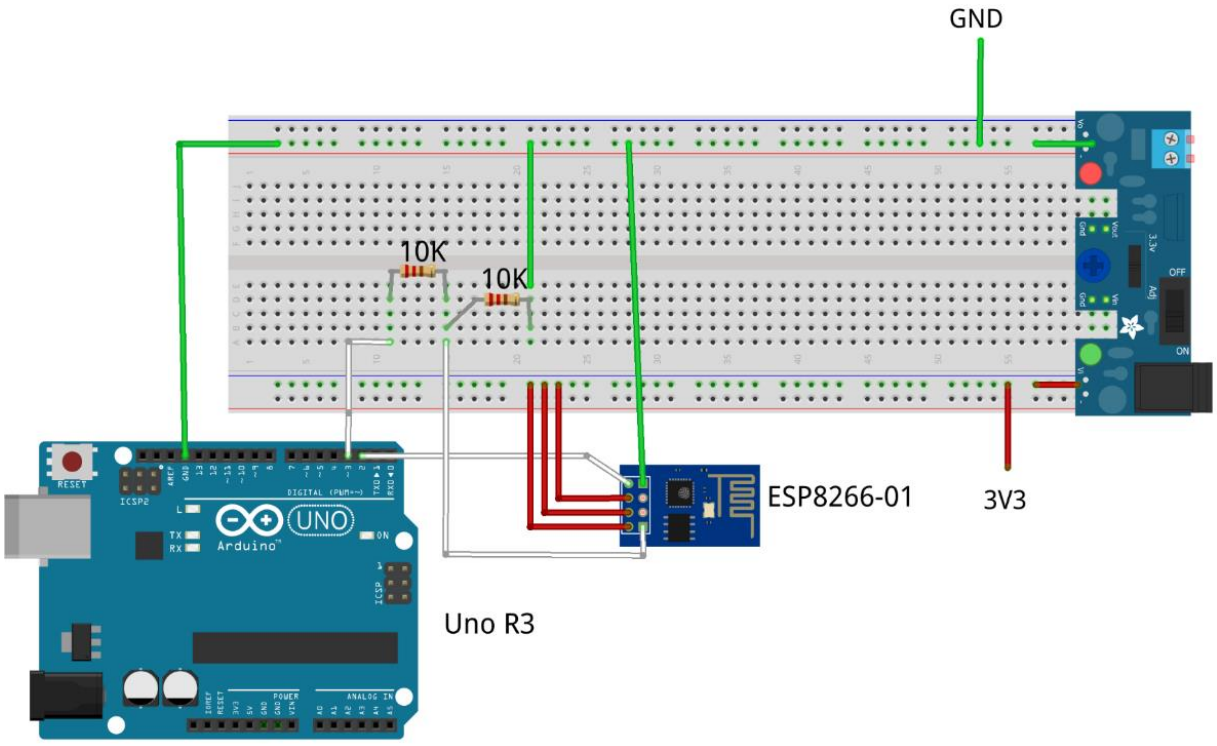

Figura 38 – Circuito De Integração Do Módulo Esp8266-01 E Arduino Uno R3

4.2.2.2 Inserção dos Sensores

Seguindo as recomendações dos fabricantes e fazendo a calibragem dos sensores quando necessário, acrescentou-se ao circuito representado pela Figura 39 os sensores de temperatura e umidade, detecção de chamas e o sensor de gás.

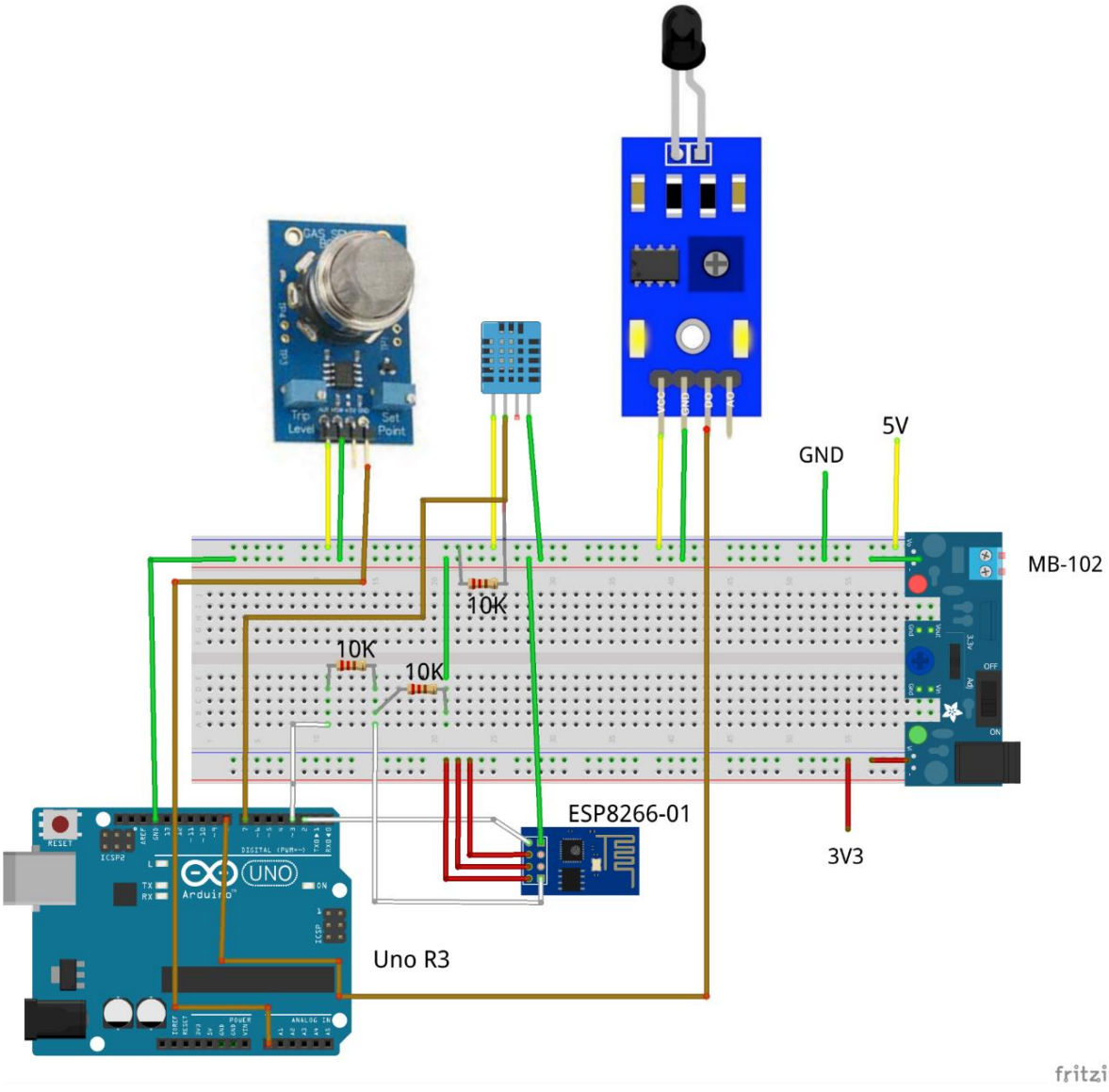

Figura 39 – Circuito de Prototipação Final

Nota-se que para incluir os sensores, bastou alimentá-los com 5 volts, interconectar os pinos terra e conectar o pino de sinal de saída dos sensores com alguma porta de entrada do Arduino. Para o sensor de gás conectou-se o pino de saída analógica AO com a entrada analógica A0 do Arduino Uno R3. Já para o sensor de chamas, conectou-se a saída digital DO com a entrada digital 8 do Arduino. Finalmente, para o último sensor de temperatura e umidade, conectou-se o seu pino de Sinal com a entrada digital 7 do Arduino. Vale ressaltar ainda o

emprego do resistor de pull-up de 10KΩ na interconexão do último sensor, conforme as recomendações do fabricante previamente comentadas.

4.2.3 Desenvolvimento do Software

Para desenvolver o Software responsável pela orquestração entre o Arduino Uno R3 e os serviços de nuvem, utilizou-se o ambiente de desenvolvimento da plataforma Arduino empregando a linguagem de programação C++, o código completo encontra-se no Apêndice D.

O algoritmo desenvolvido é apresentado na Figura 40. Cada uma das etapas do Algoritmo será discutida nos próximos tópicos.

- Início: nessa etapa fez-se todas as configurações iniciais, como a declaração de variáveis e bibliotecas que serão utilizadas ao longo do desenvolvimento.
- Estabelece e testa comunicação Arduino - Módulo Wifi: Nessa etapa, criou-se uma ponte serial para efetivar a troca de dados entre o Arduino e o Módulo Wifi. Através dessa porta, enviou-se comandos AT para verificar se a comunicação está ocorrendo corretamente.
- Comunicando?: Caso a comunicação esteja sendo feita corretamente, o algoritmo evolui para o próximo passo, caso contrário retorna para a etapa anterior.
- Configura o módulo Wifi e Conecta à Internet: Nessa etapa, fez-se uma sequência de coman4dos AT para efetivar a conexão do Hardware com a internet.
- Conectado?: Caso ocorra sucesso na tentativa de conexão com a internet o algoritmo evolui para a próxima etapa, caso contrário refaz-se todo o processo de comunicação do Arduino com o módulo Wifi
- Abre comunicação com o ThingSpeak: Através dos comandos AT, abriu-se a comunicação com o serviço de nuvem ThingSpeak indicando o endereço do mesmo e declarando quantos caracteres deseja-se enviar.
- Envia dados da leitura dos sensores: Novamente, através dos comandos AT, enviou-se os dados coletados pelos sensores.

- Fecha a comunicação com o canal ThingSpeak: Para efetivar o envio, é necessário declarar que se terminou o envio dos dados. Então, através de um comando AT, fecha-se a comunicação.
- Abre comunicação com o Web App Azure: Através dos comandos AT, abriu-se a comunicação com o serviço de nuvem Web App Azure indicando o endereço do mesmo e declarando quantos caracteres deseja-se enviar.
- Envia dados da leitura dos sensores: Novamente, através dos comandos AT, enviou-se os dados coletados pelos sensores.
- Fecha a comunicação com o canal Web App Azure: Para efetivar o envio, é necessário declarar que se terminou o envio dos dados. Então, através de um comando AT, fecha-se a comunicação e reinicia-se o ciclo do algoritmo.

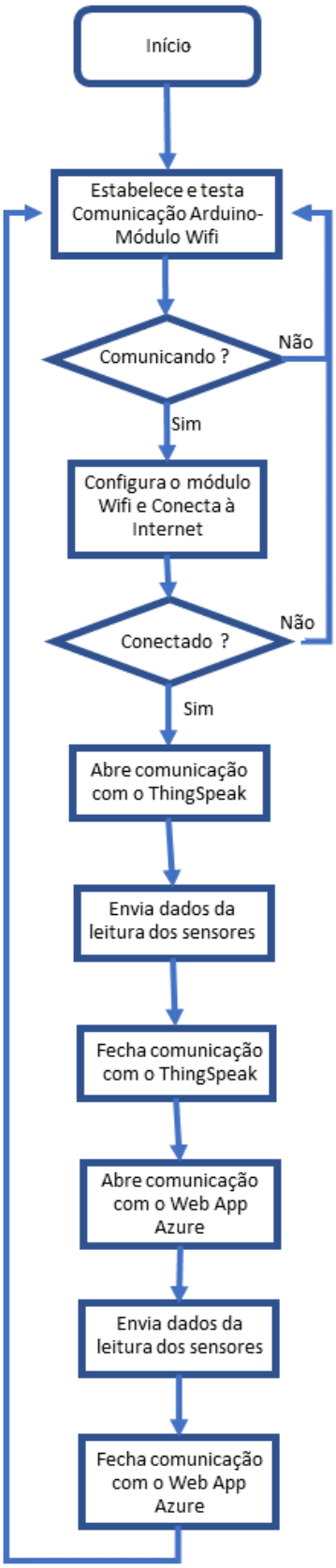

Figura 40 – Algoritmo Desenvolvido

4.2.4 Desenvolvimento dos relatórios de análises

A fim de visualização e monitoramento dos dados coletados, disponibilizou-se duas análises para o usuário final. A primeira em tempo real, praticamente, exibindo as últimas 100 amostras coletadas e a segunda apresentando uma análise histórica agregando toda a massa de dados coletada desde do início da operação do sistema

4.2.4.1 Análise em Near Real Time

A primeira análise apresenta em gráficos os dados enviados pelo Arduino em tempo real, praticamente, e é inerente da plataforma ThingSpeak. Sua configuração se dá através do código desenvolvido, onde deve-se enviar uma mensagem no formato:

GET:https.:api.thingspeak.com/update?api_key=<CHAVEAPI>&field1=<Temperatura>&field2=<Umidade>&field3=<Gas>&field4=<Chama>&field5=<ID>

Onde <CHAVEAPI> é a chave fornecida pela plataforma, conforme a Tabela 7. Já os campos <Temperatura>, <Umidade> , <Gas>, <Chama>, <ID> são as variáveis coletadas pelo Arduino que representam, em ordem, a temperatura do ambiente em graus Celsius, a umidade relativa do ar, indicativo da concentração de gás de cozinha (GLP), presença ou não de chamas no ambiente, e por último o número de identificação.

Detalhes de como foi feito o envio dos dados podem ser vistos no código completo, no Apêndice D.

4.2.4.2 Análise histórica

Para a análise histórica, utilizou-se o software Power BI Desktop. Através desse software estabeleceu-se a comunicação com o banco de dados onde estão armazenados todos os dados coletados. Detalhes de como efetuar essa conexão entre o Power BI Desktop e o Bando de Dados podem ser consultados no Apêndice B.

Após estabelecer a conexão, utilizando as funcionalidades previamente comentadas do Power BI, construiu-se diversas análises históricas para cada uma das variáveis coletadas e as disponibilizou através de gráficos interativos.

Em seguida, publicou-se as análises no serviço de nuvem do Power BI de maneira que todos aqueles que tenham permissão para visualizar as análises consigam visualizar os gráficos através de qualquer dispositivo ou plataforma, móvel ou não.

4.2.5 Desenvolvimento de Alertas

A fim de alertar possíveis alterações no ambiente e mantê-lo sob constante monitoramento, fez-se a conexão da plataforma ThingSpeak com a rede Social Twitter utilizando um recurso da plataforma ThingSpeak chamado "React". Esse recurso permite tomar algumas ações, sendo uma delas postar uma mensagem na rede Social Twitter. Essa ação de se postar uma mensagem é comumente chamada no Brasil de "Tweetar", e as mensagens são chamadas de "Tweets".

Então, criou-se um usuário no Twitter chamado "IoT - Arduino Uno R3". Esse usuário será aquele que fará as postagens dos Tweets. Dessa maneira programou-se o usuário "IoT - Arduino Uno R3" para postar diversos alertas, conforme a Tabela 8.

Tabela 8 – Alertas Twitter

Nome do alerta	**Variável monitorada**	**Frequência de verificação**	**Formato do Tweet**
Alerta de Incendio	Chama	Toda vez que um dado é inserido	Alerta de incêndio registrado em: %%datetime%% #IoTJoaoLucindo
Detecção Alta concetração de gás	Gas	Toda vez que um dado é inserido	Alta concetração de gás !!! Alerta registrado em: %%datetime%% #IoTJoaoLucindo
Não Atualiza	Todas	5 minutos	Sistema não está atualizando.... alerta registrado em: %%datetime%% #IoTJoaoLucindo
Temperatura	Temperatura	30 minutos	Temperatura atual: %%channel_IDCANAL_field_NUMEROCAMPO%% °C Registrado em: %%datetime%% #IoTJoaoLucindo
Umidade	Umidade	30 minutos	Umidade relativa do ar: %%channel_IDCANAL_field_NUMEROCAMPO%% % Registrado em: %%datetime%% #IoTJoaoLucindo

Os alertas de Incêndio e de alta concentração de gás são alertas críticos, por esse motivo a frequência de atualização é muito maior quando comparada com os demais, que tem o objetivo de somente informar a temperatura e a umidade do local ou informar que o sistema não está sendo atualizado.

Vale ressaltar que os itens %%datetime%% e %%channel_ID_field_ #NUMEROCAMPO%% são referentes a sintaxe da plataforma. O primeiro item fornece a data

e hora em que ocorreu o alerta, já o segundo fornece o valor da variável desejada, que no caso é a temperatura e a umidade.

Detalhes de como configurar a plataforma ThingSpeak para publicar tweets podem ser consultados no Apêndice C.

5 Resultados

Como resultado desse trabalho obteve-se a arquitetura ilustrada na Figura 41. Onde tem-se o Arduino trocando mensagem com o módulo Wifi via comunicação serial através de comandos AT. Por sua vez, o Módulo Wifi envia os dados para a plataforma ThingSpeak e para o Web App Azure.

Os dados recebidos pela plataforma ThingSpeak receberam um tratamento em near Real Time e foram dispostos em gráficos. Os mesmos dados também são a entrada de um sistema lógico criado através da funcionalidade "React" previamente comentada. Quando essa lógica estabelecida apresenta um resultado verdadeiro dispara-se tweets de alertas e notificação.

Por outro lado, os dados recebidos pelo Web App Azure foram armazenados em um banco de dados SQL Azure através da funcionalidade "EasyTable". Com os dados armazenados, conectou-se o ao software PowerBI e elaborou-se análises históricas.

Cada um dos resultados supracitados serão discutidos em detalhes nos próximos tópicos

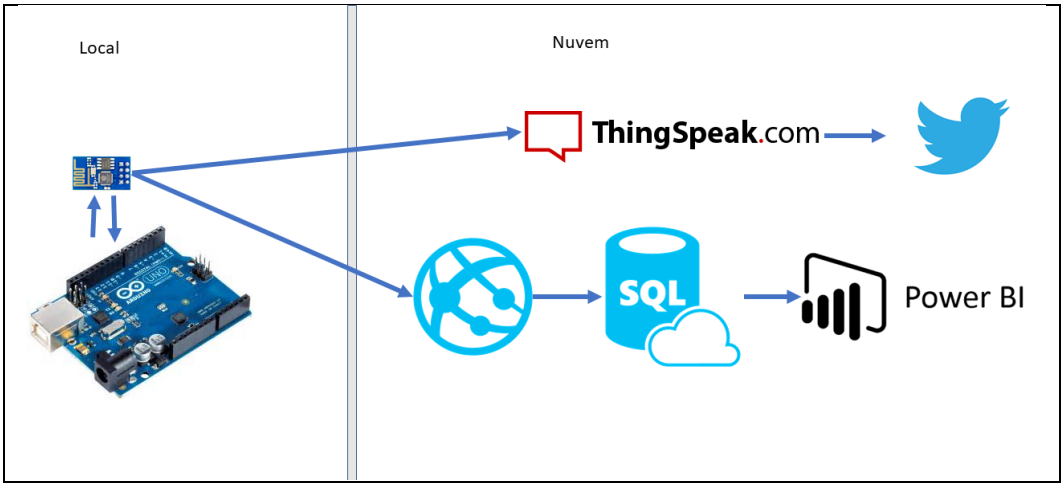

Figura 41 – Arquitetura Final

5.1 Análise em tempo real, praticamente

A Figura 42 mostra o resultado da análise em tempo real para os valores de temperatura, umidade, indicador de gases e a presença ou não de chamas. Por limitações da plataforma, os dados são atualizados a aproximadamente a cada 20 segundos, uma frequência muito satisfatória para que o sistema propõe.

No gráfico "Field 1 Chart", observa-se a como a temperatura variou nos últimos minutos. Nota-se que no período analisado a temperatura se manteve próxima dos 18 °C. Análises semelhantes podem ser feitas observando os gráficos “Field 2 Chart”, “Field 3 Chart” e “Field 4 Chart”, que mostram respectivamente a evolução da umidade relativa, do indicador de gás e se há chamas ou não no ambiente.

Essa análise pode ser conferida pelo endereço https://thingspeak.com /channels/284932.

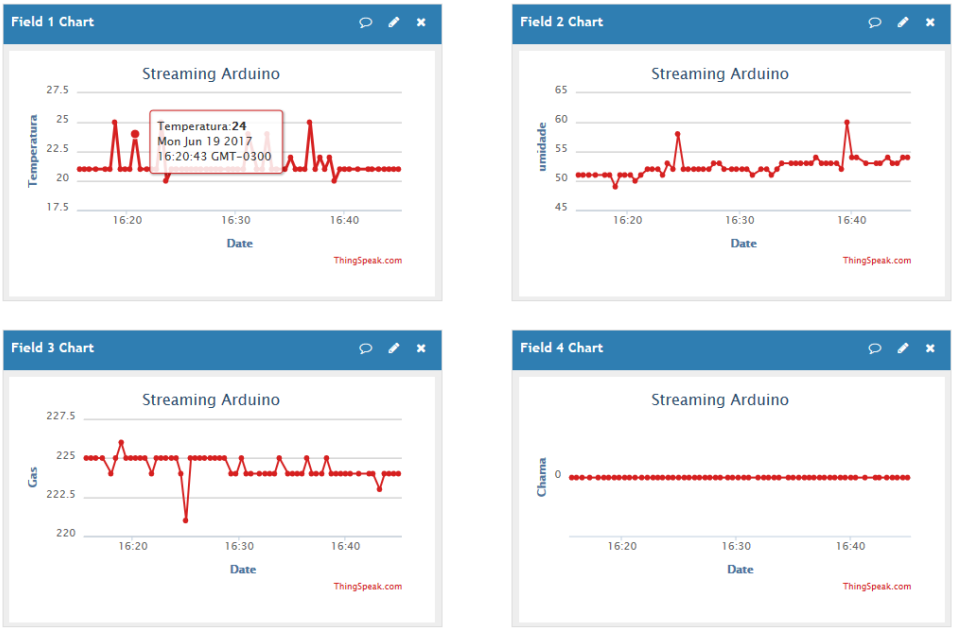

Figura 42 – Gráficos em Tempo Real, Praticamente

5.2 Análise histórica

A Figura 43 mostra a página inicial do relatório desenvolvido no software Power BI Desktop. Nessa página é possível ter uma visão geral do ambiente monitorado, onde é possível visualizar:

- Campo 1: De qual dispositivo os dados são provenientes.
- Campo 2: Número de ocorrências de incêndios registrados.
- Campo 3: Média da temperatura no período analisado.
- Campo 4: Temperatura máxima registrada no período analisado.
- Campo 5: Temperatura mínima registrada no período analisado.
- Campo 6: Média do indicativo de gás no período analisado.
- Campo 7: Número de amostras coletados.

- Campo 8: Evolução no tempo das amostras coletadas referentes à temperatura, umidade e indicativo de gás.
- Campo 9: Filtro para selecionar o intervalo de tempo em que se deseja analisar.

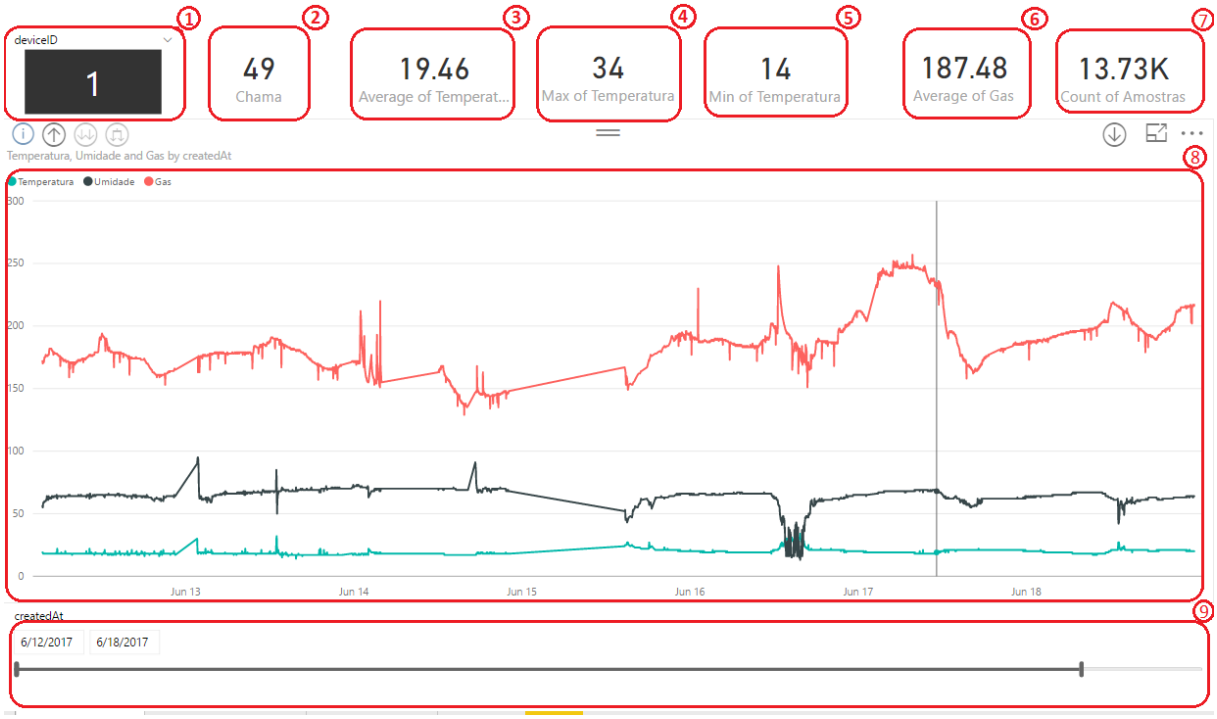

Figura 43 – Análise Histórica - Página Principal

Também fez-se análises mais detalhadas de cada uma das variáveis analisadas. As Figuras 44 e 45 mostram a análise feita para a variável temperatura.

Na Figura 44 pode-se observar:

- Campo 1: De qual dispositivo os dados são provenientes.
- Campo 2: Temperatura máxima registrada no período analisado.
- Campo 3: Temperatura mínima registrada no período analisado.
- Campo 4: Média da temperatura no período analisado.
- Campo 5: Variância da temperatura no período analisado.
- Campo 6: Desvio padrão da temperatura no período analisado.
- Campo 7: Evolução da temperatura no tempo no período analisado
- Campo 8: Filtro para selecionar o intervalo de tempo em que se deseja analisar.

Já na Figura 45 analisa-se a média da temperatura por hora por dia da semana, onde cada coluna representa um dia de semana e cada faixa um intervalo de uma hora.

Fez-se as mesmas análises para as variáveis de umidade relativa e indicativo de gás, que podem ser verificadas no Apêndice E.

Figura 44 – Análise Histórica – Temperatura Parte 1

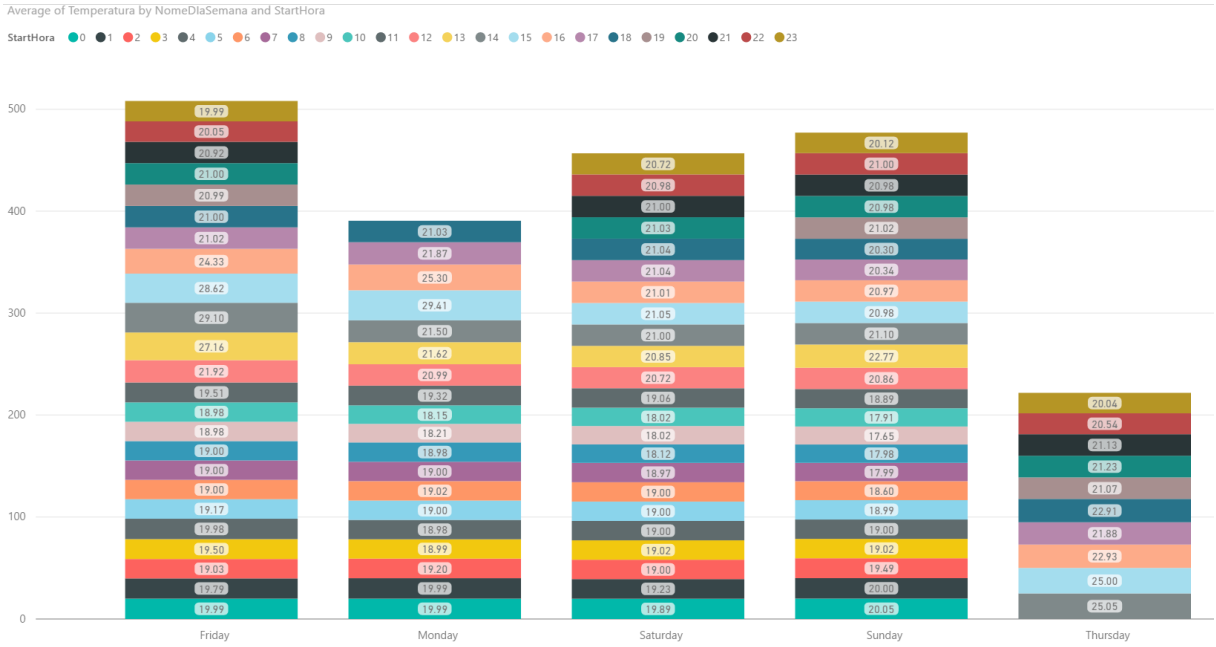

Figura 45 – Análise Histórica – Temperatura Parte 2

5.3 Alertas e notificações

Como previamente comentado, fez-se um sistema de alertas através do Twitter para manter o usuário sempre a par do ambiente monitorado. A figura 46 mostra um exemplo de um

alerta quando se detectou a presença de chamas. Já a Figura 47 apresenta um alerta de aumento de concentração de gás no ambiente.

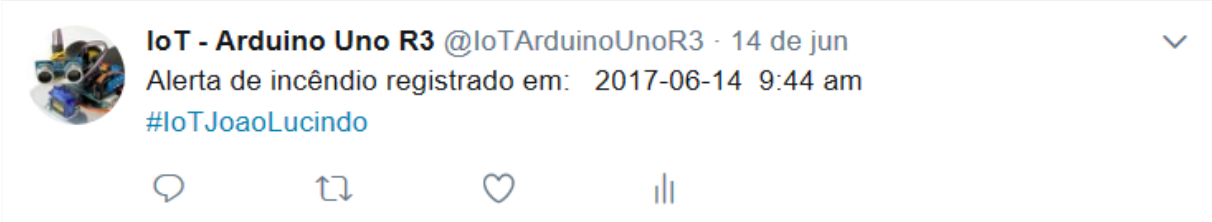

Figura 46 – Notificação de Incêndio Via Twitter

Figura 47 – Notificação de Concentração de Gás via Twitter

Os dois alertas supracitados são alertas críticos, todavia fez-se também utilizando os mesmos recursos alertas de notificação meramente que informar a umidade relativa e a temperatura do ambiente. Essas duas notificações são feitas a cada 30 minutos e são apresentadas nas figuras 48 e 49, respectivamente.

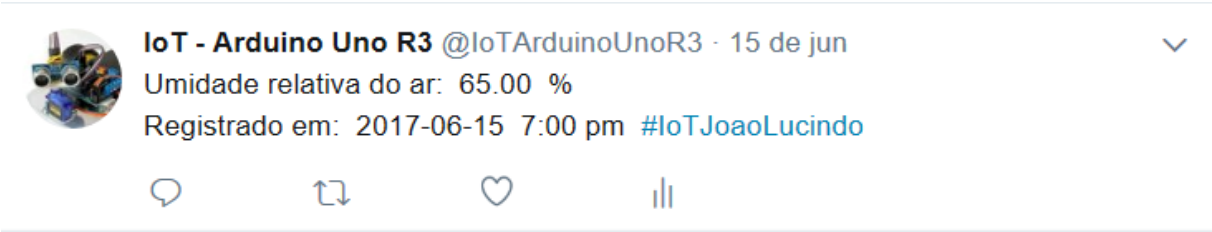

Figura 48 – Notificação Informativa de Umidade Relativa do Ar via Twitter

Figura 49 – Notificação Informativa de Temperatura via Twitter

Também se desenvolveu alertas de notificação para comunicar uma possível falha do sistema. Caso o sistema fique sem receber dados por cinco minutos, emite-se um alerta, conforme a Figura 50.

Figura 50 – Notificação de Possível Falha do Sistema via Twitter

Todos os alertas também podem ser visualizados e acompanhados através da página do Twitter do sistema, disponível no endereço: https://twitter.com/IoT ArduinoUnoR3.

5.4 Monitoração por dispositivos móveis

Todos as análises e alertas supracitados também podem ser monitorados por qualquer dispositivo móvel utilizando o aplicativo do Power BI e do Twitter. Dessa maneira, o usuário pode monitorar o ambiente e ser notificado de qualquer lugar e a qualquer momento.

Exemplos de alertas em dispositivo móvel podem ser vistos na Figura 51, onde nota-se que o horário de registro da notificação e do dispositivo móvel coincidem, ou seja, o usuário foi notificado em menos de um minuto após a ocorrência do evento.

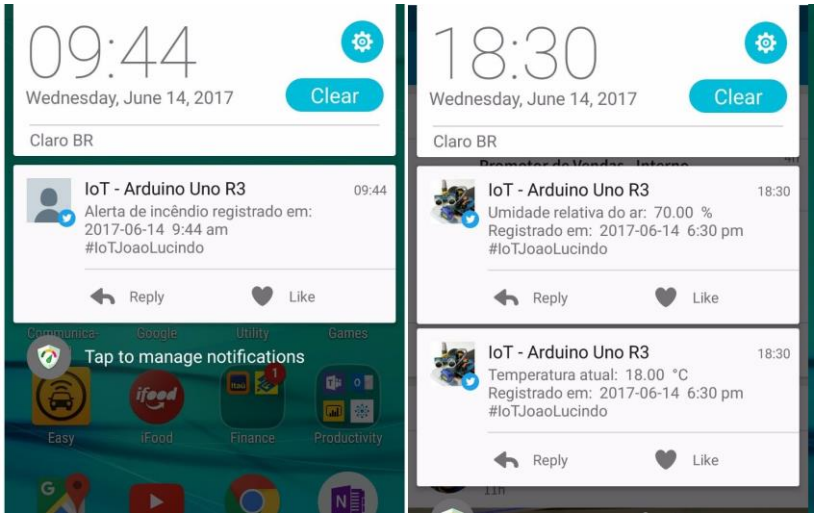

Figura 51 – Notificações via Twitter em um Dispositivo Móvel

Já a figura 52 mostra as duas análises desenvolvidas, a em praticamente tempo real e a histórica.

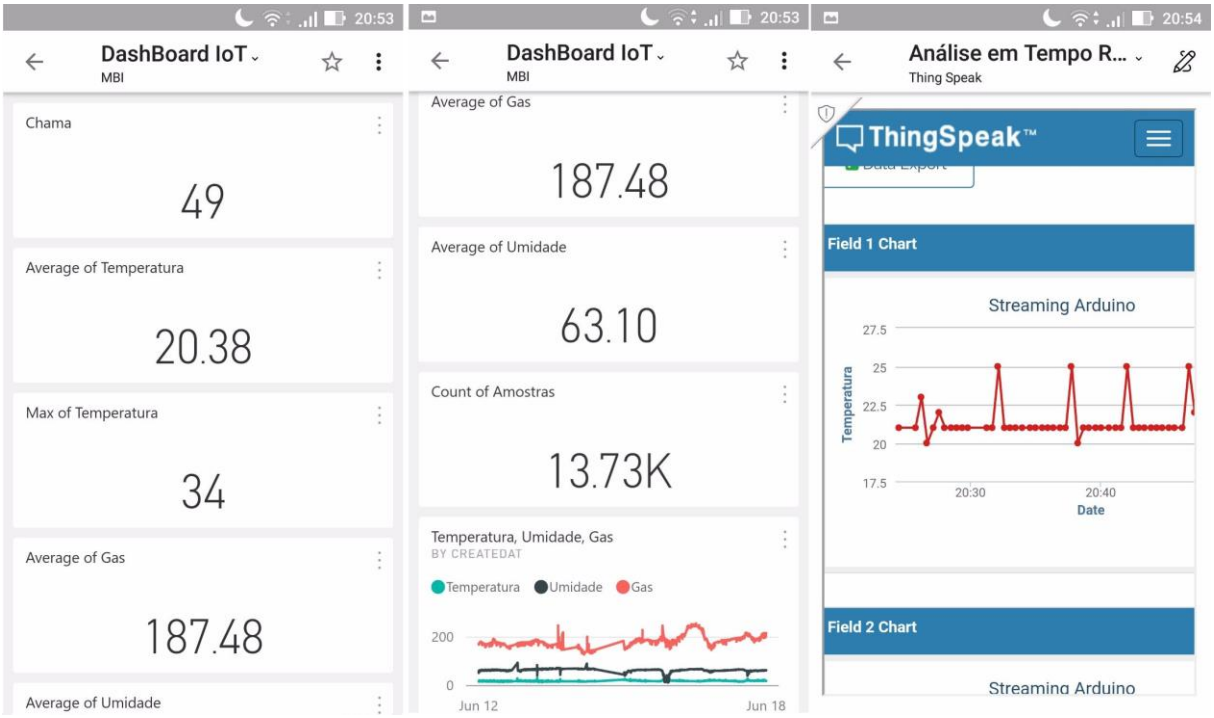

Figura 52 – Visualização das Análises em um Dispositivo Móvel

6 Conclusão

Através do estudo feito conclui-se que o sistema desenvolvido satisfez todas as expectativas, apresentando análise histórica, análise real time (praticamente), notificação em dispositivo mobile, bem como armazenamento em um banco de dados. Substancialmente obteve-se uma aplicação de IoT capaz de coletar, armazenar e exibir os dados coletados, além de ser capaz de notificar alterações críticas do ambiente.

Conjuntamente pôde-se inferir que as análises realizadas pelo sistema de monitoramento podem ser consultadas através de qualquer plataforma, por dispositivos mobile ou não. Ademais o sistema pode ser reaproveitado para aplicações em diversos cenários de coleta de dados como, por exemplo, na agricultura, setor florestal, industrial, dentre outros.

Conclui-se também que a organização do trabalho em forma de tutorial poderá auxiliar do desenvolvimento de outros sistemas IoT, simplesmente alterando os sensores empregados. Dessa forma, o trabalho é uma contribuição tecnológica que poderá ser replicada em outros diversos cenários supracitados.

Para melhoria do sistema, em um estudo futuro, sugere-se adicionar a funcionalidade de acionamento de cargas remotamente através da Internet, dessa maneira o sistema se tornará ativo, possibilitando não só monitorar, mas também agir e interagir com o ambiente.

Referências Bibliográficas

[1] ARDUINO. **What is Arduino?** Arduino Guide Introduction. Disponível em: <https://www. arduino.cc/en/GuiB42:E78de/Introduction>. Acesso em: 24 abr. 2017

[2] ARNETTE, Greg. **Cloud Investments & the Future of Cloud Computing**. 2016. Disponível em: <https://www.enterprisetech.com/2016/10/24/cloud-investments-future-cloud-comp uting/>. Acesso em: 10 fev. 2017

[3] ASHTON, Kevin. **That 'Internet of Things' Thing In the real world, things matter more than ideas**. RFID Journal, 22 jun. 2009. Disponível em: <http://www.rfidjournal.com/articles/view?4986>. Acesso em: 13 fev. 2017

[4] BIES, Lammert. **Hayes AT command set, the history**. Disponível em: <https://www.lammertbies.nl/comm/info/hayes-at-commands.html>. Acesso em: 23 abr. 2017

[5] CHEN, Edward T. **The Internet of Things: Opportunities, Issues, and Challenges**. University of Massachusetts, USA, 2017

[6] CISCO. **Global Cloud Index: Forecast and Methodology, 2015-2020**. Disponível em: <http://www.cisco.com/c/dam/en/us/solutions/collateral/service-provider/global-cloud-index-gci/white-paper-c11-738085.pdf>. Acesso em: 23 fev. 2017

[7] COLUMBUS, Louis. **Internet of Things Market To Reach $267B By 2020**. Forbes, 29 jan. 2017. Disponível em: <https://www.forbes.com/sites/louiscolumbus/2017/01/29/internet-of-things-market-to-reach-267b-by-2020/#5e4b0915609b>. Acesso em: 2 mar. 2017

[8] DORSEMAINE, Bruno et al. **Internet of Things a definition & taxonomy**. In: 2015 9th International Conference on Next Generation Mobile Applications, Services and Technologies. Paris, France, 2015.

[9] D-ROBOTICS. **Datasheet DHT11 Humidity & Temperature Sensor**. D-Robotics UK Sensor. Disponível em: <http://www.micropik.com/PDF/dht11.pdf>. Acesso em: 24 abr. 2017

[10] ESPRESSIF. **ESP8266EX Datasheet**. Espressif Documentation. Disponível em: <http://www.espressif.com/sites/default/files/documentation/0a-esp8266ex_datasheet_en.pdf>. Acesso em: 23 abr. 2017

[11] EVANS, Dave. **The Internet of Things: How the Next Evolution of the Internet Is Changing Everything. Cisco White Paper**, 2011. Disponível em: <http://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf>. Acesso em: 18 fev. 2017

[12] FARNELL. **Datasheet Arduino Uno**. Farnell Datasheets. Disponível em: <http://www.farnell.com/datasheets/1682209.pdf>. Acesso em: 24 abr. 2017

[13] FUT-ELETRONICS. **Datasheet Flame Sensor Module**. Future Electronics Egypt. Disponível em: <http://www.fut-electronics.com/wp-content/plugins/fe_downloads/Uploads/Flame-sensor-arduino.pdf>. Acesso em: 24 abr. 2017

[14] GARTNER. **Gartner Says by 2020 "Cloud Shift" Will Affect More Than $1 Trillion in IT Spending**. Gartner Press Release, 20 jul. 2016. Disponível em: <http://www.gartner.com/ newsroom/id/3384720>. Acesso em: 12 mar. 2017

[15] GARTNER. **Track Three Trends In The 2016 Gartner Hype Cycle For Emerging Technologies**. Disponível em: <https://www.forbes.com/sites/gartnergroup/2016/08/29/ track-three-trends-in-the-2016-gartner-hype-cycle-for-emerging-technologies/ #5e6c86115 e6c>. Acesso em: 1 mar. 2017

[16] GONZALEZ, Angel; DAY, Matt. **Amazon, Microsoft invest billions as computing shifts to cloud**. Seattle Times, 27 abr. 2016. Disponível em: <http://www.seattletimes.com/business /technology/amazon-microsoft-invest-billions-as-computing-shifts-to-cloud/>. Acesso em: 8 mar. 2017

[17] HALL, Adrian et al. **What is Mobile Apps?** Microsoft Azure Documentation. Disponível em: <https://docs.microsoft.com/en-us/azure/app-service-mobile/app-service-mobile-v alue-prop>. Acesso em: 9 mar. 2017

[18] HASSAN, Qusay. **Demystifying Cloud Computing**. Faculty of Computers and Information, Mansoura University, Egypt Fev. 2011. Disponível em: <https://www.research gate.net/publication/215795101_Demystifying_Cloud_Computing>. Acesso em: 15 mar. 2017

[19] HILTON, Steve. **IoT and Predictive Maintenance**. Bosch ConnectedWorld Blog, 15 fev. 2013. Disponível em: <http://blog.bosch-si.com/categories/manufacturing/2013/02/iot-and-predictive-maintenance/>. Acesso em: 3 mar. 2017

[20] INTERNACIONAL TELECOMMUNICATION UNION. **Overview of the Internet of things**. ITU-T Y.2060. Disponível em: <http://www.itu.int/rec/T-REC-Y.2060-201206-I>. Acesso em: 27 mar. 2017

[21] ISEMINGER, David. **Data sources in Power BI Desktop**. Power BI Documentation. Disponível em: <https://powerbi.microsoft.com/en-us/documentation/powerbi-desktop-data-sources/>. Acesso em: 11 mar. 2017

[22] LOPEZ RESEARCH. **An Introduction to the Internet of Things (IoT)**. 2013. Disponível em: <http://www.cisco.com/c/dam/en_us/solutions/trends/iot/introduction_to_IoT_november.pdf>. Acesso em: 13 fev. 2017

[23] MACMAHON, Adam; MILENKOVIC, Victor. **Social Volunteer Computing**. Journal of Systemics, Cybernetics and Informatics, Volume 9 - Number 4, 2011. Disponível em: <http://www.iiisci.org/journal/CV$/sci/pdfs/QN188DK.pdf>. Acesso em: 3 mar. 2017

[24] MACY, Marsh et al. **Azure Storage replication**. Microsoft Azure Documentation. Disponível em: <https://docs.microsoft.com/en-us/azure/storage/storage-redundancy>. Acesso em: 17 mar. 2017

[25] MATHWORKS. **ThingSpeak**. MathWorks Documentation. Disponível em:https://www.
mathworks.com/help/thingspeak/?requestedDomain=www.mathworks.com>. Acesso em: 10 mar. 2017

[26] MCCARTHY, John. **Reminiscences on the history of time sharing**. Stanford University 1983. Disponível em: <http://www-formal.stanford.edu/jmc/history/timesharing/time sharing.html >. Acesso em: 11 mar. 2017

[27] MELL, Peter; GRANCE, Timothy. **Recommendations of the National Institute of Standards and Technology - The NIST Definition of Cloud Computing**. Special Publication 800-145. 2011. Disponível em: <http://nvlpubs.nist.gov/nistpubs/Legacy/SP/nistspecial publication800-145.pdf>. Acesso em: 16 mar. 2017

[28] MICROSOFT. **Azure SQL Database Documentation**. Microsoft Azure Documentation. Disponível em: <https://docs.microsoft.com/en-us/azure/sql-database/>. Acesso em: 9 mar. 2017

[29] MICROSOFT. **Dartmouth-Hitchcock ushers in a new age of proactive, personalized healthcare using Cortana Analytics Suite**. Microsoft Transform Blog. 13 jul. 2015. Disponível em: <https://blogs.microsoft.com/transform/2015/07/13/dartmouth-hitchcock-ushers-in-a-new-age-of-proactive-personalized-healthcare-using-cortana-analytics-suite/#sm.00000qywcim2k7einzb69hiq2ed4v>. Acesso em: 5 mar. 2017

[30] MICROSOFT. **DAX basics in Power BI Desktop**. Power BI Documentation. Disponível em: <https://powerbi.microsoft.com/en-us/documentation/powerbi-desktop-quickstart-learn-dax-basics/>. Acesso em: 11 mar. 2017

[31] MICROSOFT. **Microsoft and Rolls-Royce collaborate to offer advanced operational intelligence to airlines**. Microsoft News Center, 24 abr. 2016. Disponível em: <https://news.microsoft.com/2016/04/24/microsoft-and-rolls-royce-collaborate-to-offer-advanced-operational-intelligence-to-airlines/#yQRTszUvpcuAgBV9.97>. Acesso em: 2 mar. 2017

[32] MICROSOFT. **Report Power BI Rolls Royce**. Disponível em: <https://powerbi.microsoft .com/en-us/industries/airline/>. Acesso em: 3 mar. 2017

[33] MICROSOFT. **The Trusted Cloud Most comprehensive compliance coverage of any cloud provider**. Microsoft Azure Documentation. Disponível em: <https://azure. microsoft.com/en-us/support/trust-center/>. Acesso em: 8 mar. 2017

[34] MICROSOFT. **What is cloud computing? A beginner's guide**. Microsoft Azure Documentation. Disponível em: <https://azure.microsoft.com/en-us/overview/what-is-cloud-computing/>. Acesso em: 10 mar. 2017

[35] MICROSOFT. **What is PaaS? Platform as a service**. Microsoft Azure Documentation. Disponível em: <https://azure.microsoft.com/en-us/overview/what-is-paas/>. Acesso em: 10 mar. 2017

[36] MICROSOFT. **What is Power BI? Power BI Documentation**. Disponível em: <https://po werbi.microsoft.com/en-us/>. Acesso em: 11 mar. 2017

[37] MIKKERS, André. **Data are the oil of the future**. Actueel PWC, 30 out. 2015. Disponível em:<https://actueel.pwc.nl/en/diensten-en-sectoren/energie-en-utilities/data-zijn-de-olie-van-de-toekomst/>. Acesso em: 17 fev. 2017

[38] PARALLAX. **Datasheet MQ-05.** Parallax Datasheets. Disponível em: <https://www. parallax.com/sites/default/files/downloads/605-00009-MQ-5-Datasheet.pdf>. Acesso em: 24 abr. 2017

[39] ROOM15. **ESP8266 - AT Command Reference**. Room15 GitHub 26 mar. 2015. Disponível em: <https://room-15.github.io/blog/2015/03/26/esp8266-at-command-reference/>. Acesso em: 23 abr. 2017

[40] SCHWAB, Klaus. **The Fourth Industrial Revolution: What it means, how to respond**. World Economic Forum, 14 jan. 2016. Disponível em: https://www.weforum.org/agenda/ 2016/01/the-fourth-industrial-revolution-what-it-means-and-how-to-respond/. Acesso em 24 maio. 2017

[41] SCHWARTZ, Marco. **Internet of Things with ESP8266**. Packt Publishing. 2016. 226p

[42] SPIEß, Patrik; KARNOUSKOS, Stamatis. **Maximizing the Business Value of Networked Embedded Systems through Process-Level Integration into Enterprise Software**. In: 2007 2nd International Conference on Pervasive Computing and Applications. Birmingham, United Kingdom, 2007

[43] TWITTER. **Getting started with Twitter**. Twitter Help Center. Disponível em: <https://support.twitter.com/articles/215585?lang=en>. Acesso em: 17 abr. 2017

[44] TWITTER. **Twitter Developer Documentation**. Twitter Developers. Disponível em: <https://dev.twitter.com/overview/api>. Acesso em: 17 abr. 2017

[45] UKESSAY. **The History of The Arduino Microcontroller**. UK Essays, 23 mar. 2015. Disponível em: <https://www.ukessays.com/essays/information-technology/the-history-of-the-arduino-microcontroller-information-technology-essay.php>. Acesso em: 24 abr. 2017

[46] VERMESAN, O. et al. **Internet of things strategic research roadmap. 2009**. Disponível em: <https://www.researchgate.net/publication/267566519_Internet_of_Things_Strategic Research_Roadmap>. Acesso em 13 fev.2017

[47] ZUSAMMEN, Hallo. **IaaS, PaaS, and SaaS - Pizza as a Service Example**. Blog Ice Wolf, 12 ago. 2014. Disponível em: <http://blog.icewolf.ch/archive/2014/08/16/iaas-paas-and-saas-pizza-as-a-service-example.aspx>. Acesso em: 7 mar. 2017

Apêndice A - Implementação do Web App Azure e SQL Database Azure

Nessa etapa do processo de criação da solução, criou-se os serviços responsáveis por armazenar todos os dados enviados pelo Arduino. Para tal, entrou-se no portal do Azure (www.portal.azure.com). Depois de acessar o portal clicou-se no símbolo "+" no canto superior esquerdo, conforme indicado na Figura 53.

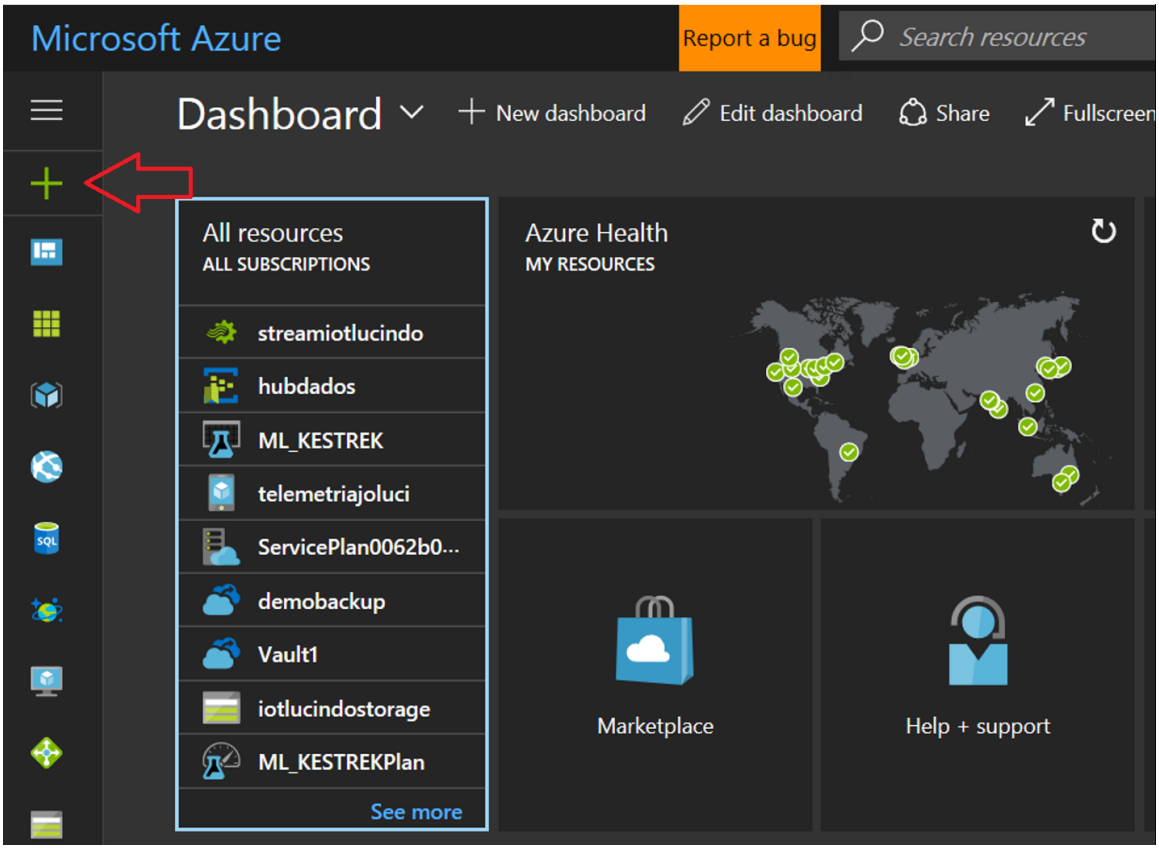

Figura 53 – Portal Azure – Criação de um Novo Serviço

Então uma lista de serviços disponíveis será exibida. Escolheu-se pelo item "Mobile Apps" e clicou-se no botão "Create" para começar a configuração do serviço. Em seguida, deu-se um nome "iotmonografia" para a aplicação e novamente clicou-se no botão "Create" para dar o início ao provisionamento do serviço em nuvem, conforme a Figura 54.

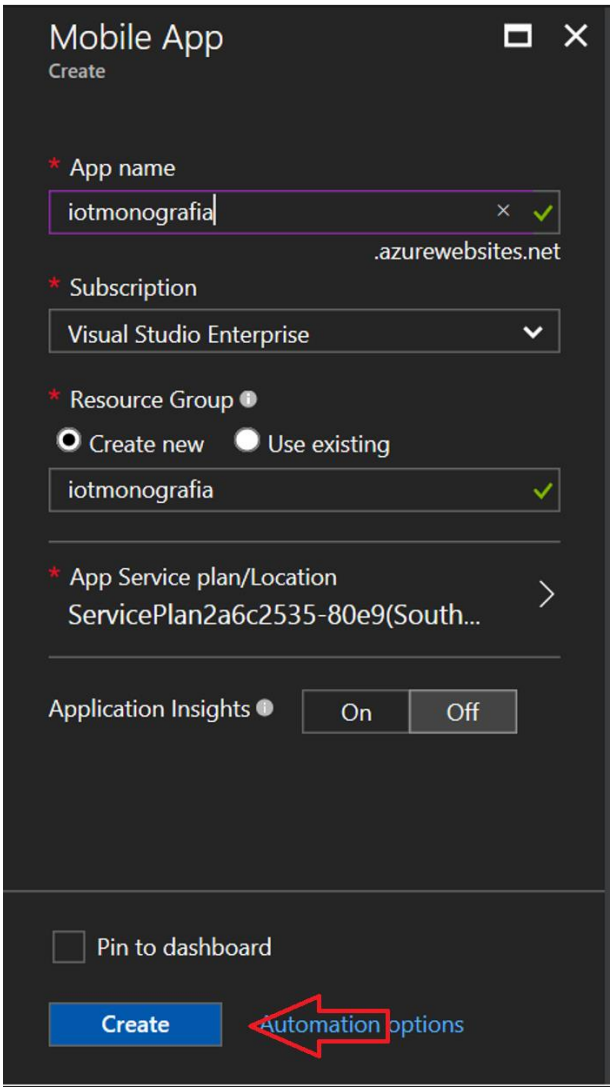

Figura 54 – Provisionamento Web App Azure

Em poucos segundos o serviço já estava disponível. Na página principal so serviço, a opção "Overview" traz as principais informações do Web App Azure recém-criado, e pode ser vista na Figura 55. Nessa figura vale destacar a URL do serviço (http://iotmonografia.azurewebsites.net) que será utilizada nas próximas etapas.

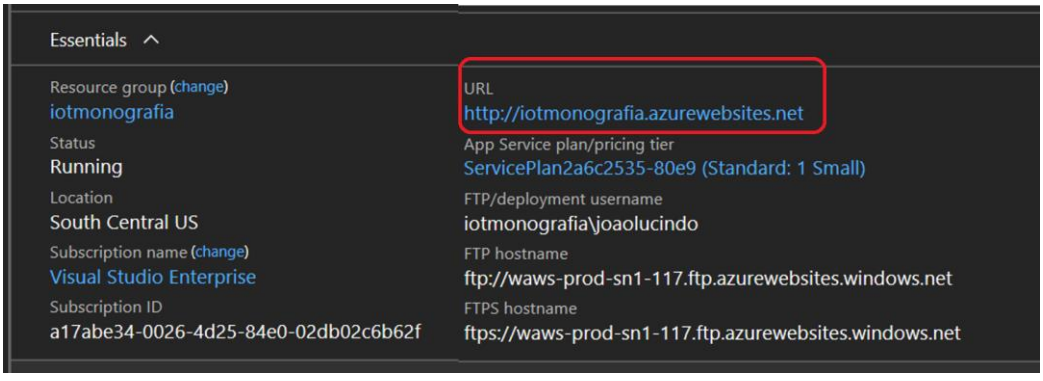

Figura 55 – Principais Informações Web App Azure

No menu do serviço, escolheu-se o item "EasyTable" e clicou-se no local indicado pela seta na Figura 56 para fazer a configuração. Esse componente do serviço chamado "EasyTable" é o item responsável por fazer a ponte entre o Web App Azure e o SQL Database Azure.

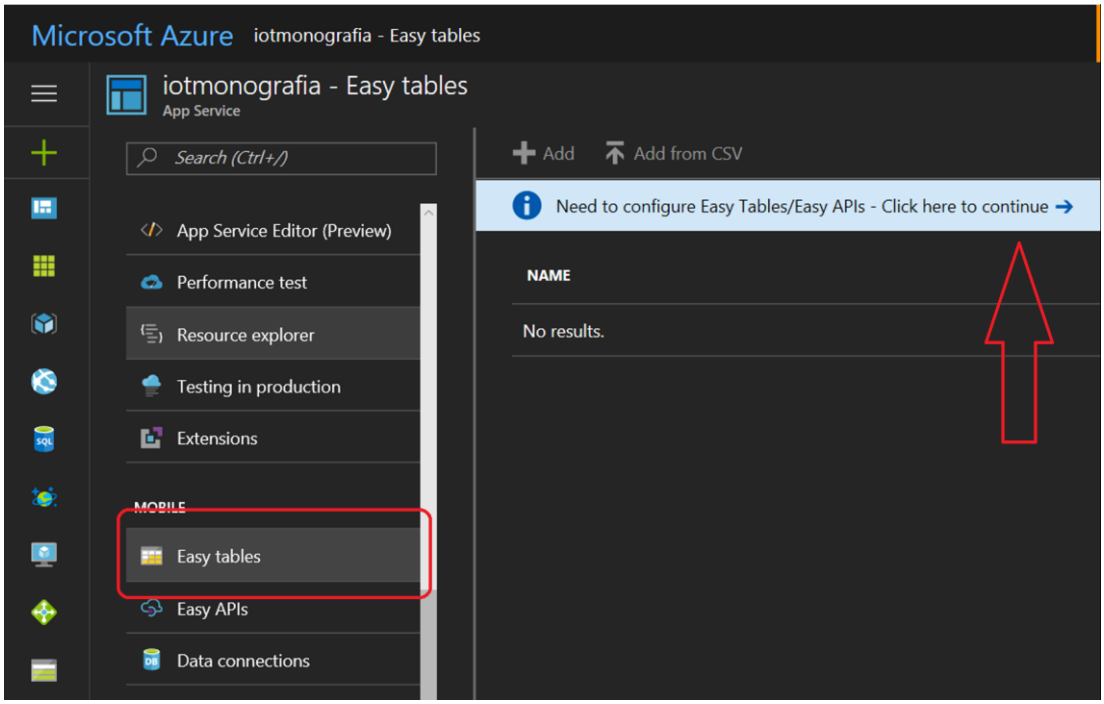

Figura 56 – Configuração Easytable

Todavia, para armazenar os dados é preciso provisionar um banco para essa função. Para tal, como próxima etapa, clicou-se em "Click here to create one" conforme indicado pela Figura 57.

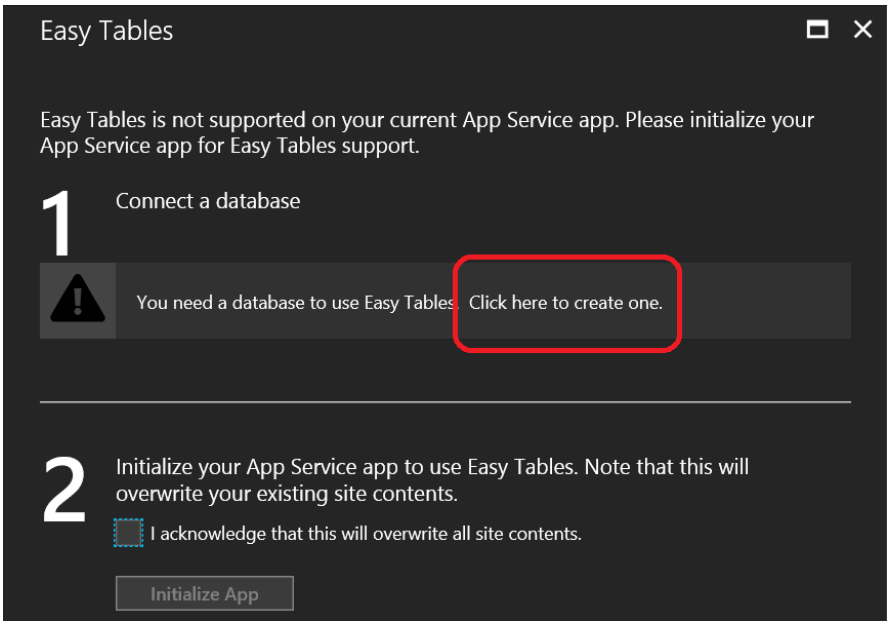

Figura 57 – Provisionamento de um Banco de Dados

Então clicou-se em "Add" e e no campo "Type" escolheu-se SQL Database. Nesse ponto precisou-se provisionar o banco de dados escolhido, o SQL Database. Para tal, clicou-se em "Configure required settings", conforme a Figura 58.

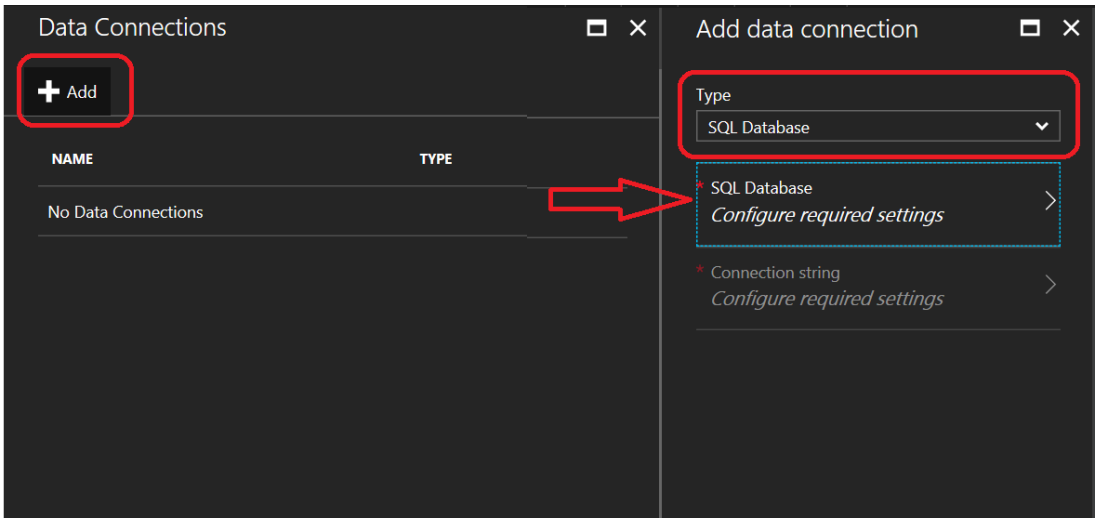

Figura 58 – Configuração Banco de Dados SQL

Nesse ponto, o portal Azure requer as configurações básicas do SQL Database. Deu-se um nome para o banco "iotdatabase" e em seguido clicou-se em "Configure required settings" para configurar o servidor que suportará o SQL Database.

Configurou-se o servidor conforme a Figura 60, informando o nome do servidor, o nome do usuário administrador, uma senha para o administrador e a localização do Datacenter em que esse serviço foi provisionado. Em seguida clicou-se em "Select", então em poucos segundos o servidor já estava pronto para receber o banco de dados.

Logo após, fez-se a última configuração do SQL Database: "Pricing tier". Esse parâmetro configura a performance do banco de dados, quanto mais performático e maior espaço de armazenamento, mais caro é o serviço. Todavia, para o escopo desse trabalho, escolheu-se o banco de dados gratuito com apenas 32MB de capacidade de armazenando e 5 DTU (Database Throughput Unit) de processamento, conforme a Figura 61. Após a escolha de performance do banco de dados, clicou-se em "Select".

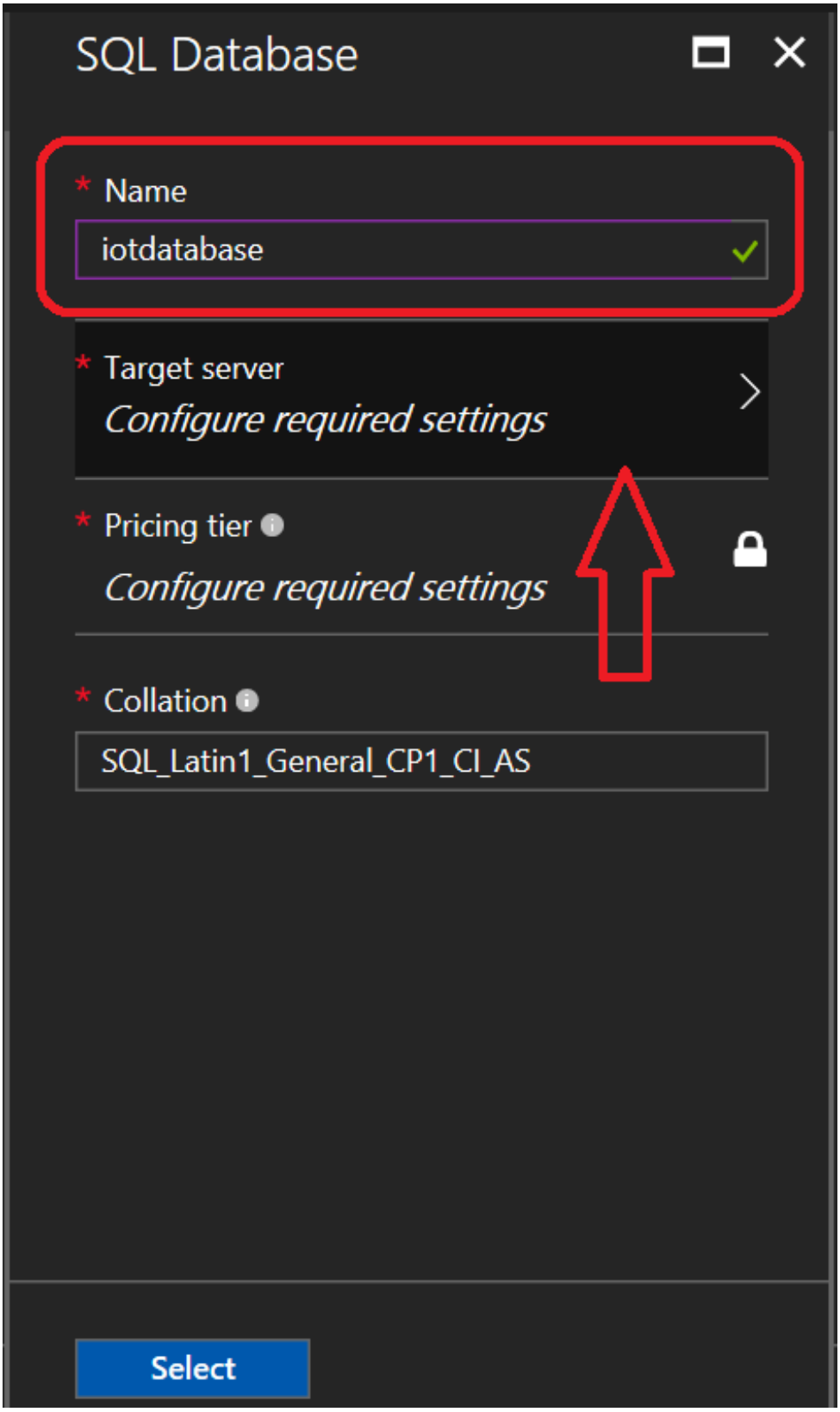

Figura 59 – Preparação para Configuração Servidor SQL

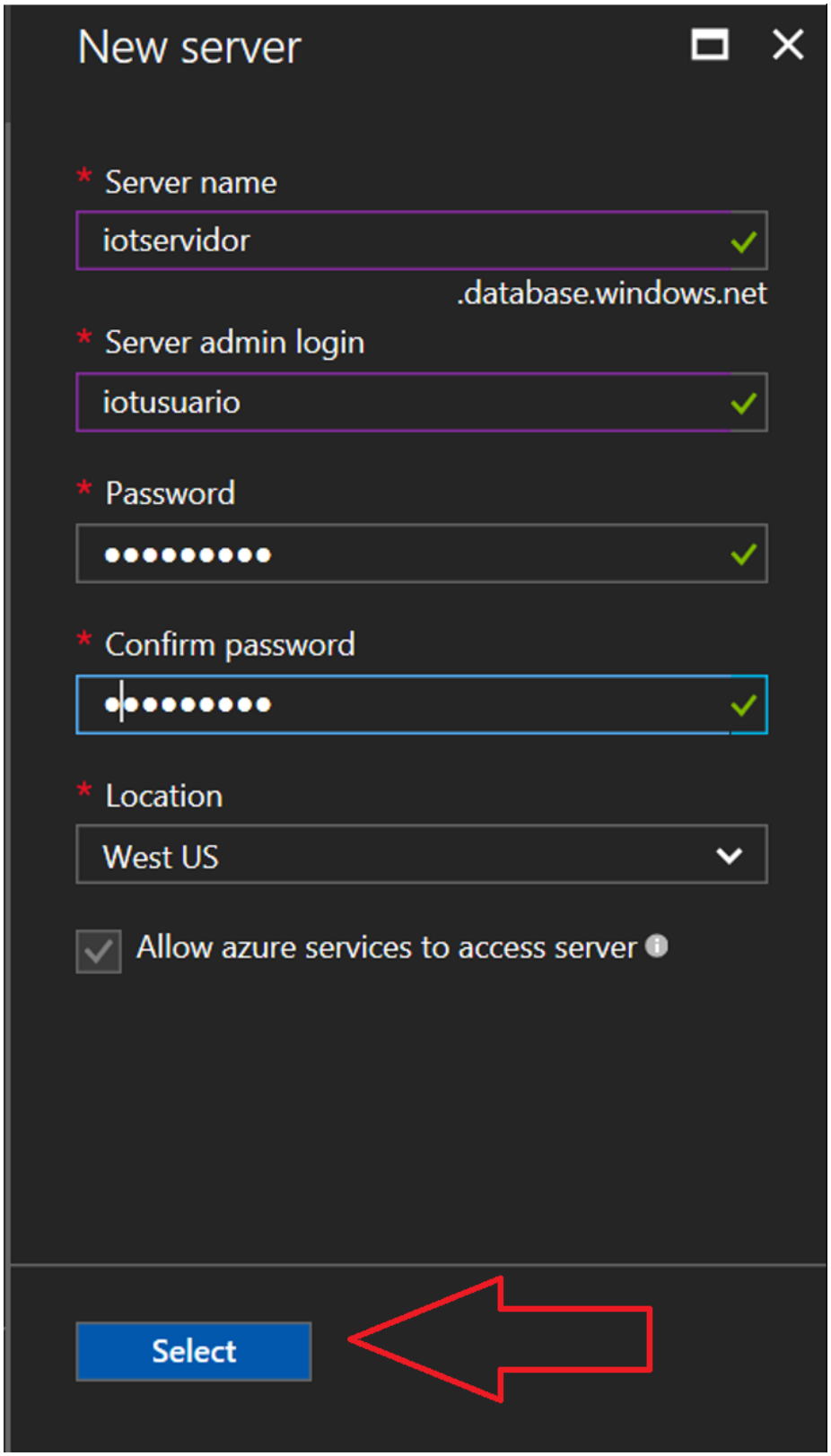

Figura 60 – Configuração Servidor SQL

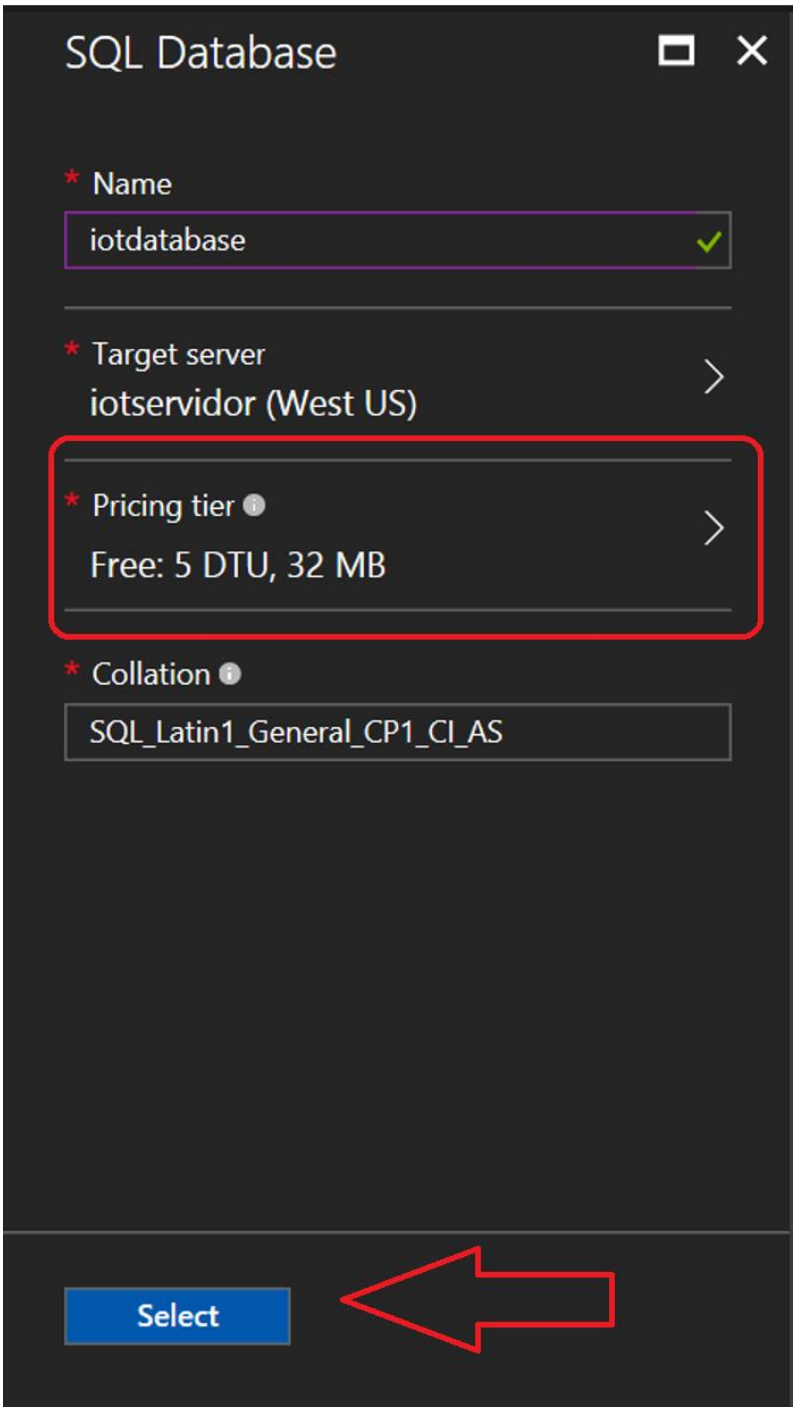

Figura 61 – Escolha de Performance do Banco de Dados SQL

Após essas etapas a conexão do Web App Azure com o SQL Database Azure está consolidada. Entretanto, para habilitar o serviço "Easy Tables" ainda é necessário iniciá-lo, para tal clicou-se em "Initialize App", conforme a Figura 62.

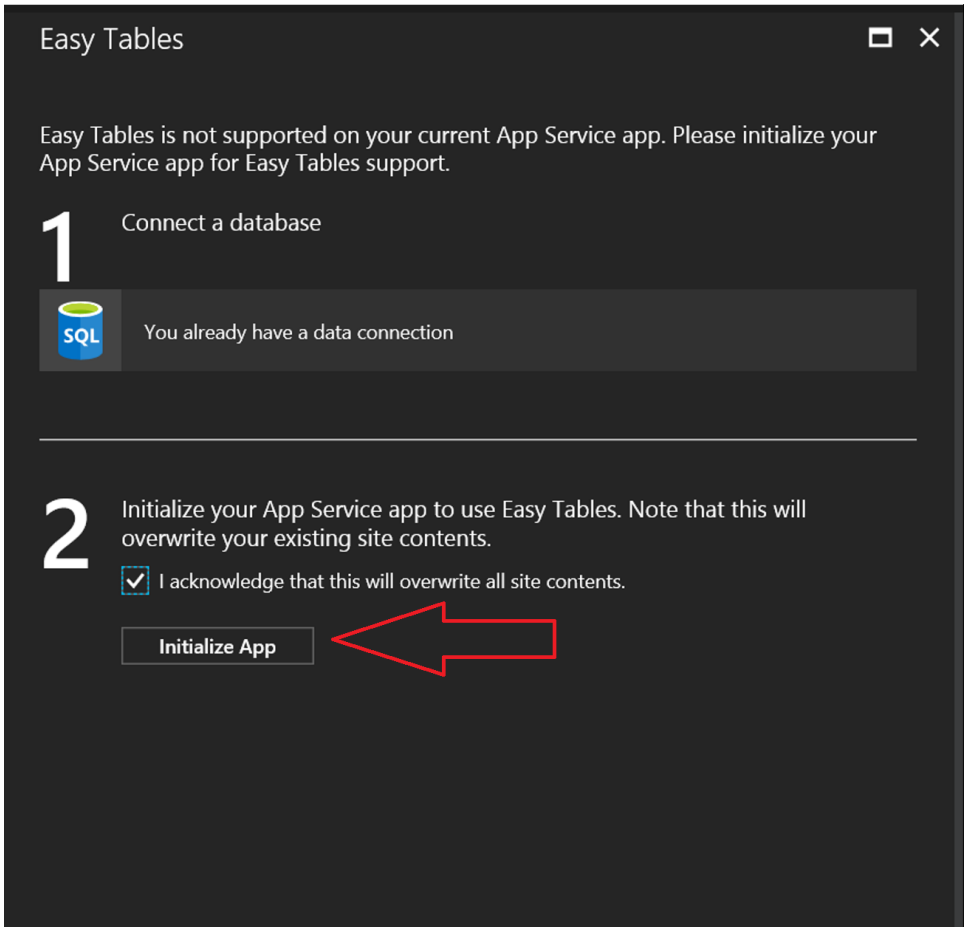

Figura 62 – Inicialização Easytable

Após inicializar a aplicação, criou-se a tabela para receber os dados provenientes do Arduino, para isso clicou-se em "Add". Então fez-se as configurações da "Easy Table" conforme a Figura 63, dando o nome "iottabela" e alterando as permissões de todas as ações para "Authenticated access only", dessa forma somente dispositivos autenticados conseguirão inserir, deletar, atualizar, recuperar ou consultar dados na tabela.

Então, nesse ponto configurou-se a tabela "iottabela" propriamente dita, inserindo as colunas dos dados que se deseja coletar. Para tal, clicou-se no nome da tabela recém-criada, e em seguida em "Manage Schema".

Em seguida, adicionou-se cinco colunas para receber os valores de temperatura, umidade, concentração de poluentes, e grau de emissão de luz infravermelha e a identificação do dispositivo. Tendo como resultado uma tabela com dez colunas, conforme ilustra a Figura 64.

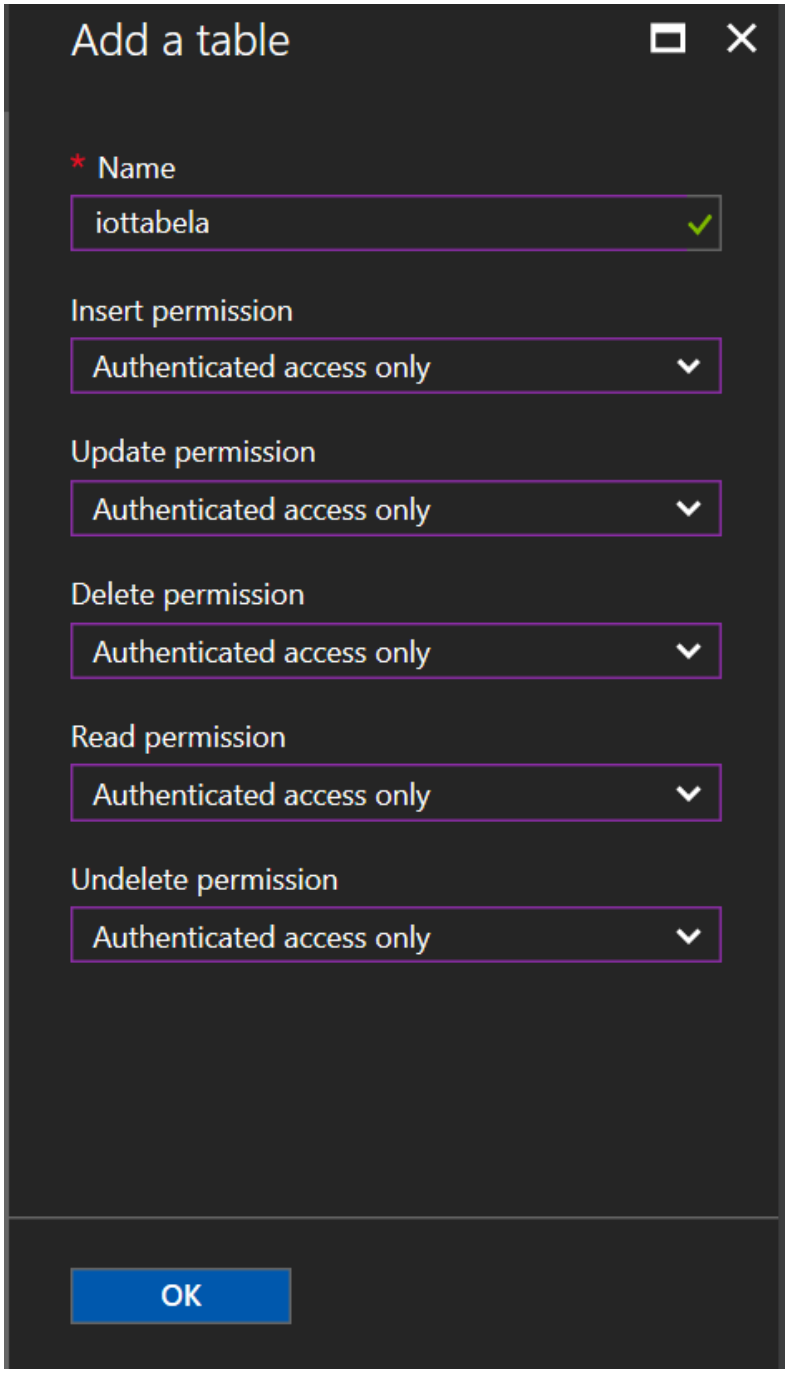

Figura 63 – Elaboração da Tabela Easytable

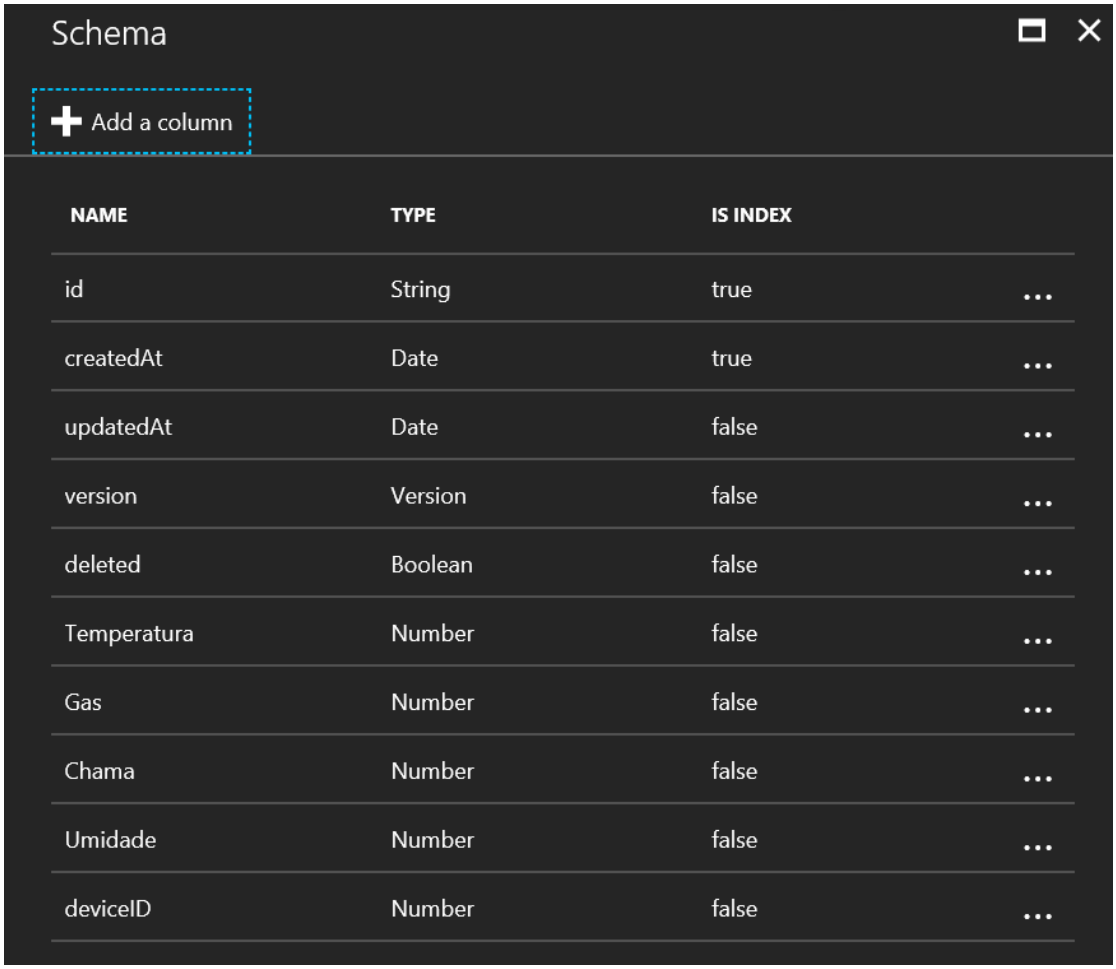

Figura 64 – Resultado Final da Tabela Easytable

Apêndice B - Conexão SQL Azure Database e Power BI Desktop

Para fazer a conexão do Power BI Desktop com SQL Azure Database, basta seguir os passos descritos.

- Passo 1: Faça o download do Power BI Desktop e em seguida instale-o
- Passo 2: Com o Power BI aberto, na barra de menu em "Get Data" clique em SQL Server, conforme a Figura 65

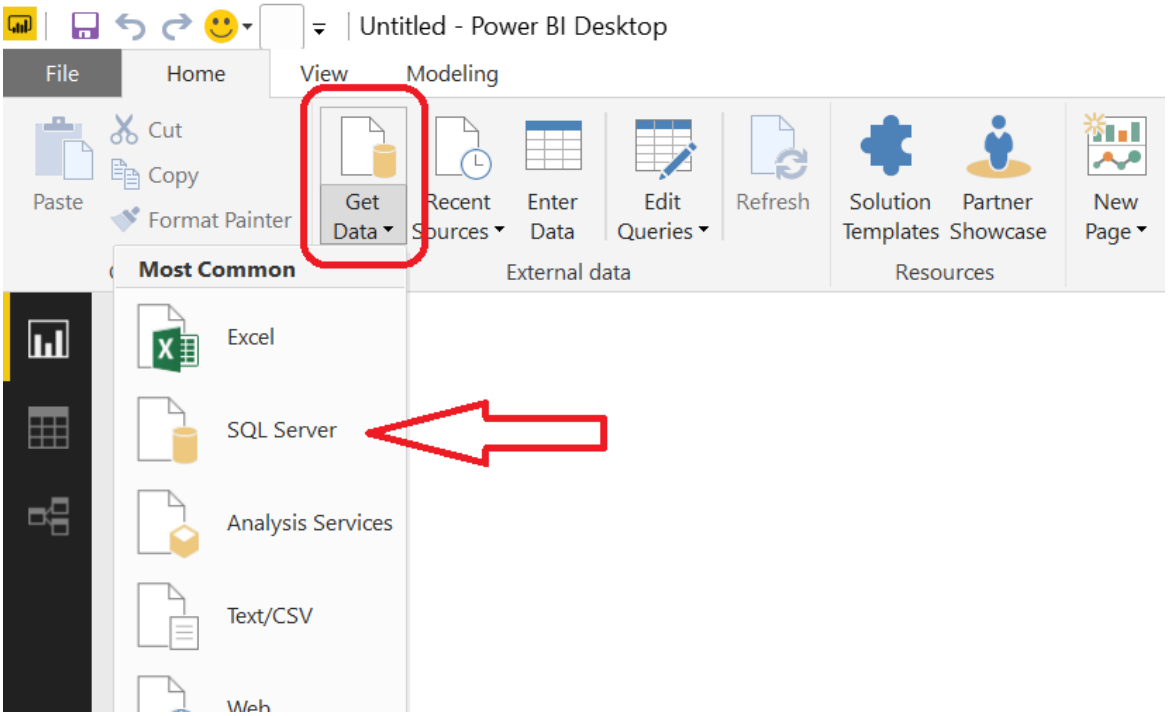

Figura 65 – Conexão Power BI Desktop com Banco de Dados SQL

- Passo 3: Entre com o nome do Servidor SQL e o nome do banco de dados, e clique em "OK", conforme a Figura 66

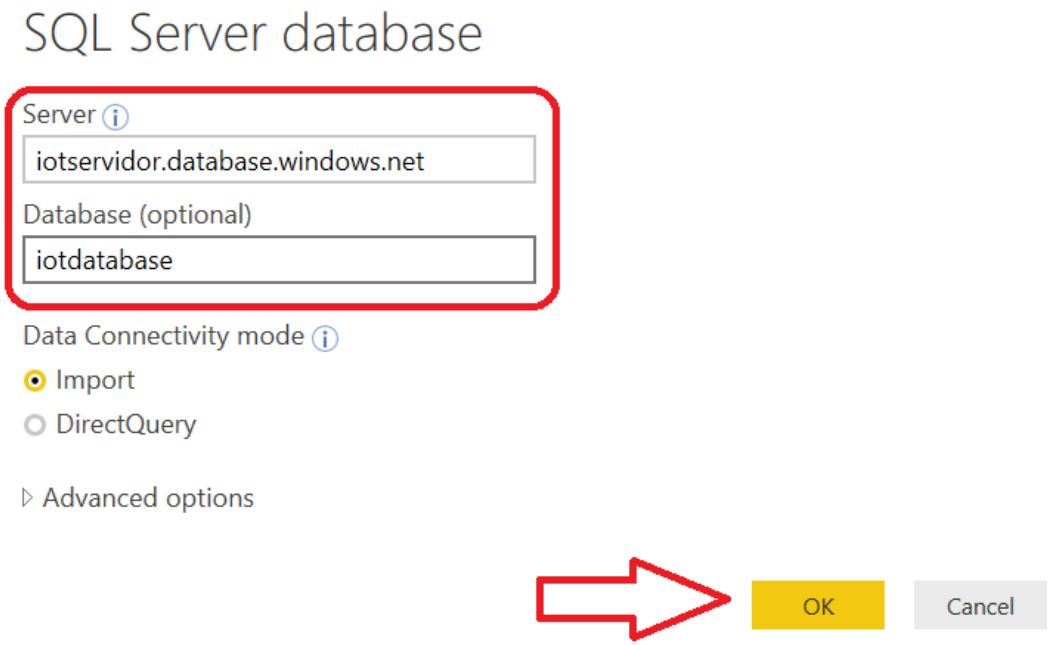

Figura 66 – Inserção das Credenciais SQL

- Passo 4: Analogamente, entre com as credenciais de usuário e clique em "OK".
- Passo 5: Selecione a tabela do banco de dados que deseja manipular e clique em "Load", conforme a Figura 67.

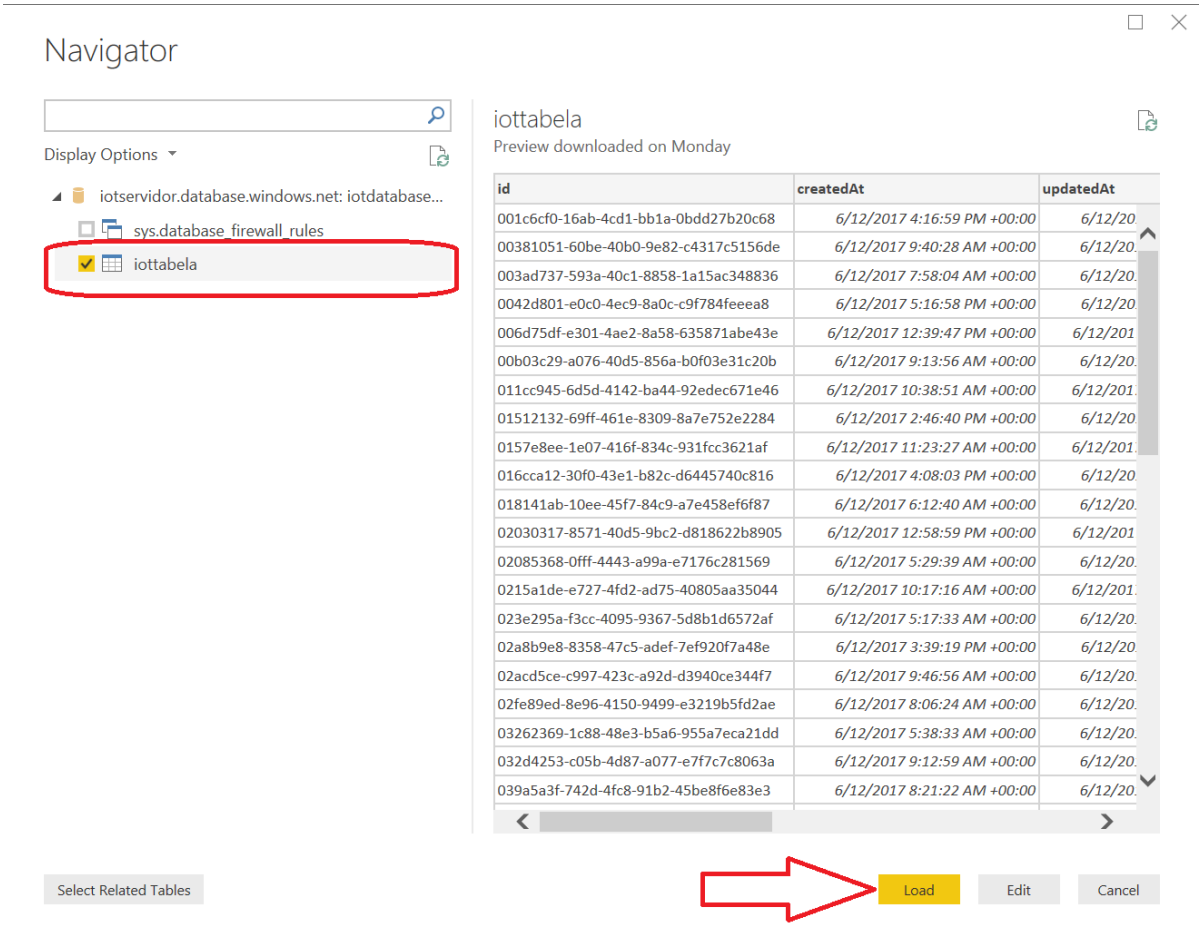

Figura 67 – Escolha da Tabela do SQL

Assim, está estabelecida e conexão com entre o Power BI Desktop e o SQL Azure Database. Pode-se então trabalhar com os dados disponíveis no SQL Azure Database e construir relatórios conforme a necessidade.

Apêndice C - Conexão Thingspeak e Twitter

Para gerar alertas e notificações utilizando a plataforma ThingSpeak, basta conectar a plataforma com a rede social Twitter. Para tal, basta seguir os passos descritos.

- Passo 1: O primeiro passo é criar uma conta na rede social Twitter, para tal basta acessar o site principal dessa rede social, inserir informações básicas e clicar em "Sign up for Twitter", conforme a Figura 68.

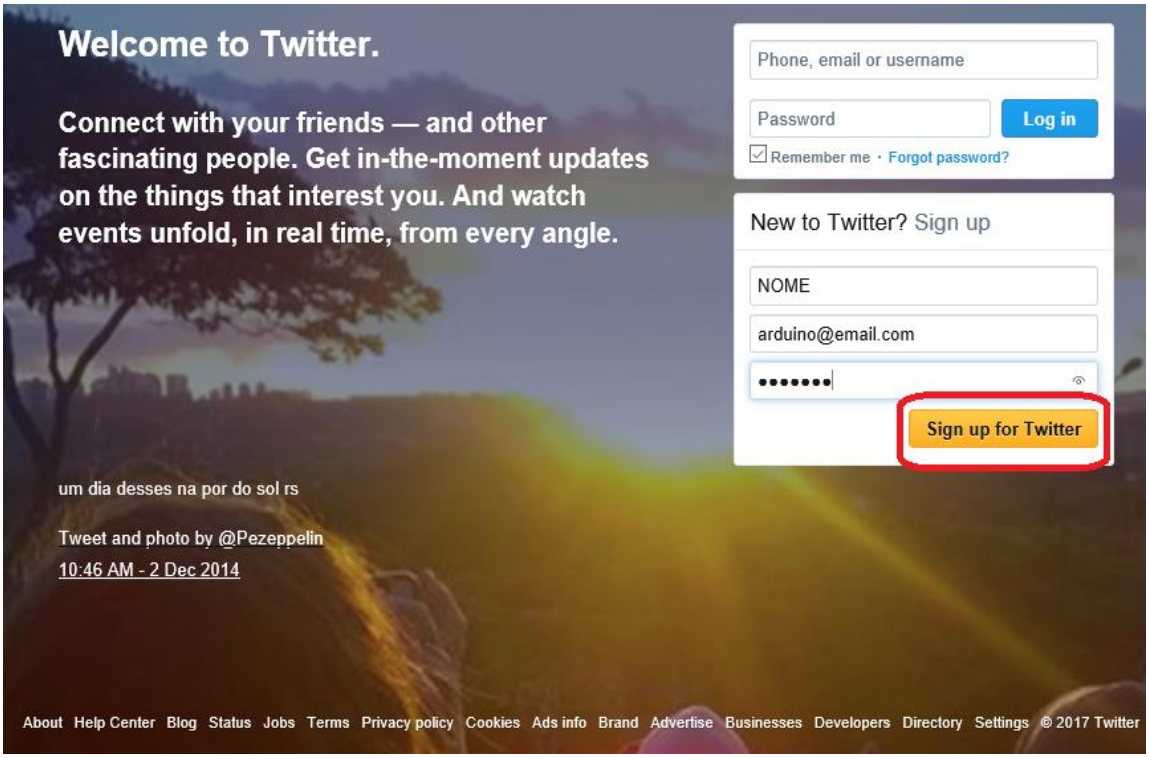

Figura 68 – Site Principal do Twitter

- Passo 2: O próximo passo é conectar a conta Twitter à plataforma ThingSpeak. Para tal, basta acessar a página principal da plataforma, dentro do menu "Apps" escolha a opção "ThingTweet", conforme a Figura 69.
- Passo 3: Em seguida, clique em "Link Twitter Account", conforme a Figura 70 e forneça o usuário e senha da conta Twitter recém-criada. Nesse ponto, tem-se as contas ThingSpeak e Twitter atreladas.
- Passo 4: No menu "Apps" do portal ThingSpeak, clique em "React", conforme a Figura 71. E em seguida, na próxima página clique em "New React".

- Passo 5: Uma tela de configuração ficará disponível, então basta fornecer o nome em "React Name", escolher o tipo de variável que será analisada em "Condition Type", escolher a frequência de verificação em "Test Frequency", apontar em qual canal os dados estão disponíveis e estabelecer a lógica em "Condition". No campo Action, deve-se selecionar a opção “ThingTweet" e em seguida fornecer o texto que se deseja publicar caso a lógica estabelecida em "Condition" seja verdadeira. Como últimas configurações deve-se escolher qual conta Twitter deseja-se utilizar e selecionar um dos campos em "Options". A primeira opção de "Options" limita a publicação dos Tweets somente a primeira vez que a lógica é verdadeira, já a segunda faz uma nova publicação toda vez que a lógica for verdadeira. A Figura 72, mostra a configuração feita para notificar a temperatura do ambiente.

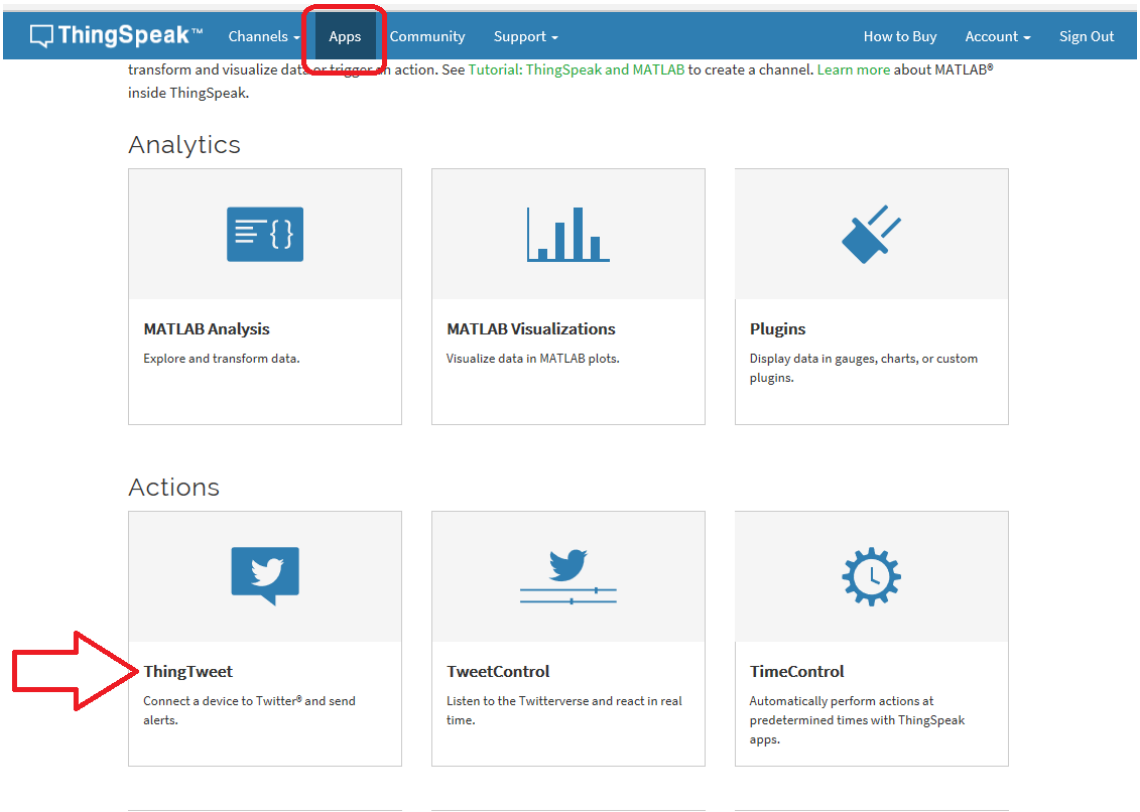

Figura 69 – Serviço Thingtweet

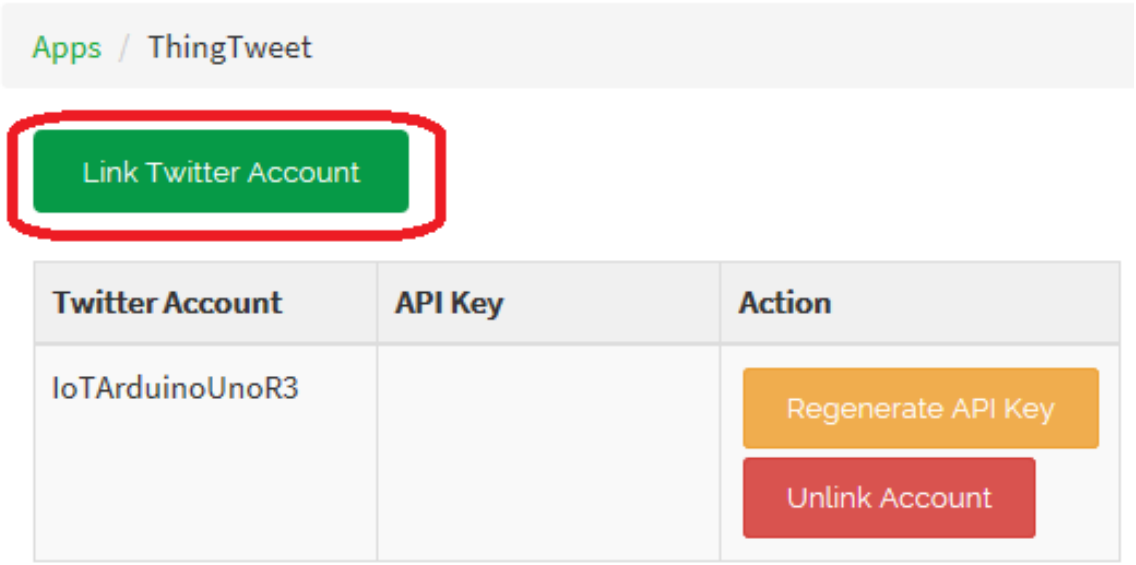

Figura 70 – Conexão Twitter com a Plataforma Thingspeak

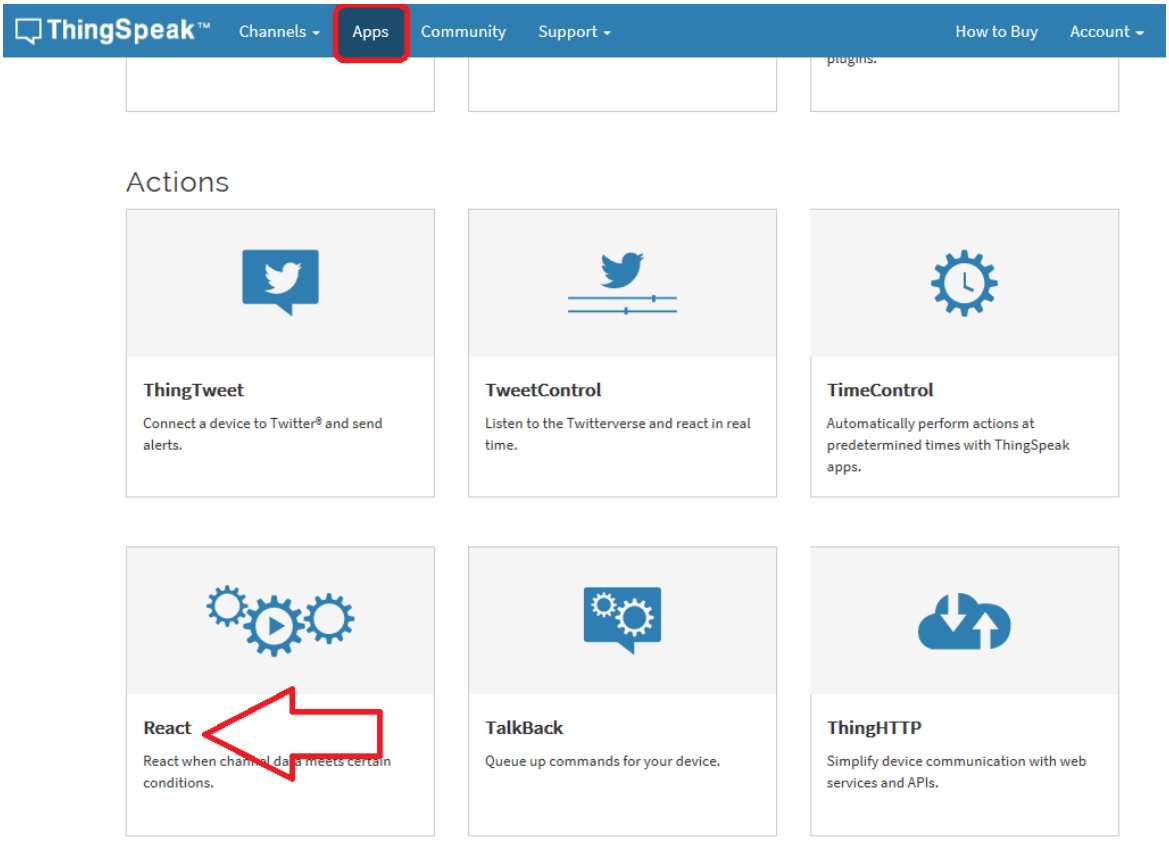

Figura 71 – Serviço React

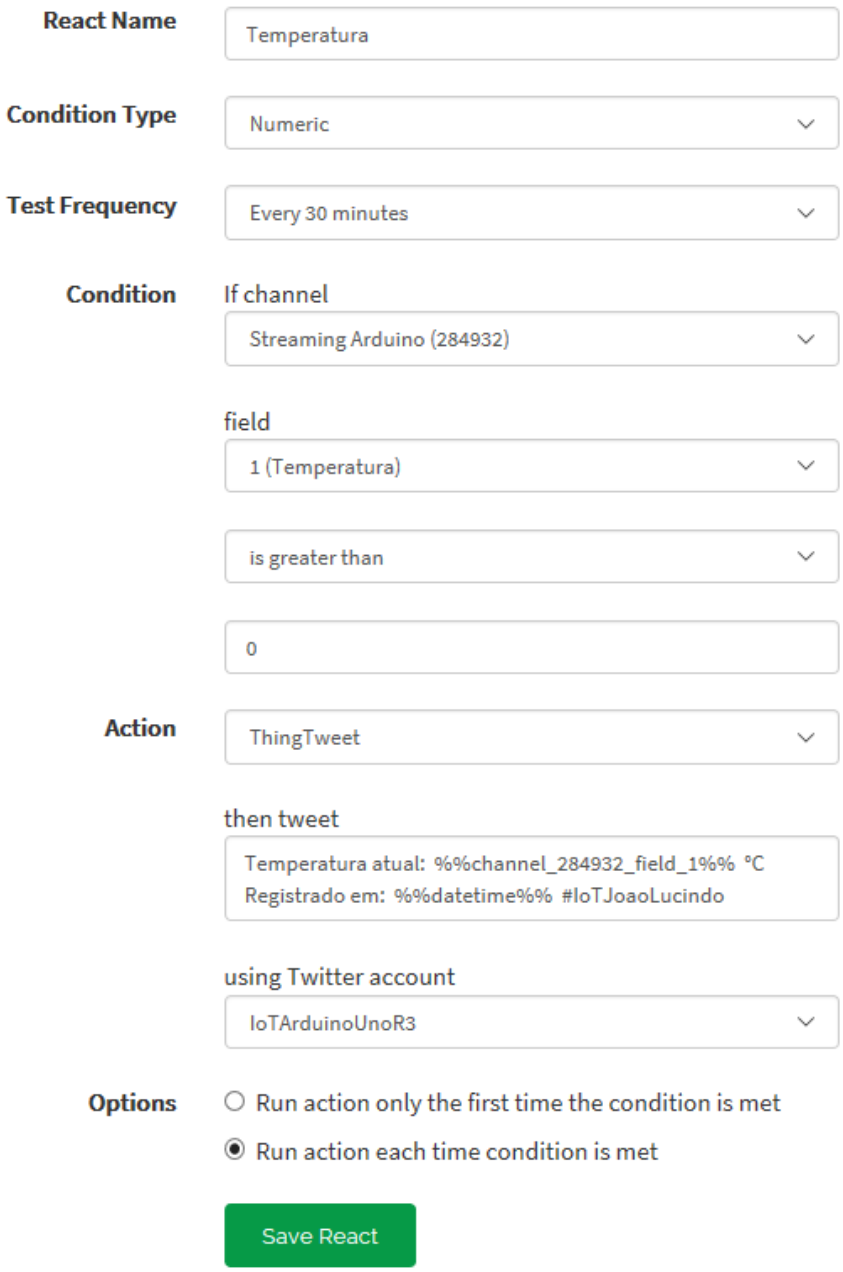

Figura 72 – Configuração Alerta de Temperatura

Apêndice D - Código Completo Desenvolvido

```
#include <DHT_U.h>
#include <DHT.h>
#include <SoftwareSerial.h>

#define REDE "SSID"
#define SENHA "SENHA"
#define IP "184.106.153.149" // thingspeak.com
#define HOST "iotmonografia.azurewebsites.net"
#define TIMEOUT 5000 // mS
#define TIMEOUT2 15000 // mS
#define DHTPIN    7        // Pino sensor temperatura e umidade
#define DHTTYPE  DHT11     // DHT 11

SoftwareSerial PonteWifiSerial(2, 3);

String GET = "GET /update?key=5BIPPLT00QWTCJ4S&field1=";

DHT dht(DHTPIN, DHTTYPE);

int PINDEBUG =12;      // pino led debug

void setup() {
   dht.begin();
   pinMode(PINDEBUG,OUTPUT);
   Serial.begin(9600);
   PonteWifiSerial.begin(115200); //  ESP baudrate

   if(EnviaComando("AT+RST","OK")){
   Serial.println("ESP8266-01 Resetado");
   delay(1000);
   }else { Serial.println("Falha ao Resetar");}
   delay(1000);

   PreparaESP8266();

   ConectaWifi();
 }
 void PreparaESP8266() {
   if(EnviaComando("AT","OK")){
   Serial.println("ESP8266-01 Respondendo");
   delay(500);
   }else{ Serial.println("ESP8266-01 não está respondendo");
   EnviaComando("AT+RST\r\n","OK");
   }
```

```
  if(EnviaComando("AT+CWMODE=3","OK")){
  Serial.println("Modo Cliente Ativado");
  delay(500);}
  if(EnviaComando("AT+CIPMUX=1","OK")){
  Serial.println("Modo múltiplas conexões Ativado");
  delay(500);
  }
}
void ConectaWifi() {
  String cmd = "AT+CWJAP=\"";
  cmd += REDE;
  cmd += "\",\"";
  cmd += SENHA;
  cmd += "\"";
  cmd += "\r\n";
  if(EnviaComando(cmd,"OK")){
  delay(500);
  Serial.println("Conectado à rede Wifi com sucesso");
  }
}

void EnviaThingSpeak(float tempF, float humP,float gas,int chama, int id){
String cmd = "AT+CIPSTART=4,\"TCP\",\"";
cmd += IP;
cmd += "\",80";
if(EnviaComando(cmd,"")){
    delay(100);
  }

if(PonteWifiSerial.find("Error")){
  Serial.println("Not good!");
  return;
}
cmd = GET;
cmd += tempF;
cmd += "&field2=";
cmd += humP;
cmd += "&field3=";
cmd += gas;
cmd += "&field4=";
cmd += chama;
cmd += "&field5=";
cmd += id;
cmd += "\r\n";
PonteWifiSerial.print("AT+CIPSEND=4,");
PonteWifiSerial.println("100");
delay(1000);
if(PonteWifiSerial.find(">")){
  Serial.println("sending command... ");
```

```
  PonteWifiSerial.print(cmd);
  Serial.println("Temperatura: ");
  Serial.println(tempF);

  Serial.println("Humidade: ");
  Serial.println(humP);

  Serial.println("Gas: ");
  Serial.println(gas);

  Serial.println("Chama: ");
  Serial.println(chama);

  Serial.println("ID: ");
  Serial.println(id);



delay(500);
PonteWifiSerial.println("AT+CIPCLOSE=4");
PonteWifiSerial.println();
delay(1000);


return;
}
else{
   PonteWifiSerial.println("AT+CIPCLOSE=4");
   PonteWifiSerial.println();

   return;
}
}

void loop()
{
float vu = dht.readHumidity();
float vt = dht.readTemperature();
int vgas = analogRead(0);
float chama2 = analogRead(2);
int bin_chama=1;
delay(1000);

if(chama2 < 500){
int bin_chama = 1;
} else {bin_chama = 0;}
digitalWrite(PINDEBUG,HIGH);
EnviaWebAppAzure(vt, vu, vgas,bin_chama, 1);
delay(1000);
```

```
 EnviaThingSpeak(vt, vu, vgas,bin_chama, 1);
 digitalWrite(PINDEBUG,LOW);
 delay(1000);

}


 boolean EnviaComando (String cmd, String ack) {
 PonteWifiSerial.println(cmd); // Envia "AT+"
 if (!EscutaEco(ack)) // tempo esgotado
   return true; // ack nao encontrado
 }

 boolean EscutaEco(String keyword) {
 byte current_char = 0;
 byte keyword_length = keyword.length();
 long deadline = millis() + TIMEOUT;
 while (millis() < deadline) {
   if (PonteWifiSerial.available()) {
   char ch = PonteWifiSerial.read();
   Serial.write(ch);
   if (ch == keyword[current_char])
     if (++current_char == keyword_length) {
     Serial.println();
     return true;
     }

 }
 return false; // Tempo esgotado
 }
 }


 void EnviaWebAppAzure(int tempF, int humP,int gas,int chama, int id){

 String cmd ="AT+CIPSTART=1,\"TCP\",\"";
 cmd += HOST;
 cmd += "\",80";

 Serial.println();
 PonteWifiSerial.println(cmd);

 delay(1000);
 if(PonteWifiSerial.find("Error")){
   Serial.println("Error");
   return;
 }

 PonteWifiSerial.print("AT+CIPSEND=1,");
 PonteWifiSerial.println("250");    // abre o canal de comunicação para enviar 250 bytes
```

```
  delay(1000);

  char buffer[64];

  if(PonteWifiSerial.find(">")){

   // POST URI
    PonteWifiSerial.println("POST /tables/iottabela HTTP/1.1");
    Serial.println("POST /tables/iottabela HTTP/1.1");

   // Host header
    PonteWifiSerial.println("Host: iotmonografia.azurewebsites.net");
    Serial.println("Host: iotmonografia.azurewebsites.net");
    // ZUMO
    PonteWifiSerial.println("ZUMO-API-VERSION: 2.0.0");
    Serial.println("ZUMO-API-VERSION: 2.0.0");
  //Content-Type:
    PonteWifiSerial.println("Content-Type: application/json");
    Serial.println("Content-Type: application/json");
  //POST body
      sprintf(buffer,
"{\"Temperatura\":\"%d\",\"Gas\":\"%d\",\"Chama\":\"%d\",\"Umidade\":\"%d\",\"deviceID
\":\"%d\"}\r\n\r\n", tempF, gas, chama, humP, id);
  //Content-:Length:
    PonteWifiSerial.print("Content-Length: ");
    PonteWifiSerial.println(strlen(buffer));
    Serial.print("Content-Length: ");
    Serial.println(strlen(buffer));

   PonteWifiSerial.println();
   Serial.println();
   PonteWifiSerial.println(buffer);
   Serial.println(buffer);

   PonteWifiSerial.print("Content-Length: ");
   PonteWifiSerial.println(strlen(buffer));



    delay(500);

  if(EnviaComando("AT+CIPCLOSE=1","OK")){   // fecha a comunicação TCP

    delay(1000);

    }else{ Serial.println("Erro Fechar canal");}
```

```
delay(500);
return setup();
}
else{
   PonteWifiSerial.println("AT+CIPCLOSE=1");
  PonteWifiSerial.println("AT+CIPCLOSE=1");  // fecha a comunicação TCP

delay(500);
return;
}
}
```

Apêndice E - Relatórios de Análises Históricas

Desenvolveu-se o relatório de análise histórica em quatro páginas, sendo elas a Análise principal, Análise de Temperatura, Análise de Umidade, Análise de Indicativo de gás. Cada uma dessas análises pode ser verificada nas Figuras de 73 a 79. Detalhes de cada um dos campos foram discutidos na seção de Resultados.

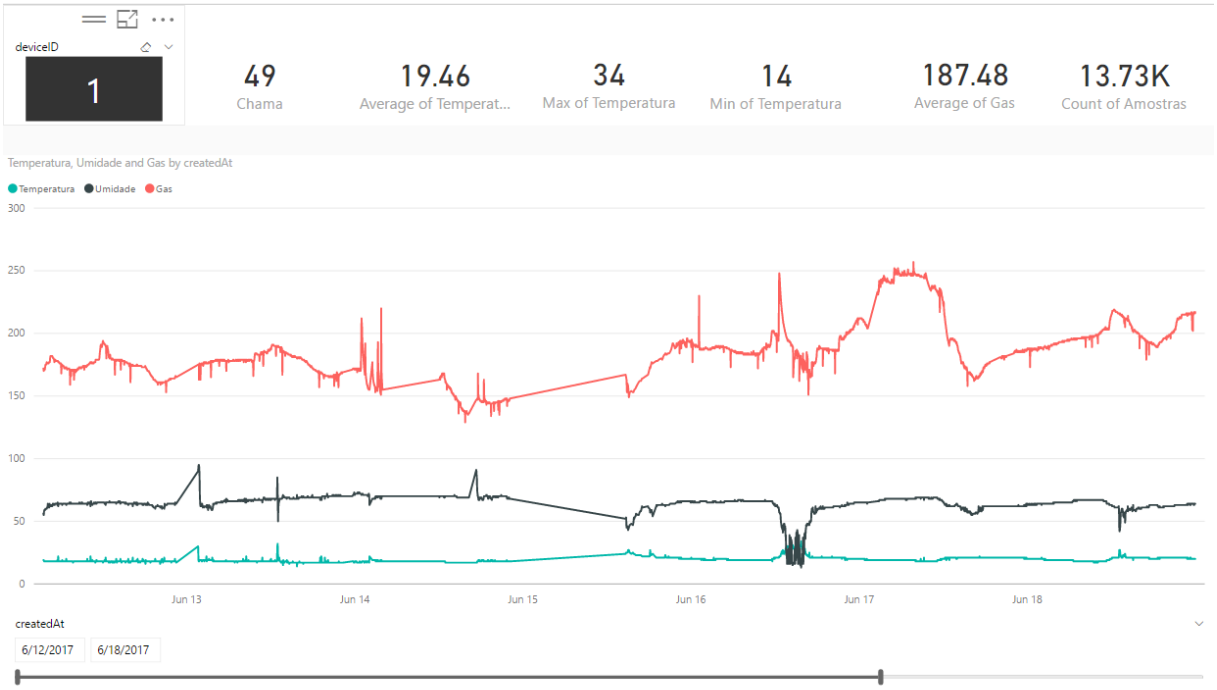

Figura 73 – Análise Principal Histórica

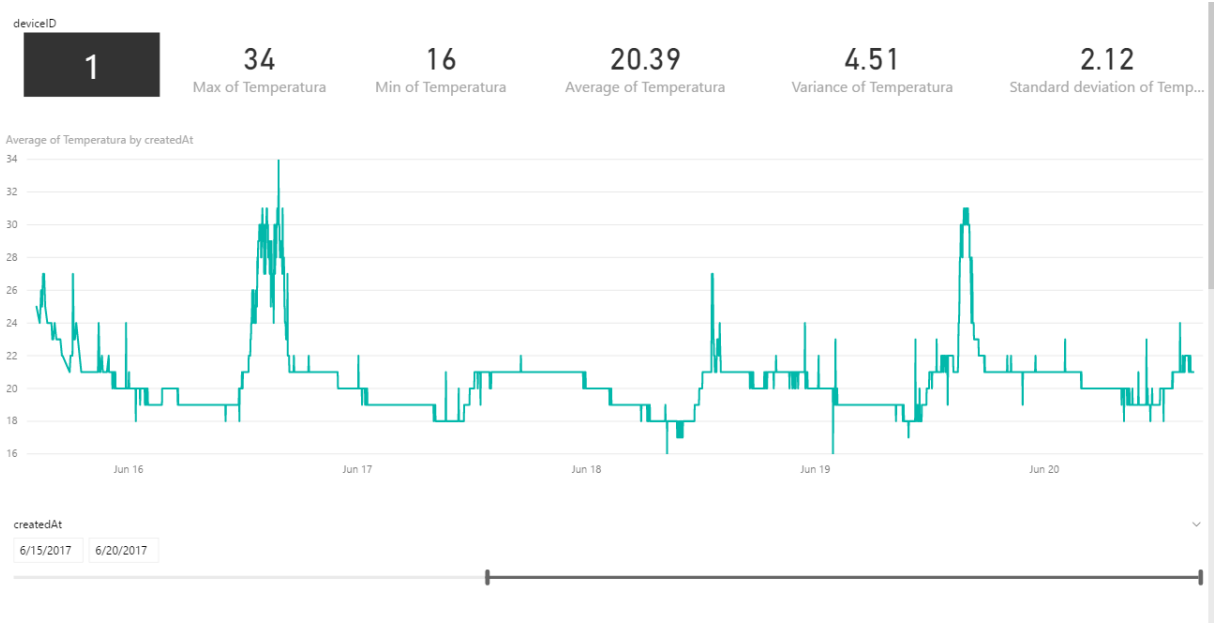

Figura 74 – Análise Temperatura Histórica – Primeira Parte

Figura 75 – Análise Temperatura Histórica – Segunda Parte

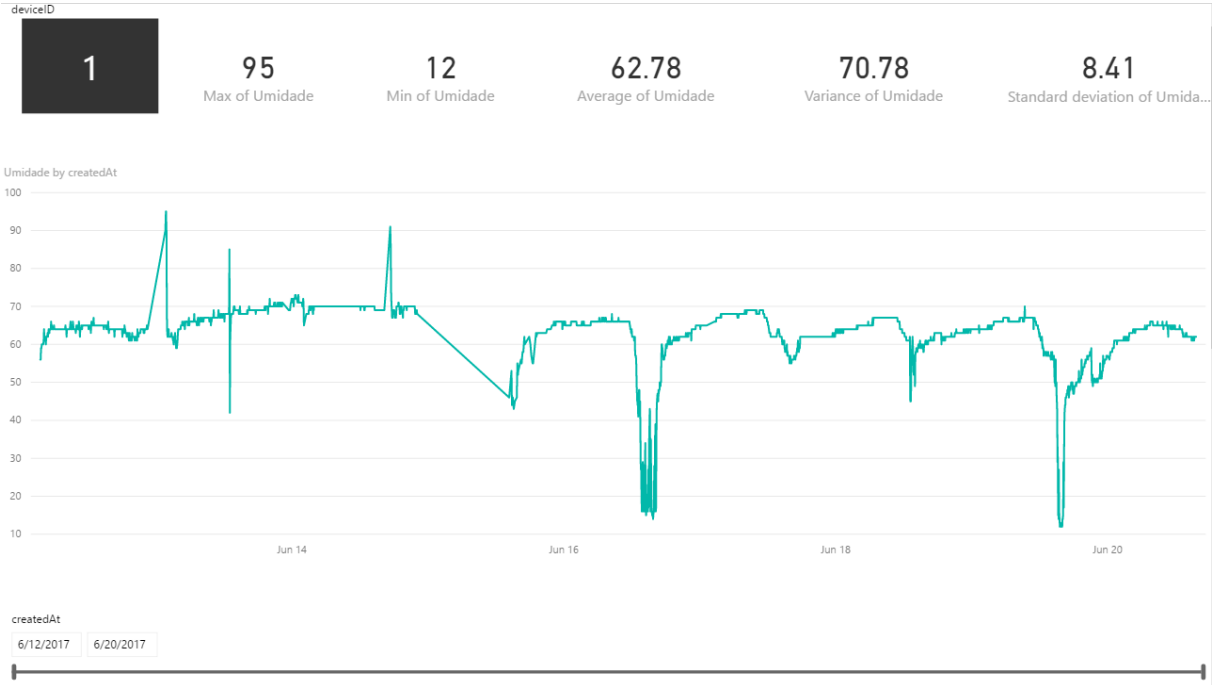

Figura 76 – Análise Umidade Relativa Histórica – Primeira Parte

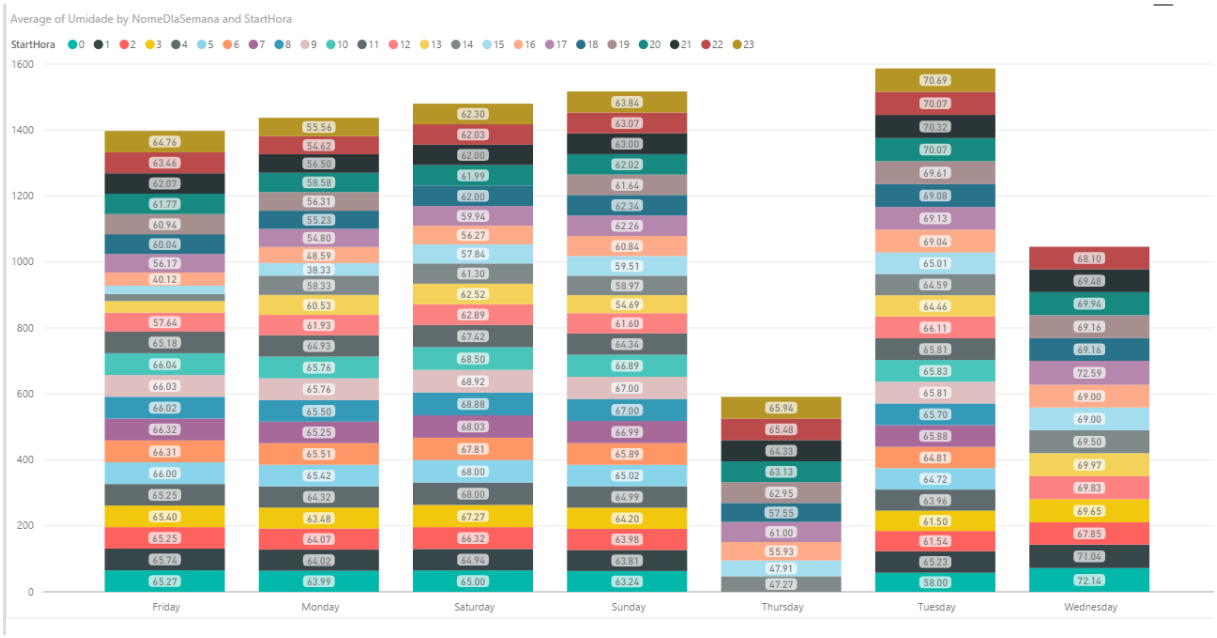

Figura 77 – Análise Umidade Relativa Histórica – Segunda Parte

Figura 78 – Análise Indicativo da Concentração de Gás GLP Histórica – Primeira Parte

Figura 79 – Análise Indicativo da Concentração de Gás Glp Histórica – Segunda Parte

Apêndice F - Diagrama Esquemático Arduino Uno R3

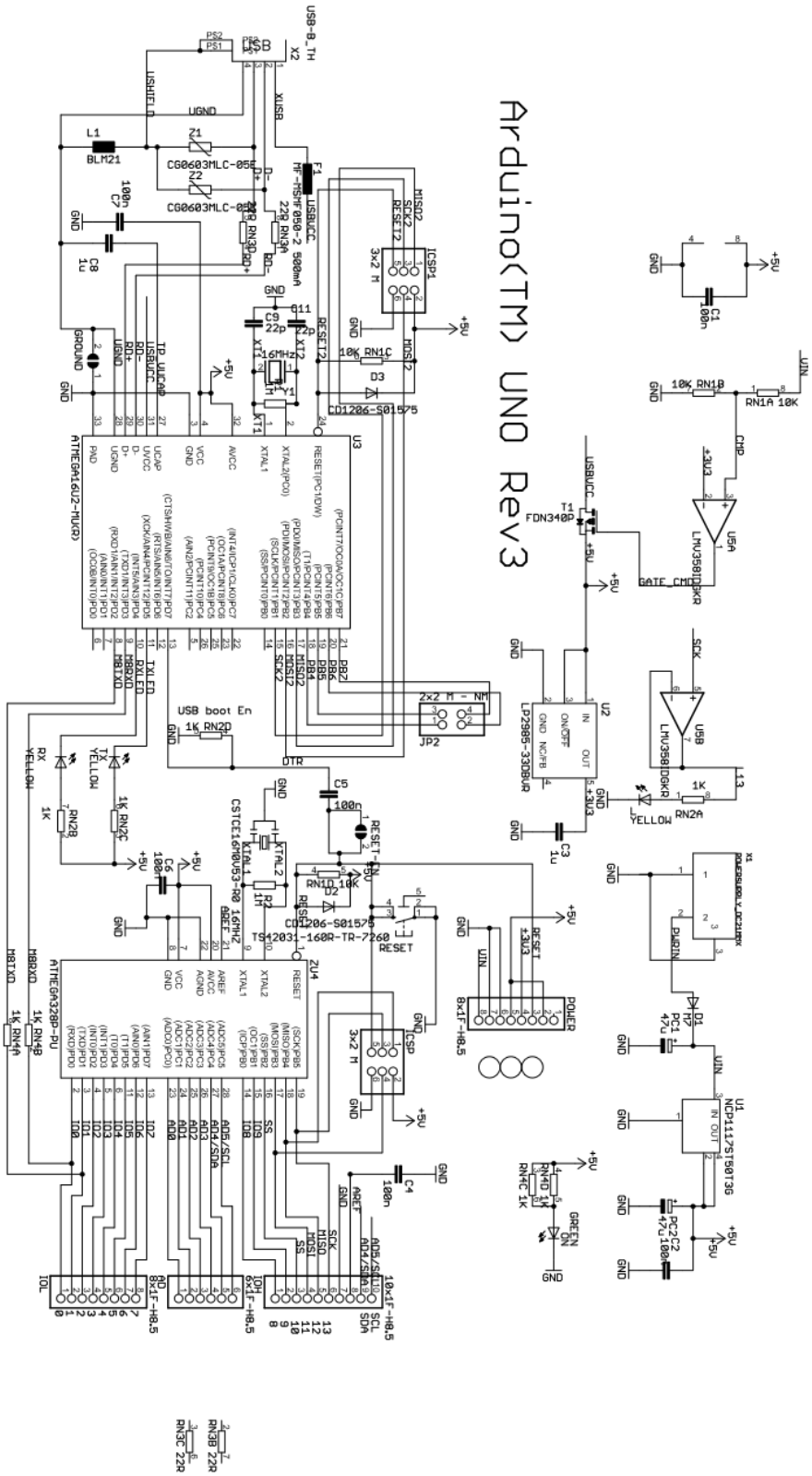

Figura 80 – Diagrama Esquemático Arduino Uno R3

Apêndice G - Preparação do Ambiente Thingspeak

O primeiro passo foi criar uma conta MathWorks®. Para tal acessou-se thingspeak.com e clicou-se em "Get started for free", conforme a Figura 81.

Figura 81 – Página Inicial da Plataforma Thingspeak

Após ter criado a conta, acessou-se o serviço. No menu, escolheu-se a opção "Channels" e clicou-se em "New Channel", conforme a Figura 82.

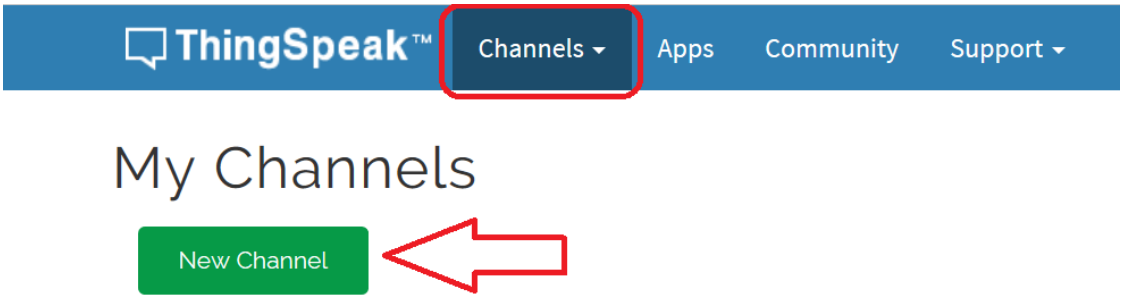

Figura 82 – Criação de um Novo Canal Thingspeak

Esse processo é o responsável por começar a configuração do provisionamento de um canal para receber os dados em tempo real. Em seguida, para continuar a configuração do canal, deu-se um nome para o mesmo, fez-se uma breve descrição e adicionou-se novos campos para receber os valores da temperatura, umidade, concentração de poluentes e grau de emissão de luz infravermelha e a identificação do dispositivo. Essa etapa da configuração é apresentada na Figura 83.

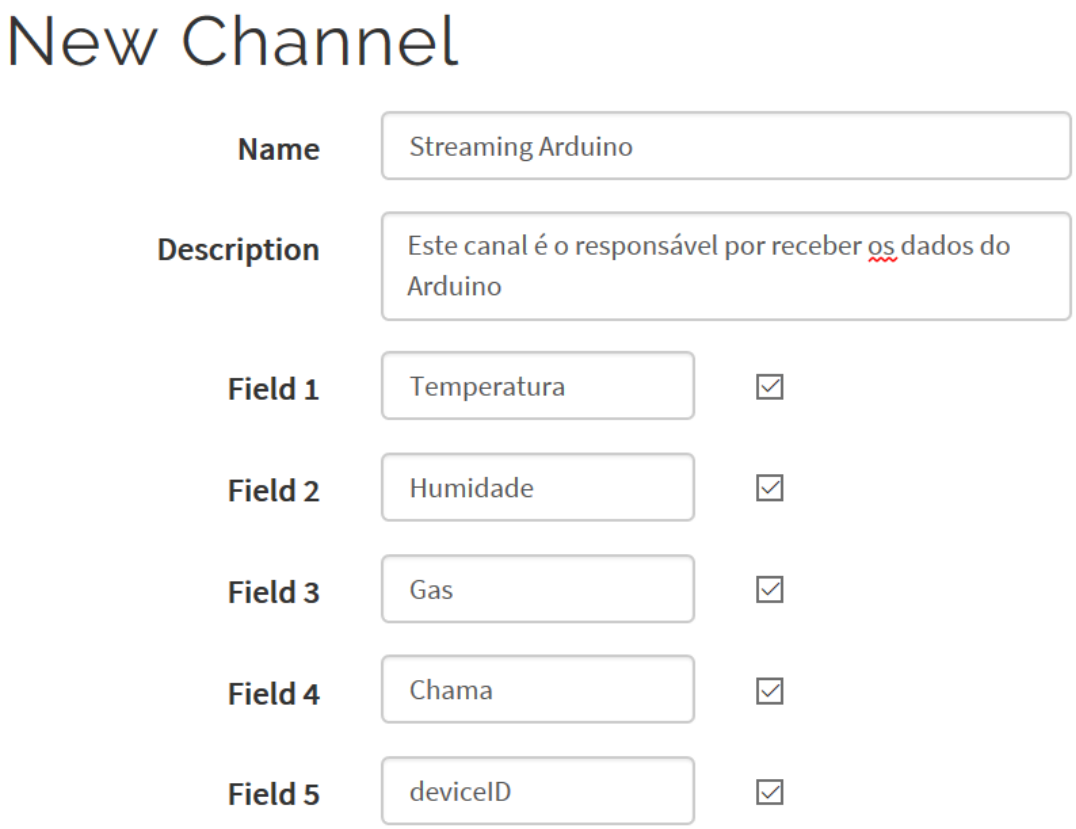

Figura 83 – Configuração do Novo Canal Thingspeak

Ainda na mesma página pode-se configurar outros parâmetros como a localização geográfica. Todavia não se fez nenhuma alteração e clicou-se em "Save Channel" para concluir a configuração do canal ThingSpeak.

Após esses passos, o serviço disponibilizou a visualização dos dados escolhidos em gráficos, conforme pode ser visto na Figura 84. Nessa mesma página escolheu-se no menu a opção "Data Import / Export".

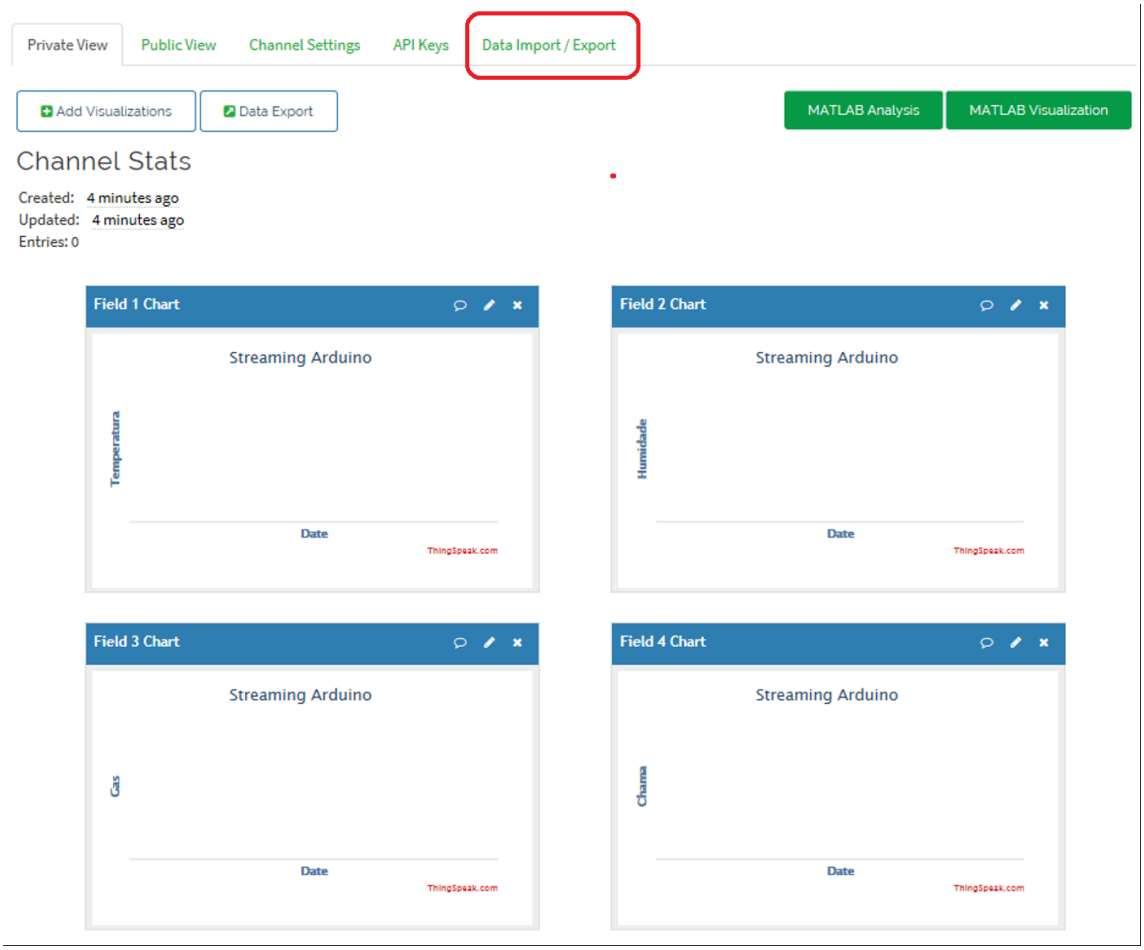

Figura 84 – Layout do Canal Recém-Criado

Então teve-se acesso à sintaxe às chaves necessárias para inserir dados no canal, conforme a Figura 85. Essa chave será utilizada no código desenvolvido para o Arduino enviar os dados coletados.

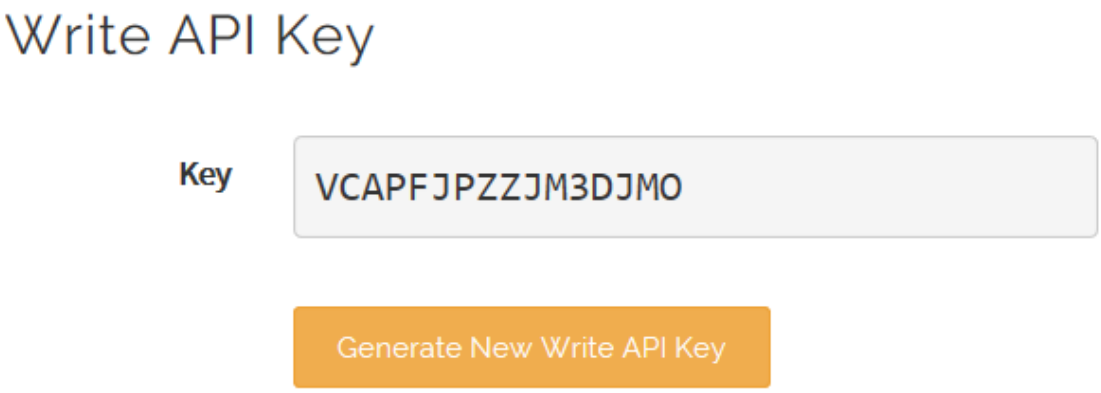

Figura 85 – Chave do Canal

Outra informação muito importante é o número de identificação do canal, chamado de "Channel ID". Esse número pode ser objetivo do cabeçalho da página do canal, conforme a Figura 86.

Figura 86 – ID do Canal