

**UNIVERSIDADE FEDERAL DO MARANHÃO
CENTRO DE CIÊNCIAS EXATAS E TECNOLÓGICAS
CURSO DE ENGENHARIA ELÉTRICA**

LUCAS ANDRADE BRAÚNA CUNHA

**SISTEMA DE SEGURANÇA E MONITORAMENTO
RESIDENCIAL VIA INTERNET UTILIZANDO SISTEMA
EMBARCADO COM COMPUTAÇÃO EM NUVEM**

São Luís

2019

LUCAS ANDRADE BRAÚNA CUNHA

**SISTEMA DE SEGURANÇA E MONITORAMENTO
RESIDENCIAL VIA INTERNET UTILIZANDO SISTEMA
EMBARCADO COM COMPUTAÇÃO EM NUVEM**

Trabalho de Conclusão de Curso apresentado ao Colegiado de Curso da Engenharia Elétrica do Centro de Ciências Exatas e Tecnologia da Universidade Federal do Maranhão, como parte dos requisitos para obtenção do diploma de Graduação em Engenharia Elétrica.

Orientador: Prof. Dr. José de Ribamar Braga Pinheiro Júnior

São Luís

2019

ANDRADE BRAÚNA CUNHA, LUCAS.

SISTEMA DE SEGURANÇA E MONITORAMENTO RESIDENCIAL VIA INTERNET UTILIZANDO SISTEMA EMBARCADO COM COMPUTAÇÃO EM NUVEM / LUCAS ANDRADE BRAÚNA CUNHA. - 2019.

70 f.

Orientador(a): JOSÉ DE RIBAMAR BRAGA PINHEIRO JÚNIOR.
Monografia (Graduação) - Curso de Engenharia Elétrica,
Universidade Federal do Maranhão, SÃO LUÍS, 2019.

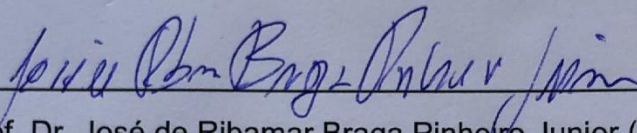
1. COMPUTAÇÃO EM NUVEM. 2. SEGURANÇA. 3. SISTEMA EMBARCADO. I. DE RIBAMAR BRAGA PINHEIRO JÚNIOR, JOSÉ. II. Título.

LUCAS ANDRADE BRAÚNA CUNHA

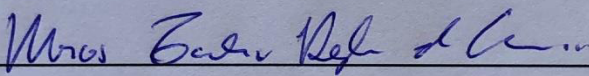
**SISTEMA DE SEGURANÇA E MONITORAMENTO RESIDENCIAL VIA INTERNET
UTILIZANDO SISTEMA EMBARCADO COM COMPUTAÇÃO EM NUVEM**

Aprovada em: 09/07/2019

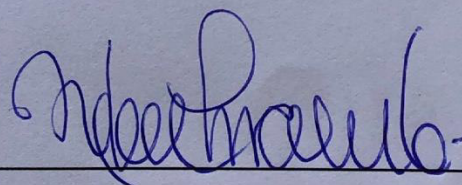
Banca Examinadora:



Prof. Dr. José de Ribamar Braga Pinheiro Junior (Orientador)
Departamento de Engenharia Elétrica - UFMA



Prof. MSc. Marcos Tadeu Rezende De Araújo (Membro)
Departamento de Engenharia Elétrica - UFMA



Prof. MSc. Nelson José Camelo (Membro)
Departamento de Engenharia Elétrica - UFMA

São Luís

2019

AGRADECIMENTOS

Agradeço a Deus, sem ele, nada seria. Sempre me dando força, direcionamento e conforto em todos os momentos.

Aos meus pais Lucineide Andrade e Carlos Augusto, e ao meu irmão Vitor Augusto, por todo amor, apoio, dedicação e confiança ao longo desses anos. Pelo cuidado, atenção e dedicação durante o meu crescimento em todas as áreas.

Aos meus avós maternos e paternos, pelo amor, auxílio, cuidado e orações ao longo da minha trajetória.

Aos meus primos, por serem companheiros, amigos e por sempre compartilhar bons momentos comigo.

Ao meu orientador José de Ribamar Braga e toda a equipe do LABGEA, por todo auxílio, atenção, disposição e conselhos fornecidos ao longo do desenvolvimento desta monografia e durante a graduação.

Aos Amigos de Daniel Seixas, ao Kaskay, e a Apoteose, pelo apoio e suporte. Pelos momentos de descontração compartilhados ao longo desses anos.

Ao meu professor e conterrâneo Flávio Fonseca, quem me despertou interesse pelas ciências exatas.

A todos que contribuíram de maneira direta ou indireta para a realização deste trabalho e conclusão desta etapa.

RESUMO

A crescente integração da tecnologia no cotidiano e o surgimento de diversas plataformas microcontroladas aliado à facilidade de acesso vem aumentando o número de possibilidades de aplicações. Dentre as diversas possibilidades podemos citar a automação residencial por proporcionar comodidade, praticidade e segurança. Este trabalho se propõe a utilizar a plataforma Arduino no desenvolvimento de uma alternativa de baixo custo, confiável e eficiente para o gerenciamento de segurança residencial, com base no conceito de Internet das Coisas. O sistema pode ser acessado por dispositivos *mobile* através de uma aplicação, na qual os usuários podem verificar as informações relativas ao sistema. Os recursos acessados são armazenados em nuvem, utilizando memória, capacidade de armazenamento e cálculo de computadores e servidores hospedados em *Datacenter* e interligados por meio da Internet.

Palavras-chave: *Automação residencial, Arduino, Segurança, nuvem*

ABSTRACT

The increasing integration of technology in everyday life and the emergence of several microcontrolled platforms to an ease of access comes a number of possibilities of use, among them the residential automation for convenience, security and safety. This use the platform of the development system in the Internet. Mobile devices through an application, in which the components can be consulted as information related to the system, can access the system. The resources accessed are stored in the cloud, storage capacity and evaluation of systems and servers hosted in datacenter and interconnected through the Internet.

Keywords: Home automation, Arduino, Security, cloud.

ÍNDICE DE FIGURAS

Figura 1 - Insegurança no mundo	13
Figura 2: Organização lógica de um microcontrolador.....	19
Figura 3: Sensores usados em sistemas embarcados	20
Figura 4: Ferramentas de computação em nuvem.....	22
Figura 5: Blocos básicos na construção do IoT	25
Figura 6: Arduino UNO R3	28
Figura 7: Pinos do microcontrolador ATmega328	29
Figura 8: Reed-Switch	32
Figura 9: Reed-Switch na presença de um ímã permanente	33
Figura 10: Sensor de chama.....	34
Figura 11: Sensor PIR	36
Figura 12: Funcionamento do sensor PIR.....	37
Figura 13: Lentes Fresnel no sensor PIR	38
Figura 14: Shield Ethernet acoplado no Arduino UNO.....	39
Figura 15: Esquema de montagem dos sensores no Arduino em <i>protoboard</i>	42
Figura 16: Montagem do sistema em protoboard.....	45
Figura 17: Fluxograma do código do sistema de monitoramento residencial.....	46
Figura 18: Dados no Firebase	48
Figura 19: Tela inicial do aplicativo	51
Figura 20: Histórico de ocorrências.....	51
Figura 21: Tela de notificações.....	52
Figura 22: Reed Switch na porta principal do LABGEA	53
Figura 23: Sensor de chama sobre o quadro elétrico do LABGEA.....	54
Figura 24: Sensor de presença.....	55

ÍNDICE DE TABELAS

Tabela 1: Tipos de sensores e sua classificação	21
Tabela 2: Conexão entre o módulo Ethernet e o Arduino	43
Tabela 3: Conexão entre o módulo relé e o Arduino	43
Tabela 4: Conexão dos sensores com o Arduino	43

ÍNDICE DE ABREVIações

ABESE	Associação Brasileira das Empresas de Sistemas Eletrônicos de Segurança
API	<i>Application Programming Interface</i>
ARP	<i>Address Resolution Protocol</i>
CFTV	Circuito fechado de Televisão
CPU	<i>Central Processing Unit</i>
DIY	<i>Do It Yourself</i>
EEPROM	<i>Electrically-Erasable Programmable Read-Only Memory</i>
EXI	<i>Efficient XML Interchange</i>
FPGAs	<i>Field Programmable Gate Array</i>
GfK	<i>Growth from Knowledge</i>
GLP	Gás Liquefeito de Petróleo
GND	<i>Ground</i>
HTTP	<i>Hypertext Transfer Protocol</i>
IaaS	<i>Infrastructure as a Service</i>
IBGE	Instituto Brasileiro de Geografia e Estatística
ICMP	<i>Internet Control Message Protocol</i>
IEEE	<i>Institute of Electrical and Electronic Engineers</i>
IGMP	<i>Internet Group Management Protocol</i>
IoT	<i>Internet of Things</i>
IP	<i>Internet Protocol</i>
IPv4	Protocolo de Internet versão 4
LABGEA	Laboratório do Grupo de Eletrônica Aplicada
LED	<i>Light Emitting Diode</i>
MAC	<i>Media Access Control</i>
NFC	<i>Near Field Communication</i>
NIST	<i>National Institute of Standards and Technology</i>
OWL	<i>Web Ontology Language</i>
PaaS	<i>Platform as a Service</i>

PHY	<i>Physical</i>
PIR	<i>Passive Infrared Radiation</i>
PPoE	<i>Point to Point Protocol over Ethernet</i>
PWM	<i>Pulse Width Modulation</i>
RAM	<i>Random Access Memory</i>
RDF	<i>Resource Description Framework</i>
RFID	<i>Radio Frequency Identification</i>
RISC	<i>Reduced Instruction Set Computer</i>
SaaS	<i>Software as a Service</i>
SPI	<i>Serial Peripheral Interface</i>
TCP	<i>Transmission Control Protocol</i>
UC	Unidade De Controle
UDP	<i>User Datagram Protocol</i>
ULA	Unidade Lógica Aritmética
VIN	Tensão de entrada

SUMÁRIO

1 INTRODUÇÃO	13
1.1 Objetivos	14
Nesta seção são descritos os objetivos gerais e específicos esperados	14
durante e após a realização deste trabalho.	14
1.1.1 OBJETIVOS GERAIS.....	14
1.1.2 OBJETIVOS ESPECÍFICOS	15
1.2 Justificativa	15
1.4 Estrutura do trabalho	17
2 FUNDAMENTAÇÃO TEÓRICA	18
2.1 Microcontroladores	18
2.2 Sensores	20
2.3 Computação em nuvem	21
2.4 Internet das coisas	24
3 MATERIAIS E MÉTODOS	27
3.1 Hardware	27
3.1.1 PLATAFORMA ARDUINO	27
3.1.2 ARQUITETURA DO ARDUINO UNO.....	29
3.1.3 SOFTWARE ARDUINO IDE	30
3.1.3 SENSORES	31
3.1.4 SHIELD ETHERNET	38
3.2 Computação em nuvem	39
3.2.1 FIREBASE	40
4 RESULTADOS	41
4.1 Características Gerais	41
4.2 Protótipo	41
4.2.1 HARDWARE	42
4.2.2 SOFTWARE.....	45
4.2.3 COMPUTAÇÃO EM NUVEM	47

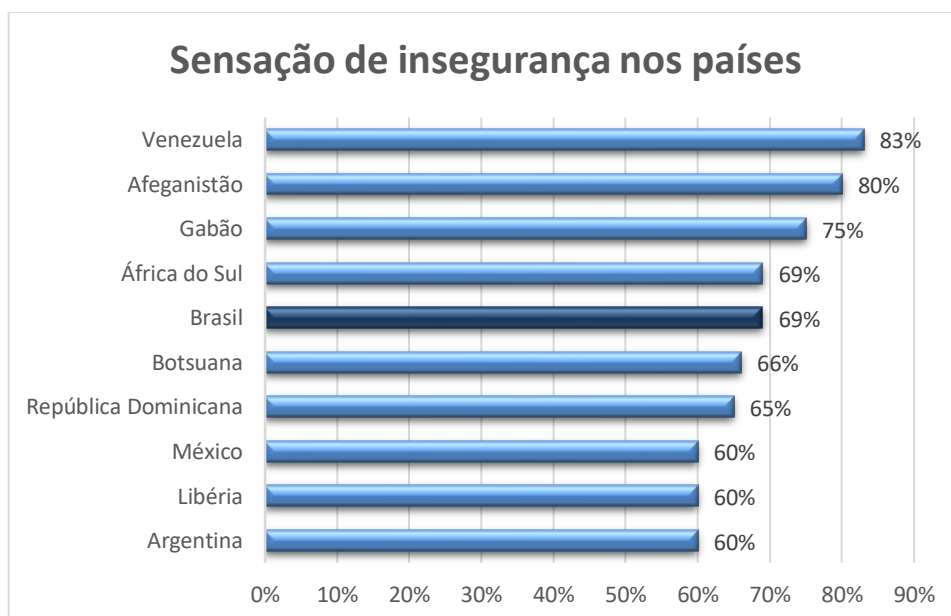
4.3 Implementação no LABGEA.....	52
5 CONCLUSÃO	56
5.1 Trabalhos futuros	57
REFERÊNCIAS.....	58
APÊNDICE.....	62

1 INTRODUÇÃO

A tecnologia disseminou-se a tal ponto, que hoje está ao alcance de todos, e é muito difícil citar um ramo na qual a automação não está presente: de eletrodomésticos controlados por computador, à medicina, engenharia, escolas, e tudo mais que se possa imaginar (MANDULA et al., 2016). Diante disso, é possível observar, principalmente nas economias mais desenvolvidas, que o cenário para as chamadas “casas inteligentes” tem evoluído de maneira muito significativa nos últimos anos. Segundo dados do instituto americano de pesquisas ABI, em 2012 já existiam 1,5 milhão de sistemas autônomos de residências instalados nos Estados Unidos (ABIRESEARCH, 2012).

Paralelo a isso, no Brasil, a sensação de insegurança da população aumentou bastante, estima-se que cerca de 70% das pessoas sentem-se inseguras, seja nos locais públicos, ou dentro da própria casa (IBGE, 2018). No gráfico da Figura 1, é mostrada a relação dos países com os maiores índices de insegurança, com destaque para o Brasil, como 5º colocado (GALLUP, 2018).

Figura 1 - Insegurança no mundo



Fonte: "2018 Global Law and Order", pesquisa do Instituto Gallup.

Uma pesquisa realizada pela GfK (*Growth from Knowledge*) em 2016 expôs que cerca de 57% da população brasileira que possui acesso à Internet acreditam que a automação residencial terá mais impacto e ganhará mais espaço no Brasil nos próximos anos. De acordo com o estudo, o maior apelo está relacionado com a segurança e ao monitoramento de suas residências, representando 80% dos entrevistados (GfK BRAZIL, 2016).

Todavia, os recursos mais aprimorados e que garantem maior segurança e conforto possuem preços elevados e muitas vezes inacessíveis para o brasileiro médio. Segundo a Revista de Segurança Eletrônica, os serviços de alarmes sonoros (em torno de 10 sensores) custam em torno de 3 a 4 mil reais, chegando a custar até 10 mil reais se forem equipados com sistemas de monitoramento remoto e controle inteligente (ELETRÔNICA, 2017).

Buscando transpassar o obstáculo que é o alto custo de um sistema de segurança residencial, este trabalho propõe o desenvolvimento de um sistema de vigilância e monitoramento, de baixo dispêndio e utilizando os princípios de sistemas embarcados e da computação em nuvem, visando solucionar os problemas aludidos.

1.1 Objetivos

Nesta seção são descritos os objetivos gerais e específicos esperados durante e após a realização deste trabalho.

1.1.1 OBJETIVOS GERAIS

Estudar, fundamentar, desenvolver e implementar um sistema de segurança e monitoramento residencial usando sistema embarcado controlado via Internet com computação em nuvem e que seja de baixo custo comparado com as soluções de mercado. Além disso, propor um projeto adaptativo, podendo variar, em relação à quantidade e tipos de sensores, de acordo com o consumidor final.

1.1.2 OBJETIVOS ESPECÍFICOS

Os objetivos específicos são os seguintes:

- Fazer uso de ferramentas livres, como *software* e *hardware open source*, garantindo o baixo custo do projeto;
- Possibilitar o uso de variados dispositivos que imprimem maior segurança e comodidade, como por exemplo: sensores de presença, fogo, acionamento automático de lâmpadas, entre outros;
- Assegurar o correto funcionamento do sistema através da divisão de tarefas entre: monitoramento, envio de dados, computação em nuvem e exibição e controle pelo cliente.

1.2 Justificativa

A segurança residencial é um segmento de participação bastante relevante no mercado de casas inteligentes. Segundo a Associação Brasileira das Empresas de Sistemas Eletrônicos de Segurança (ABESE), cerca de 26 mil empresas atuam no setor. Estas empresas concentram seus serviços de instalação em diversos níveis, sobretudo em equipamentos, sistemas de circuito fechado (CFTV) e alarmes. Contudo, o alto preço praticado por essas empresas inviabiliza o acesso desses serviços para muitas pessoas, por isso é de relevante importância alternativas de baixo custo e de fácil aquisição (ABESE, 2018).

Automatizar é substituir o trabalho humano ou animal por máquinas ou sistemas. No início a automação era voltada principalmente para área industrial, que após ser aplicada em larga escala nas mais variadas áreas de produção houve um crescimento no mercado de automação de pequeno porte. Introduziu-se o uso de microcontroladores, incentivado pelo desenvolvimento econômico que aumentou o poder aquisitivo das pessoas e pela busca de conforto, praticidade e segurança. Celulares que hoje são comparados a computadores ajudam ainda mais no desenvolvimento do setor de automação.

A computação em nuvem (*Cloud Computing*), é um novo paradigma de computação onde os recursos disponíveis estão alocados na Internet. Os principais conceitos associados com a computação em nuvem são: a virtualização, ou seja, a criação de ambientes virtuais para os usuários, escondendo as características físicas da plataforma computacional; escalabilidade, que diz respeito à capacidade de aumento ou redução do tamanho dos ambientes virtuais, caso seja necessário; e, por último, modelo *pay-per-use*, em que o usuário só paga por aquele serviço que consome (VAQUERO et al., 2009). Em um ambiente como esse, todas as complexidades das instalações dos sistemas computacionais ficam transparente para o usuário, pois ele não precisa se preocupar com a manutenção, instalação, muito menos da localização do sistema.

O Google, por exemplo, disponibiliza através da plataforma FireBase uma série de serviços com estas características. Estes serviços incluem: *Cloud Firestore*, que permite armazenar e sincronizar dados em escala global; *Cloud Functions*, através do qual podemos executar código de *back-end* para dispositivos móveis sem gerenciar servidores; Autenticação que simplifica o processo de autenticação de usuário; *Cloud Storage* onde podemos armazenar e oferecer arquivos na escala do Google; *Realtime Database* com a disponibilidade de armazenar e sincronizar dados de aplicativos em milissegundos; entre outros (GOOGLE, 2019a).

Dentro do contexto deste trabalho, a utilização de computação em nuvem servirá como base para a troca de informações e processamento de dados. Sendo assim, quando uma informação for gerada pelo sistema embarcado proposto neste plano, por exemplo, ela será imediatamente disponibilizada para a Nuvem que processará esses dados e os entregará ao remetente para que este tome as devidas ações. A resposta a essas ações será feita no caminho contrário utilizando os mesmos recursos. O importante aqui é ressaltar que uma vez seguidas todas as interfaces de comunicação entre os sistemas, não caberá ao nosso sistema a preocupação de onde os recursos disponibilizados pela Nuvem estão, como foram implementados, em que plataformas de hardware eles serão processados, entre outros questionamentos.

1.4 Estrutura do trabalho

Este trabalho está estruturado da seguinte forma: no segundo capítulo são apresentados alguns conceitos e definições a respeito do tema do trabalho, para que o leitor possa compreender a utilização dos materiais e métodos escolhidos. No capítulo 3 são apresentadas as ferramentas escolhidas para desenvolver o sistema de monitoramento residencial e a metodologia de desenvolvimento do trabalho.

Os resultados do projeto proposto já implementado, juntamente com a maquete do sistema serão apresentados no capítulo 4. Por fim, no capítulo 5 serão apresentadas as conclusões do trabalho, assim como algumas sugestões para trabalhos futuros.

2 FUNDAMENTAÇÃO TEÓRICA

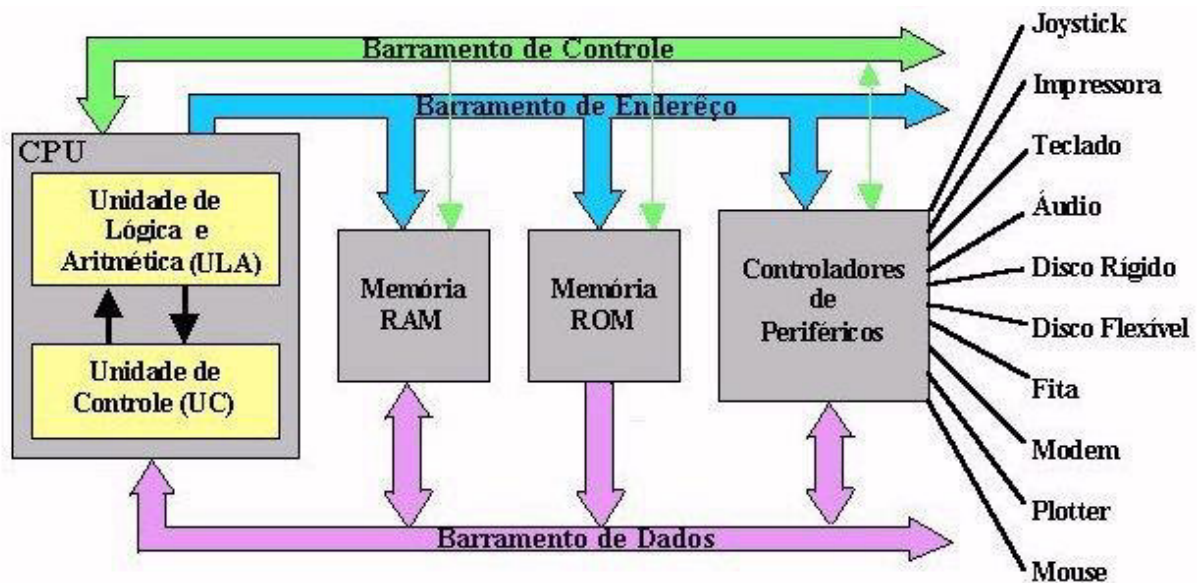
Neste capítulo são abordados os principais blocos para construção de um sistema embarcado responsável pelo monitoramento residencial. A seção 2.1 expõe sobre microcontroladores e seu funcionamento. Os sensores e sua finalidade são abordados na seção 2.2, seguido da fundamentação de computação em nuvem na seção 2.3. Por fim, na seção 2.4, há um panorama sobre a Internet das coisas e como esse conceito está diretamente relacionado com a revolução tecnológica do século XXI.

2.1 Microcontroladores

Nos últimos anos, o uso de microcontroladores e sua difusão têm sido fortemente influenciados pelo avanço da tecnologia (BOLANAKIS, 2017). Isso se deve ao fato, principalmente, pelo desenvolvimento de dispositivos e equipamentos semicondutores mais avançados, construídos com alta densidade, possibilitando a realização de processos mais complexos (DU BUF; BAYER, 2010). Para Bolanakis (2017, p. 3), microcontroladores são sistemas de computador de chip único (e também em uma única placa), incorporando (a) unidade central de processamento (CPU, *central processing unit*), (b) memória, (c) periféricos de entrada e saída, bem como um código de *firmware* personalizado para controle de *hardware*.

A CPU de um microcontrolador é composta por uma unidade lógica aritmética (ULA), por uma unidade de controle e por unidades de memória especiais, os registradores. Para que a CPU possa realizar tarefas são necessários outros componentes, como unidades de memória, entrada e saída. Na Figura 2 é possível observar um diagrama de blocos com uma possível interface entre a CPU e outros dispositivos.

Figura 2: Organização lógica de um microcontrolador



Fonte: (FILHO, 2016)

A ULA é a área da CPU na qual as operações lógicas e aritméticas são realizadas sobre os dados. O tipo de operação realizada é determinado pelos sinais da unidade de controle (UC). Os dados a serem operados pela ULA podem ser oriundos de uma memória ou de uma unidade de entrada. Os resultados das operações realizadas na ULA podem ser transferidos tanto para uma memória de dados como para uma unidade de saída (FILHO, 2016).

A função da UC é comandar as operações da ULA e de todas as outras unidades conectadas a CPU, fornecendo sinais de controle e temporização. Essa unidade contém circuitos lógicos e de temporização que geram os sinais adequados e necessários para executar cada instrução de um programa (FILHO, 2016).

As unidades de memória permitem armazenar grupos de dígitos binários que podem representar instruções para serem executadas pelo processador ou dados para serem manipulados pelo mesmo (FILHO, 2016). O periférico de entrada consiste em todos os dispositivos utilizados para obter informações e dados externos ao processador. Já os periféricos de saída consistem em dispositivos capazes de transferir dados e informações do processador para o exterior.















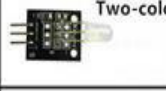





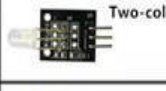















Segundo Martins (2005, p. 15), existem vários tipos de microcontroladores e eles são diferenciados da seguinte forma: velocidade de processamento; quantidade

de memória interna para armazenar os dados (memória de dados) e para armazenar as instruções dos programas (memória de programa); quantidade de pinos; forma de alimentação; quantidade de periféricos; arquitetura e o conjunto de instruções dos circuitos internos.

2.2 Sensores

Sensores são elementos de um sistema de medição que são diretamente afetados por um fenômeno, corpo ou substância que contém a grandeza a ser medida. Dessa forma, esses dispositivos transformam grandezas físicas em sinais que podem ser processados eletronicamente (STANLEY; LEE, 2018). Existem normalmente muitas maneiras de medir qualquer propriedade física através de sensores. A distância pode ser medida usando um sensor ultrassônico ou radar. A rotação pode ser estimada com acelerômetros, sensores magnéticos e sensores de variação angular (giroscópio). A Figura 3, ilustra alguns sensores disponíveis no mercado utilizados em sistemas embarcados.

Figura 3: Sensores usados em sistemas embarcados

 JoyStick XY	 Flame	 RGB LED	 Heartbeat	 Light Cup	 Hall magnetic
 Relay	 Linear Hall	 SMD RGB	 7Color flash	 Tilt switch	 TEMP 18B20
 Big sound	 Touch	 Two-color	 Laser emit	 Ball switch	 Analog temp
 Small sound	 Digital temp	 Two-color	 Button	 photoresistor	 TR emission
 Tracking	 Buzzer	 Reed switch	 Shock	 temp and humidty	 IR receiver
 Avoid	 Passive buzzer	 Mini Reed	 Rotary encoders	 Analog Hall	 Tap module Light blocking

Fonte: (GEARBEST BRASIL, 2019)

Todo sensor pode ser classificado como passivo ou ativo. Um sensor passivo não precisa de uma fonte de energia adicional, ou seja, é gerado diretamente um sinal elétrico em resposta a um estímulo externo. Os sensores ativos exigem uma fonte externa para a sua operação, que é chamada de sinal de excitação que é utilizado pelo sensor para produzir um sinal de saída. Por exemplo, enquanto o sensor passivo de movimento não emite, somente detecta a movimentação de luz infravermelha emitida por corpos quentes, o sensor ativo conta com um circuito que emite um feixe de luz e outro que a detecta. A tabela 1, a seguir, mostra alguns tipos de sensores com sua classificação e função:

Tabela 1: Tipos de sensores e sua classificação

Sensor	Tipo	Função
Infravermelho	Passivo	Movimento
Indutivo	Ativo	Movimento/Presença
Óticos	Ativo	Movimento/Presença
Chave Magnética	Passivo	Aberto/Fechado
Chama/Fogo	Passivo	Detecção de Fogo
Gás	Ativo	Detecção de Gás

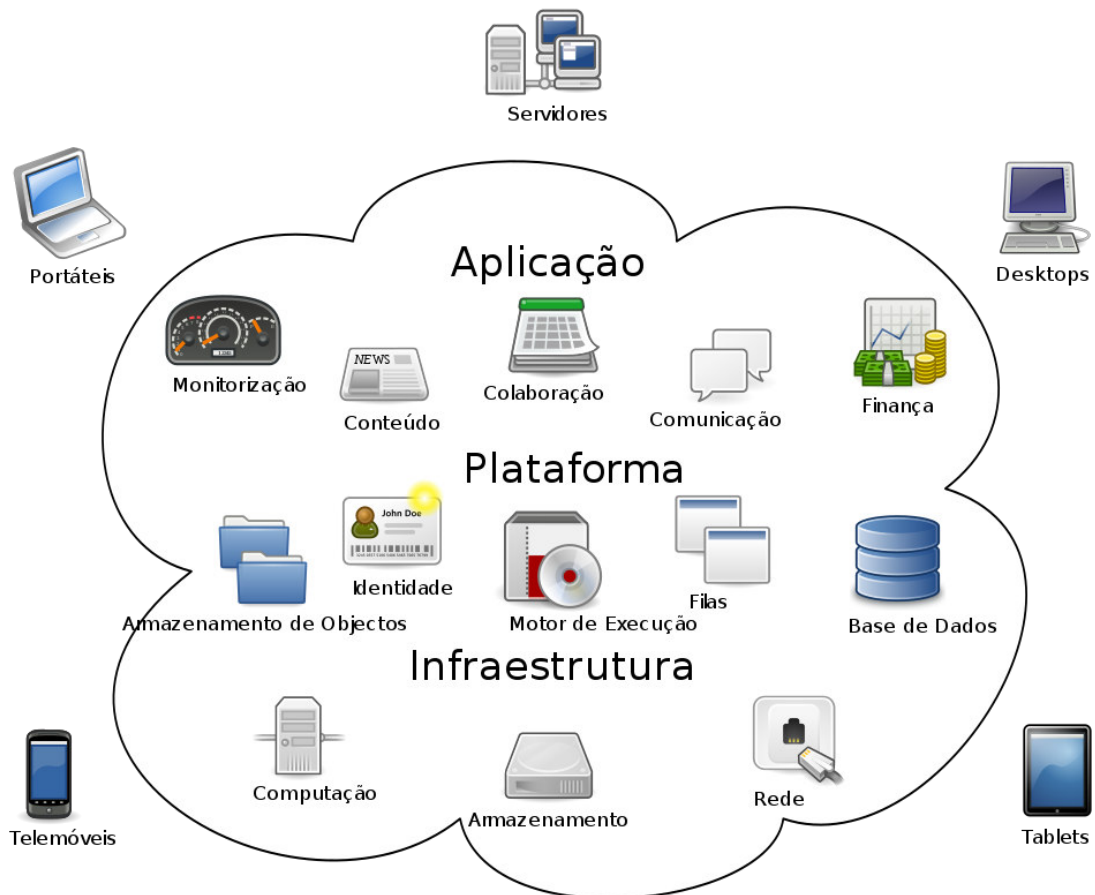
Fonte: (Autor, 2019)

2.3 Computação em nuvem

Segundo Borges (2011), a computação em nuvem pode ser definida como um paradigma de infraestrutura que fornece um grande conjunto de serviços baseados na *web*, com o objetivo de fornecer funcionalidades que antes só eram disponíveis com um considerável investimento em *hardware* e *software*. O NIST (*National Institute of Standards and Technology* – USA) define computação em nuvem como a representação do mais conveniente modelo de acesso, sempre que necessário, de um conjunto compartilhado de recursos computacionais, entre eles, redes, servidores, armazenamento, aplicações e serviços, podendo ser acessados rapidamente e de qualquer ambiente com o esforço mínimo dos recursos do cliente.

A computação em nuvem apresenta um grande número de recursos e características que validam e atestam seu uso. A elasticidade e escalonamento de recursos possibilita ao cliente ter a expectativa de possuir recursos ilimitados, disponíveis na quantidade e no momento que precisar, podendo aumentar ou diminuir esta demanda. Além disso, há uma espécie de autoatendimento, onde os usuários podem solicitar, personalizar, pagar e utilizar os recursos sem a intervenção humana. Assim, para o funcionamento dessa tecnologia, os usuários e desenvolvedores devem ter acesso a rede e à dispositivos que permitam acessá-la. A Figura 4 apresenta alguns recursos disponibilizados na computação em nuvem.

Figura 4: Ferramentas de computação em nuvem



Fonte: (WIKIPÉDIA, 2019)

A execução dos serviços em nuvem depende da necessidade da aplicação a ser contratada pelo cliente. Podendo ser pública, onde é disponibilizada por um

prestador de serviço, de forma igualitária para todas as pessoas no mesmo nível hierárquico. Podem ainda ser privadas, onde são operadas por uma única organização, sendo gerenciada pela mesma. Já o modelo comunitário é compartilhado por várias organizações ou grupos de pessoas engajadas no mesmo propósito, elas são geralmente gerenciadas por organizações ou por terceiros.

Atualmente a computação em nuvem pode ser dividida em dez tipos, os principais são:

- IaaS - Infraestrutura como Serviço (*Infrastructure as a Service*): Essa tipologia contém os componentes básicos da TI em nuvem e possibilita o acesso à recursos de armazenamento e de computação baseados em Internet. É a categoria mais básica entre os tipos de computação em nuvem, no IaaS o cliente pode utilizar servidores, máquinas virtuais, armazenamento, redes e sistemas operacionais. Os principais fornecedores desse tipo de serviço são o Google, Amazon e Microsoft.
- PaaS - Plataforma como Serviço (*Platform as a Service*): Possibilita os clientes as ferramentas e serviços necessários para criar e hospedar aplicações na Web. Ela foi desenvolvida, principalmente, para o desenvolvedor operar rapidamente os aplicativos Web ou móveis na Internet, dispensando a preocupação com a configuração do sistema ou o gerenciamento dos servidores, armazenamento, redes e banco de dados. Os principais fornecedores são: IBM Bluemix, Windows Azure e Jelastic.
- SaaS - Software como Serviço (*Software as a Service*): Nesse modelo, o fornecedor do *software* se responsabiliza por toda a estrutura necessária para a que o sistema seja executado pelo cliente, que acessa esse serviço via Internet. No SaaS, o usuário não administra as características individuais da aplicação, exceto configurações específicas. Esse tipo de tipologia é amplamente utilizado por possuir plataformas gratuitas que possibilitam seu uso, as principais são: Google Docs, Microsoft SharePoint Online, entre outros.

2.4 Internet das coisas

A Internet das Coisas (*Internet of Things* (IoT)) é fruto dos avanços de várias áreas como sistemas embarcados, microeletrônica, comunicação e sensoriamento (SANTOS et al., 2016). A Internet das coisas é considerada a extensão da Internet atual, na qual possibilita que objetos comuns e do dia-a-dia adquiram capacidade computacional e de comunicação, para que sejam controlados ou acessados remotamente.

Os objetos inteligentes com diferentes recursos, quando conectados a rede, potencializam o surgimento de novas aplicações. Dessa forma, conectar esses objetos à Internet significa criar a Internet das Coisas. Na IoT, os objetos e dispositivos enviam informações para o usuário, com a possibilidade de controlá-lo. Isso proporciona o surgimento de inúmeras aplicações tais como monitoramento residencial, coleta de dados de pacientes, monitoramento de idosos, sensoriamento de locais de difícil acesso e inóspito, entre outras.

Para Wang (2015), a IoT é responsável por uma nova revolução na tecnologia da informação. Assim, por meio da Internet das Coisas, há a possibilidade de tornar a interação homem-computador invisível, ou seja, integrar a informática com as ações e comportamentos naturais das pessoas, caminhando para a computação ubíqua.

Segundo Santos (2019), IoT combina diversas tecnologias complementares com o objetivo de propiciar a integração dos objetos do mundo real com o ambiente virtual. Na Figura 5 observa-se os principais blocos para construção da IoT, são eles:

Figura 5: Blocos básicos na construção do IoT



Fonte: (SANTOS et al., 2016)

- **Identificação:** Esse bloco é responsável por utilizar tecnologias como RFID, NFC e/ou endereçamento IP para identificar objetos e conectá-los à Internet;
- **Sensores/Atuadores:** Conforme mencionado na seção 2.2, os sensores e atuadores são responsáveis por coletar, armazenar, encaminhar e interagir com o ambiente (no caso dos atuadores);
- **Comunicação:** Desempenha papel importante no IoT pois diz respeito as mais diversas formas de conectar os objetos inteligentes com a Internet. As tecnologias mais utilizadas são WiFi, Bluetooth, IEEE 802.15.4 e RFID. Outro fator importante na escolha da comunicação é em relação ao consumo de energia, visto que esses equipamentos não são econômicos nesse aspecto;
- **Computação:** Corresponde às unidades de processamento e controle automático, como é o caso dos microcontroladores, processadores e FPGAs, onde são executados os algoritmos nos objetos inteligentes;
- **Serviços:** Dentre os diversos serviços providos pela IoT, destacam-

se os serviços de identificação, onde são mapeadas entidades físicas em entidades virtuais; serviços de Agregação de dados, que coletam e ordenam dados obtidos através dos objetos inteligentes; e Serviços de Ubiquidade, que visam uma colaboração inteligente e em qualquer momento com seus serviços.

- Semântica: Competência de extrair conhecimento dos objetos na IoT. Para tanto, podem ser usadas diversas técnicas como *Resource Description Framework* (RDF), *Web Ontology Language* (OWL) e *Efficient XML Interchange* (EXI).

3 MATERIAIS E MÉTODOS

Neste capítulo são abordados os dispositivos de *hardware* e as ferramentas de computação em nuvem que foram utilizadas neste projeto. Foi separada em duas partes, a primeira foi o estudo e aquisição do microcontrolador mais adequado, sensores e módulo de comunicação com a Internet. A segunda parte trata da ferramenta de computação em nuvem com a funcionalidade de banco de dados.

3.1 Hardware

Dentre todas as plataformas de *hardware* livre disponíveis no mercado, o Arduino é uma das mais conhecidas e de mais fácil utilização pela ampla gama de conteúdo desenvolvido disponibilizado pela empresa. O modelo Arduino UNO utilizado neste projeto tem seu formato para permitir que outras pessoas possam estender as funcionalidades do mesmo por meio de *shields*, além disso ele possui o microcontrolador ATmega328P, um microcontrolador de fácil acesso no mercado.

Na subseção 3.1.1 deste capítulo será abordado sobre a plataforma Arduino e suas características. Em seguida, na subseção 3.1.2 será explanada a arquitetura do Arduino Uno e seu microcontrolador ATmega328p. Na subseção 3.1.3 será apresentada a plataforma de desenvolvimento IDE, onde os comandos são escritos e enviados para a placa. Na subseção 3.1.4 são listados os sensores adequados para o monitoramento residencial. Por fim, na seção 3.1.5, será exposto sobre o *shield* Ethernet que permite conexão com a rede.

3.1.1 PLATAFORMA ARDUINO

O conceito Arduino de *hardware* código aberto (*open source*) foi desenvolvido pelo visionário grupo integrado por Massimo Banzi, David Cuartiles, Tom Igoe, Gianluca Martino e David Mellis em Ivrea, Itália. O objetivo da equipe era desenvolver uma linha de microcontrolador fácil de usar tanto *hardware*, como

software, de modo que o poder de processamento esteja prontamente disponível para todos (BARRET, 2013). O movimento DIY (*do it yourself*), conforme explica Chris Anderson em seu livro “*Makers: A nova revolução industrial*” se apropriou do Arduino por conta de seu baixo custo e grande gama de possibilidades, criando desde Drones até próteses robóticas. A Figura 6 mostra um Arduino UNO R3, modelo que será usado nesse projeto.

Figura 6: Arduino UNO R3



Fonte: (ARDUINO, 2019)

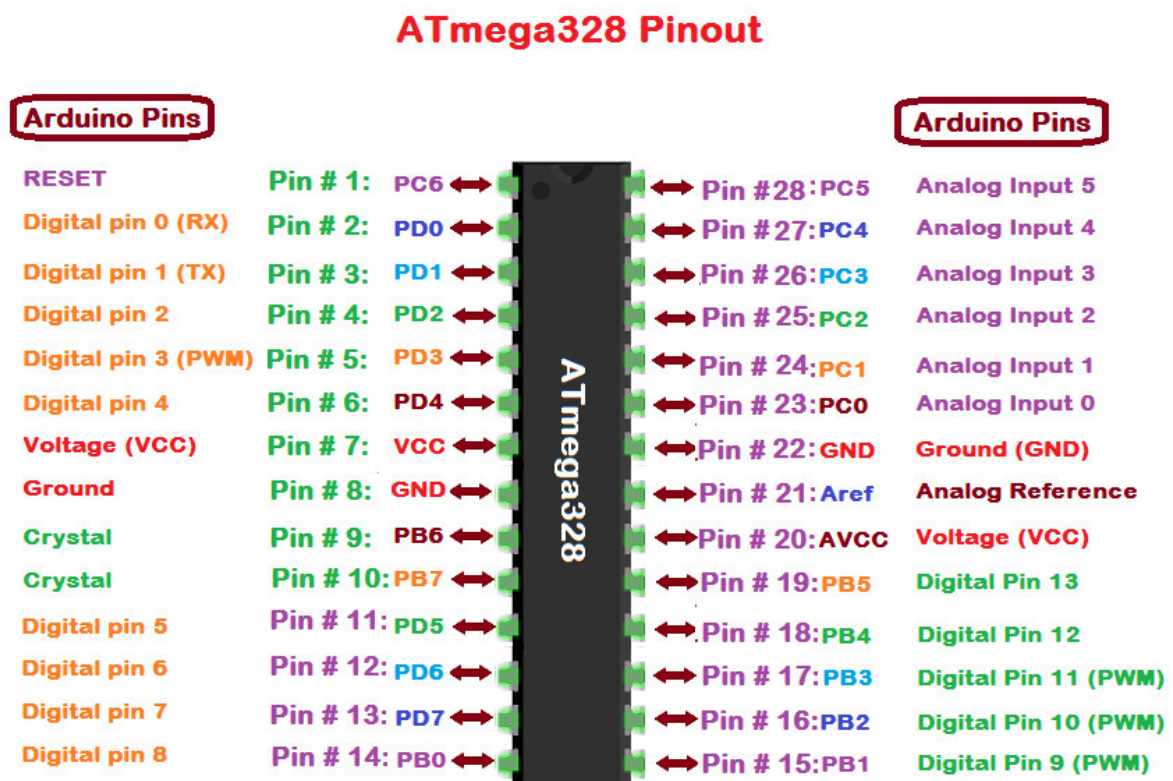
Na versão UNO, uma das mais utilizadas e com um dos melhores custo/benefício, a plataforma acompanha um microcontrolador ATmega328 (MCROBERTS; JOICE; COSTA, 2015). Esse ambiente de desenvolvimento pode receber sinais elétricos de vários sensores e lidar com esses dados para controlar motores, luzes, fechaduras, relés e diversos outros atuadores.

O Arduino também pode ser estendido com a utilização de *shields* (escudos), que são placas de circuito que contêm outros dispositivos (por exemplo, displays de LCD, módulos de Ethernet etc.) que permite conectar à parte superior do seu equipamento para obter funcionalidades extras (MCROBERTS; JOICE; COSTA, 2015).

3.1.2 ARQUITETURA DO ARDUINO UNO

O Arduino UNO conta com um microcontrolador ATMEL ATmega328, que é um dispositivo de 8 bits da família AVR com arquitetura RISC avançada e com encapsulamento DIP28. Ele dispõe de 32 KB de memória Flash, com 512 Bytes reservados para o *bootloader*; 2 KB de memória RAM; e 1 KB de memória EEPROM. Além disso, este microcontrolador pode operar com uma frequência de até 20 MHz, contudo, devido ao cristal externo que está conectado aos pinos 9 e 10 do microcontrolador (Figura 7), ele opera à 16 MHz no Arduino UNO.

Figura 7: Pinos do microcontrolador ATmega328



Fonte: (NASIR, 2016)

Para detalhar a funcionalidade dos pinos do microcontrolador quando conectados à placa Arduino UNO, observa-se na Figura 5, a partir do primeiro pino no canto superior direito da placa, pino digital 0 e seguindo no sentido anti-horário, temos as seguintes funções:

- Pinos 0 e 1: Pinos utilizados para comunicação serial. Esses pinos são ligados ao microcontrolador responsável pela comunicação USB com o computador;
- Pinos 2 a 13: Portas digitais de entrada/saída. Nesses pinos o sinal de saída será de 0 V ou 5 V, de acordo com o código fonte. Dentre esses pinos, tem-se dois pinos de interrupções externas (pinos 2 e 3) e seis pinos que dispõe dos canais PWM da placa (pinos 3, 5, 6, 9, 10, 11);
- Pino GND: Pino referente à terra digital, que utiliza o *hardware* externo como referencial;
- Pino AREF: Pino de entrada. Recebe um sinal de referência externo para o conversor analógico/digital do Arduino. Quando não há referência a placa considera 0 V.
- Pinos de potência: O pino IOREF fornece uma tensão de referência para que os shields selecionem a tensão apropriada para seu funcionamento; o pino RESET está conectado diretamente ao microcontrolador e pode ser utilizado para fazer um reset externo da placa; os pinos 3,3 V e 5 V fornecem este nível de tensão para shields e módulos externos; dois pinos de GND responsáveis pela referência, terra; e o pino VIN usado para alimentar a placa através de shield ou bateria externa;
- Pino Analógicos (A0 a A5): Portas analógicas de entrada e saída. Para interface com o mundo analógico o microcontrolador possui 6 entradas, onde cada uma delas possui a resolução de 10 bits e tensão de 5 V, ou seja, quando a entrada estiver com 5 V, o valor da conversão analógico/digital será de 1023. O valor da referência pode ser mudado através do pino AREF.

3.1.3 SOFTWARE ARDUINO IDE

O Arduino possui a plataforma IDE, que é uma aplicação multiplataforma escrito em JAVA, utilizado para desenvolver, salvar, modificar e executar os

programas e carregá-los na placa Arduino, além de apresentar outros recursos como gravação do *bootloader* em um microcontrolador ATMEGA e um monitor serial.

Esse ambiente de desenvolvimento possui a capacidade de programar em C/C++, a partir da biblioteca *Wiring*. Dessa forma, é possível que sejam criadas facilmente muitas operações de entrada e saída, bastando definir apenas duas funções no *prompt* de comando, são elas:

- *setup ()*: Os códigos dentro dessa função são executados apenas uma vez. Nesse espaço são declaradas as informações iniciais como, por exemplo, se um LED começa desligado ou ligado, quais são os pinos de entrada e saída, entre outras coisas.
- *loop ()*: Ele é iniciado após a execução do *setup ()*. Essa função vai ser executada em um laço infinito ou até que um novo código seja carregado, reiniciando o processo. O programa começa logo após a abertura da chave (*{*), e o processador vai executando cada linha até o fechamento da chave (*}*), em seguida, o programa volta para a primeira linha do *loop* e recomeça a execução.

A ferramenta está disponível para download gratuito na página oficial do Arduino (ARDUINO, 2019), além disso, no site é possível compartilhar e ter acesso a projetos desenvolvidos utilizando a plataforma e conta também com uma ampla lista de bibliotecas e fóruns de suporte ao usuário.

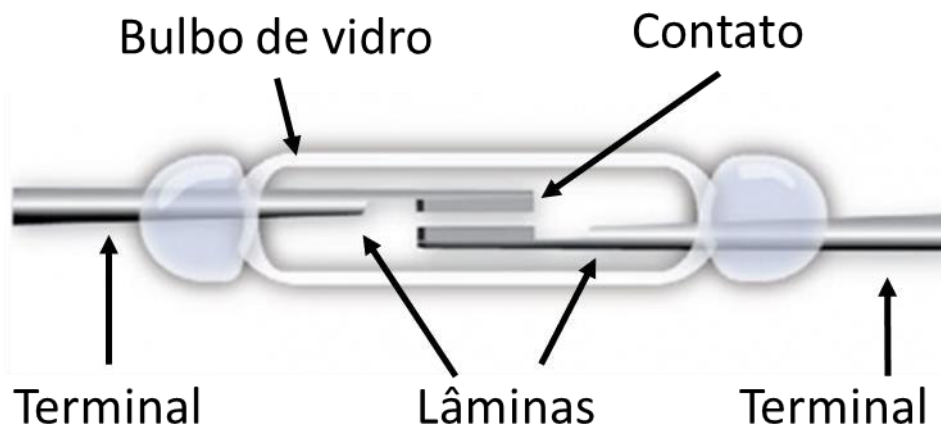
3.1.3 SENSORES

A seguir serão descritos os principais sensores usados no monitoramento residencial. No item “a” será abordado sobre sensores magnéticos e seu funcionamento. Em seguida, no item “b”, será descrito o sensor de chama e por último o sensor PIR no item “c”.

a. *Reed Switch* (Sensor magnético)

Interruptores de lâminas (*reed-switch*) são dispositivos formados por um bulbo de vidro no interior do qual existem lâminas flexíveis feitas de materiais que são sensíveis à ação de campos magnéticos (BRAGA, 2015). O bulbo de vidro é cheio com um gás inerte que evita a ação corrosiva do ar sobre as lâminas. A Figura 8 mostra os elementos do *reed-switch*.

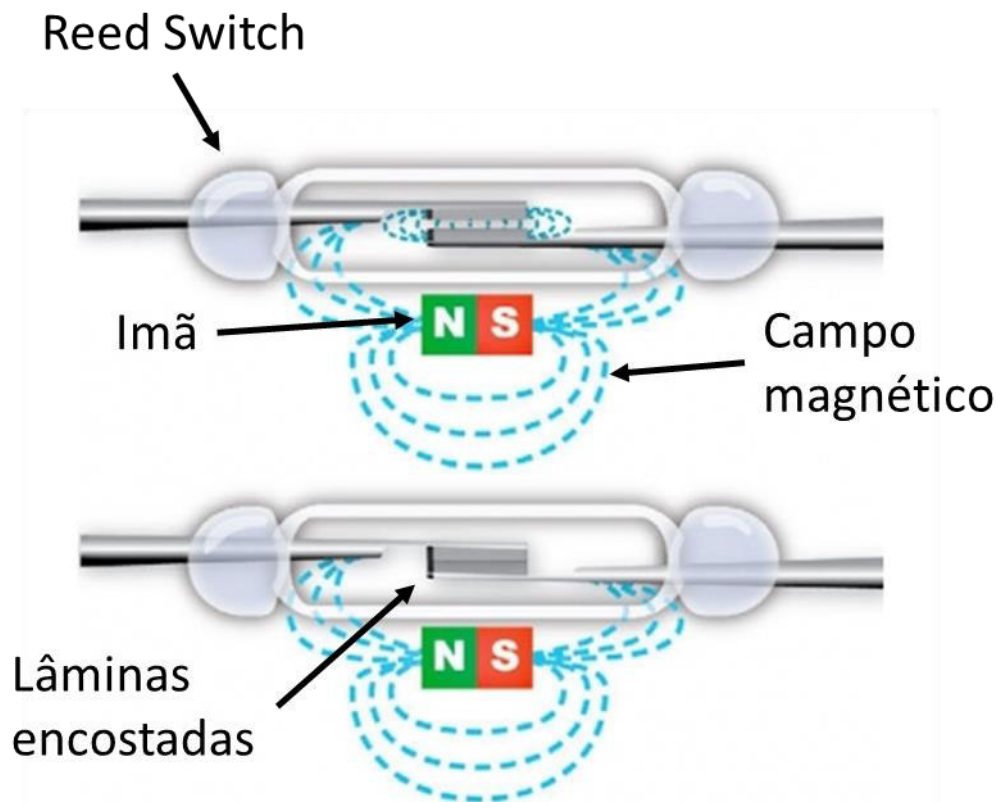
Figura 8: Reed-Switch



Fonte: (BRAGA, 2015)

Em condições normais, ou seja, sem a presença de um campo magnético próximo ao componente, as lâminas encontram-se separadas e nenhuma corrente elétrica circula através do componente, como observado na Figura 5. Quando se aproxima do dispositivo um ímã permanente, a ação do campo magnético faz com que as lâminas magnetizem e com isso se atraiam, unindo-se, dessa forma, fechando o contato elétrico e permitindo a passagem de corrente, conforme observado na Figura 9, na página seguinte.

Figura 9: Reed-Switch na presença de um ímã permanente



Fonte: (BRAGA, 2015)

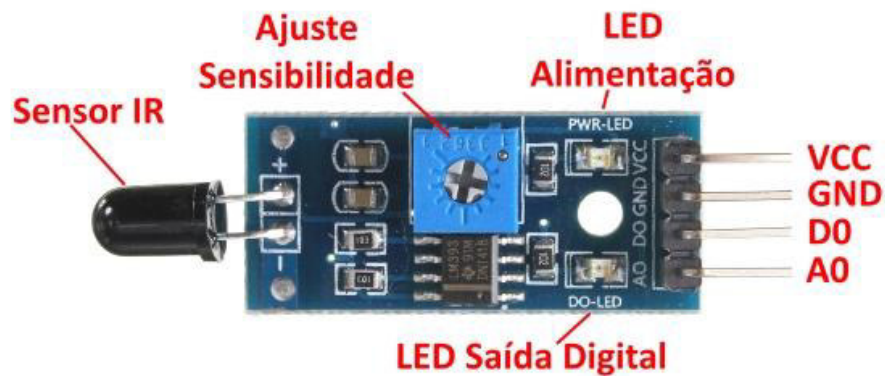
Os *reed-switches* são dispositivos de baixa corrente elétrica, os tipos mais comuns e que são usados na maioria dos circuitos, tem corrente nominal na ordem de 200 mA, assim, seu uso é limitado para circuitos de baixa potência, nunca devendo ser usados para controlar diretamente cargas maiores como motores, solenoides, lâmpadas ou qualquer equipamento que possuam correntes maiores que os valores indicados (BRAGA, 2015).

Neste trabalho este dispositivo será usado como sensor de abertura, ou seja, na presença de um ímã próximo, o circuito enviará a informação que os contatos foram fechados, variando a forma natural indicando o estado de aberto e fechado. Propõe-se que sejam colocados na estrutura móvel das portas e janelas um ímã permanente capaz de acionar o sensor, que estará disposto na parte fixa da porta, no portal. Assim, quando a porta estiver fechada, os contatos estarão fechados. Já quando a porta se abrir, os contatos são abertos, devido à ausência do campo magnético, mudando o estado do sensor.

b. Sensor de chama

O sensor de chama permite que seja verificada a presença de fogo ou fontes de calor por meio de um sensor infravermelho (IR, *infrared radiation*), que detecta luz com comprimento de onda entre 760 nm e 1100 nm e com distância de até 1m do sensor (THOMSEN, 2015). O ângulo de detecção pode chegar a 60° e a sensibilidade de detecção pode ser ajustada a partir de um pequeno potenciômetro (trimpot) que se encontra na placa, dessa forma tornando o módulo mais prático. Seu princípio de funcionamento é baseado na tecnologia infravermelha (Receptor Infravermelho de alta sensibilidade). A leitura de sinal do módulo pode ser feita através de uma porta analógica ou através de uma porta digital. O módulo possui furo para fixação. A Figura 10 mostra o sensor de chama e a especificação de cada um de seus pinos.

Figura 10: Sensor de chama



Fonte: (THOMSEN, 2015)

Apesar de ser muito eficiente, o módulo possui algumas limitações no que diz respeito às chamas produzidas por alguns gases, tais como o gás natural e o GLP. Logo, chamas produzidas por gás natural e o GLP não são detectadas por esse sensor.

Quando há fogo, a saída do sensor fica em estado baixo (0) e quando não há detecção em estado alto (1). Este limite pode ser ajustado através do

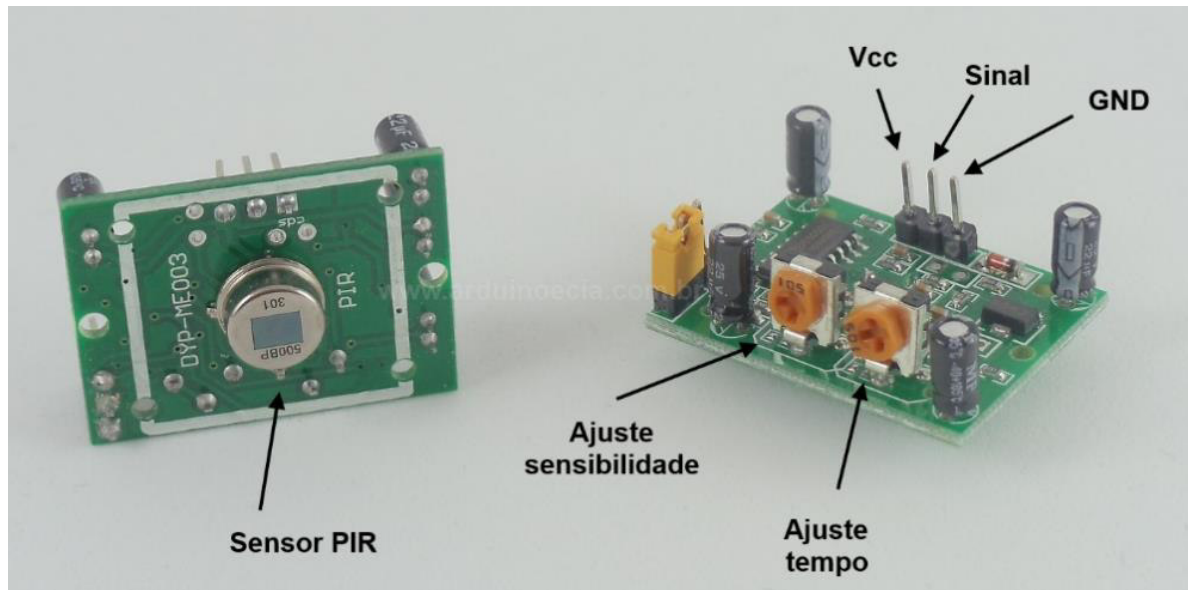
potenciômetro presente no sensor que regulará a saída digital D0. Contudo, para ter uma resolução melhor é possível utilizar a saída analógica A0 e conectar a um conversor A/D, como a presente no Arduino, por exemplo.

Neste trabalho a alocação desse sensor na residência será próximo a equipamentos que utilizam fogo ou possuem algum risco inflamável, como é o caso de fogões, exaustores, quadro de distribuição de energia, botijão de gás ou ambientes que armazenam líquidos inflamáveis.

c. Sensor de movimento (PIR)

Os sensores PIR (*passive infrared radiation*) são usados para detectar se um humano entrou ou saiu da faixa de detecção do sensor. Eles são pequenos, baratos, de baixa potência, fáceis de usar e não se desgastam com muita facilidade. Por essa razão, eles são comumente usados em residências ou empresas. São chamados também de sensores PIR, "infravermelho passivo", "piroelétrico". Esse sensor possui em si dois slots, em que cada um é composto de um material sensível ao IR e esse conjunto é envolvido por uma lente Fresnel, responsável por ampliar a faixa de detecção (ADAFRUIT, 2010). A Figura 11, na página seguinte, mostra o sensor PIR compatível com a plataforma Arduino e utilizado nesse trabalho.

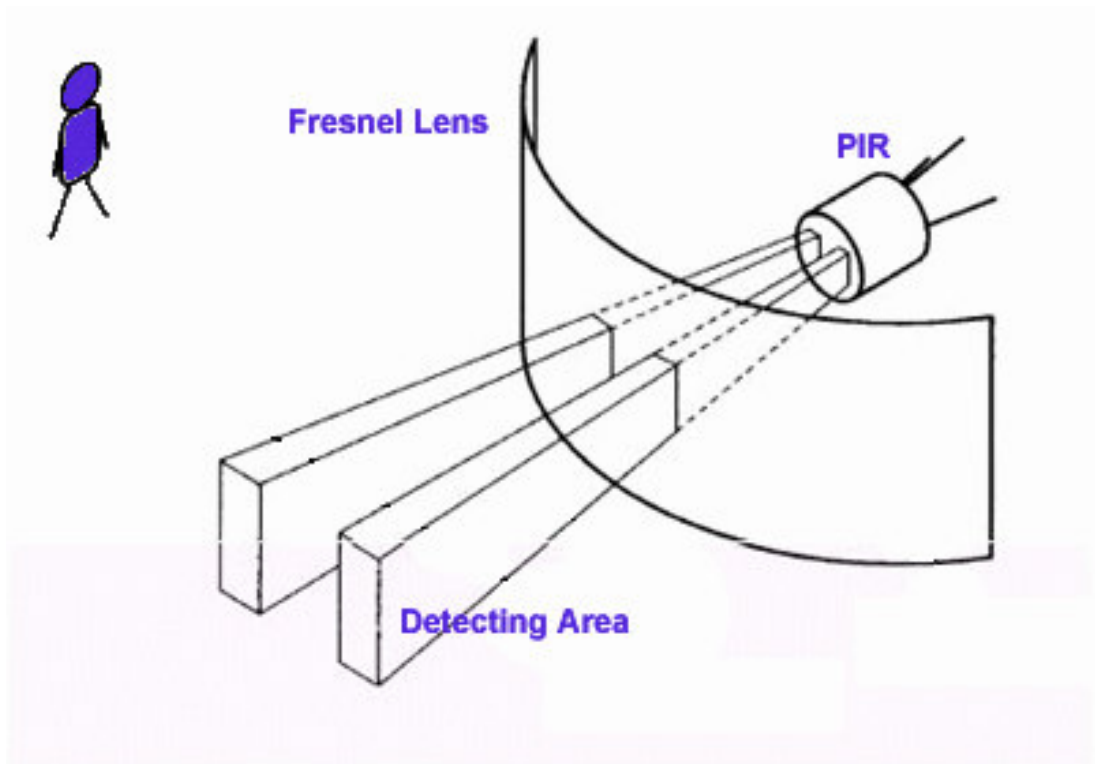
Figura 11: Sensor PIR



Fonte: (FELIPEFLOP, 2019)

Quando o sensor está ocioso, ou seja, quando ambos os slots detectam a mesma quantidade de IR, significa que não há um corpo quente na zona de observação de movimento. Quando um corpo quente, como um humano ou animal, passa, primeiro intercepta uma metade do sensor PIR, o que provoca uma mudança diferencial positiva entre as duas metades e em seguida passa pela área de detecção da outra metade, fazendo com que ocorra o inverso, gerando uma mudança diferencial negativa do pulso de detecção, caracterizando, assim, a ocorrência do movimento. Por fim, quando ambos detectam a mesma quantidade de IR, o sensor volta para o estado ocioso (ADAFRUIT, 2010). A Figura 12, na página seguinte, ilustra a detecção do movimento pelo sensor PIR.

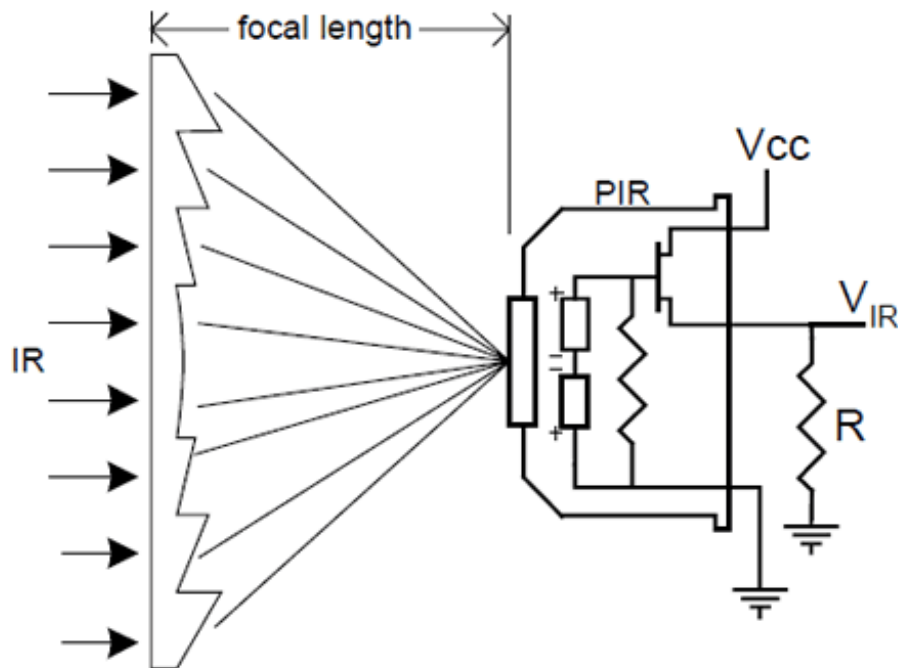
Figura 12: Funcionamento do sensor PIR



Fonte: (ADAFRUIT, 2010)

A Figura 13, na página seguinte, ilustra a utilização do sistema com uma lente Fresnel que limita a área de detecção a apenas dois retângulos. Esse tipo de acessório condensa a luz, fornecendo uma faixa mais ampla de IR ao sensor e pode ser fabricada de plástico.

Figura 13: Lentes Fresnel no sensor PIR

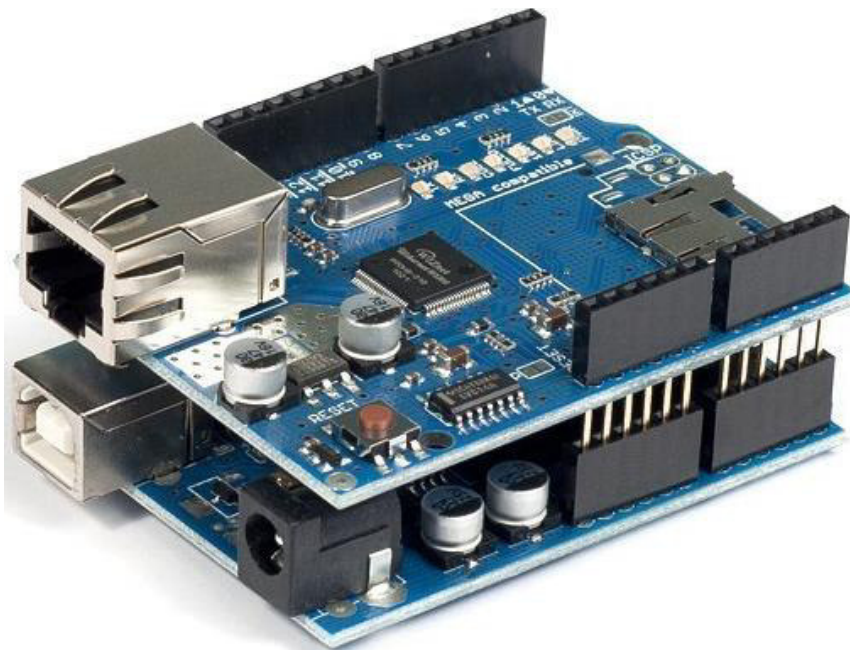


Fonte: (ADAFRUIT, 2010)

3.1.4 SHIELD ETHERNET

O Arduino UNO não possui conexão com a Internet, mas para isso utilizam-se módulos que permitem essa conexão, entre as alternativas está o shield Ethernet, que, por sua vez, conecta-se à Internet através de um cabo RJ45. Esse módulo possui o W5100 como chip de interface de rede de chip único, com um controlador Ethernet de 10/100Mbps integrado internamente. É usado principalmente em sistemas integrados de alto desempenho e baixo custo. Além disso, o chip permite conectar-se à Internet sem a necessidade de um sistema operacional. Também é compatível com IEEE802.3 10BASE-T e 802.3u 100BASE-TX. Na Figura 14, na página seguinte, pode-se visualizar como esse módulo é acoplada no Arduino.

Figura 14: Shield Ethernet acoplado no Arduino UNO



Fonte: (CABALLERO, 2016)

O W5100 integra uma pilha de protocolos TCP/IP de hardware completo que incluem a camada de controle de acesso à mídia Ethernet (MAC) e camada física (PHY). A pilha de protocolos TCP/IP de hardware suporta os seguintes protocolos: TCP, UDP, IPV4, ICMP, ARP, IGMP e PPOE. Além disso, o W5100 também integra 16 KB de memória para transmissão de dados. Para o W5100, não é necessário considerar o controle da Ethernet apenas a programação do socket.

3.2 Computação em nuvem

Nesta seção será apresentado sobre a plataforma de computação escolhida para este trabalho, o Firebase, que atualmente pertence ao Google e para utilizar as ferramentas disponíveis é necessário criar uma conta no site (GOOGLE, 2019b).

3.2.1 FIREBASE

O Firebase é uma plataforma de desenvolvimento de aplicações para dispositivos móveis e web, criada por uma empresa de mesmo nome em 2011 e adquirida pelo Google em 2014 (GOOGLE, 2019b). Atualmente a plataforma possui 18 produtos e é utilizado por cerca de 1,5 milhões de aplicativos (MA, 2018). Os principais produtos oferecidos são: Teste A / B, Indexação de Aplicativos, Analytics, Autenticação, Cloud Firestore, Cloud Functions, Cloud Messaging, Cloud Storage, Crashlytics, Links Dinâmicos, Hospedagem, Mensagens no Aplicativo, Kit ML, Monitoramento de Desempenho, Previsões, Banco de Dados em Tempo Real, ConFiguração Remota Laboratório de Testes.

Dentre os produtos ofertados pelo Firebase, o *Realtime Database*, que fornece um banco de dados em tempo real, foi utilizado neste trabalho. Esse serviço providencia aos desenvolvedores de aplicativos uma API que permite que os dados de aplicativos sejam sincronizados entre os clientes e armazenados na nuvem do *Firebase* (FARR, 2013). A empresa fornece bibliotecas de cliente que permitem a integração com Android, iOS, JavaScript, Java, Objective-C, Swift e Node.js. O banco de dados também é acessível por meio de uma API REST e ligações para várias estruturas Java Script. A API REST usa o protocolo *Server-Sent Events*, que é uma API para criar conexões HTTP para receber notificações *push* de um servidor.

4 RESULTADOS

Neste capítulo são apresentados os resultados e discussões do projeto proposto. Na seção 4.1 serão apresentadas as características gerais do protótipo. Em seguida, na seção 4.2, o esquema de montagem de um protótipo do sistema de monitoramento. Seguindo da implementação do projeto no LABGEA na seção 4.3.

4.1 Características Gerais

O Arduino UNO R3 possui um posicionamento padrão de pinos que permitem que diversos módulos sejam acoplados nele, como é o caso do módulo Ethernet, que foi montado conforme ilustrado na Figura 14. Essa configuração permite que os pinos responsáveis pela comunicação SPI (*Serial Peripheral Interface*) sejam conectados automaticamente, devendo apenas ter a cautela de não utilizar essas portas durante a montagem do sistema.

Monaro (2007, p. 11) define SPI como, “uma interface de dados serial síncrona padronizada pela Motorola que funciona em modo *full-duplex* (transmissão e recebimento de informações simultaneamente). Os equipamentos se comunicam em modo mestre/escravo, onde o equipamento mestre inicializa a comunicação”.

Todos os sensores conectados precisam ser monitorados constantemente pelo Arduino. Assim, quando um sensor muda seu estado, o Arduino precisa tomar uma decisão sobre o evento ocorrido. Feito isso, os dados são enviados para um banco de dados na nuvem através da conexão com a Internet e, por fim, sinalizados através de um aplicativo de celular para o usuário final.

4.2 Protótipo

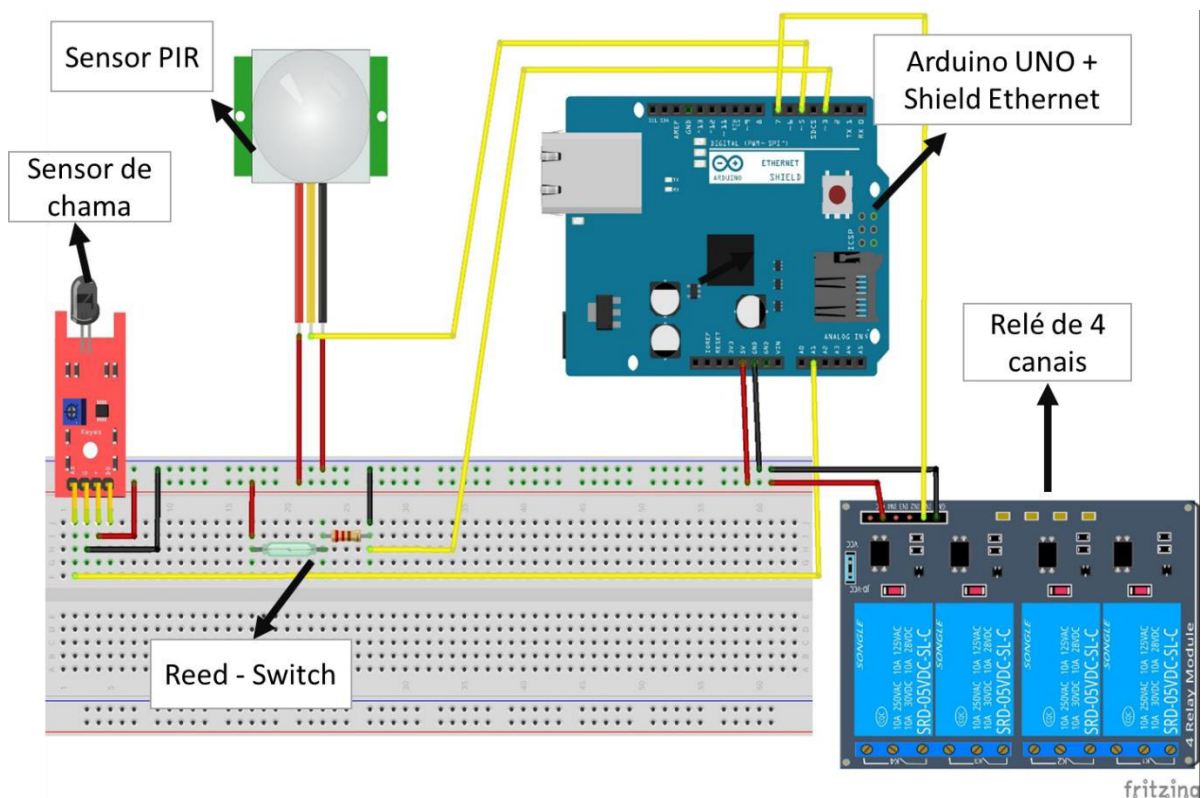
Nesta seção será descrito a montagem do protótipo do sistema de monitoramento residencial em protoboard, onde foram realizados ajustes e parâmetros para que o sistema pudesse ser futuramente implementado.

4.2.1 HARDWARE

Para validação da funcionalidade dos sensores e dos dispositivos em geral, em primeiro momento, foi simulado um protótipo do projeto em *protoboard*. Dessa forma, torna-se mais acessível a configuração do sistema e o desenvolvimento do mesmo.

Foram utilizados um sensor de presença (PIR), um sensor detector de chama e um reed switch, este para detecção de abertura de portas. O esquema de ligação dos sensores com o Arduino pode ser verificado na Figura 15, da página seguinte, onde os fios em vermelho representam a conexão Vcc, os fios em preto a conexão *ground* e em amarelo as conexões de controle que são ligadas nos pinos digitais e/ou analógicos do Arduino.

Figura 15: Esquema de montagem dos sensores no Arduino em *protoboard*



Fonte: (Autor, 2019)

A ligação entre os pinos do módulo Ethernet, módulo relé e dos sensores com o Arduino foi realizada conforme a configuração exibida nas tabelas 2, 3 e 4, respectivamente:

Tabela 2: Conexão entre o módulo Ethernet e o Arduino

Módulo Ethernet	Arduino UNO
Pino MISO (<i>Master In Slave Out</i>)	Ligado no pino 12 do Arduino
Pino MOSI (<i>Master Out Slave In</i>)	Ligado no pino 11 do Arduino
Pino SCK (<i>Serial Clock</i>)	Ligado no pino 13 do Arduino
Pino SS (<i>Slave Select</i>)	Ligado no pino 14 do Arduino

Fonte: (WIZNET, 2008)

Tabela 3: Conexão entre o módulo relé e o Arduino

Módulo Relé	Arduino UNO
Pino Vcc	Ligado no pino 5 V do Arduino
Pino GND	Ligado no pino GND do Arduino
Pino N1	Ligado no pino 7 do Arduino

Fonte: (Autor, 2019)

Tabela 4: Conexão dos sensores com o Arduino

Sensores	Arduino UNO
Sensor de Presença (PIR)	Ligado no pino 5 do Arduino
Sensor de chama	Ligado no pino A1 do Arduino
Reed switch	Ligado no pino 3 do Arduino

Fonte: (Autor, 2019)

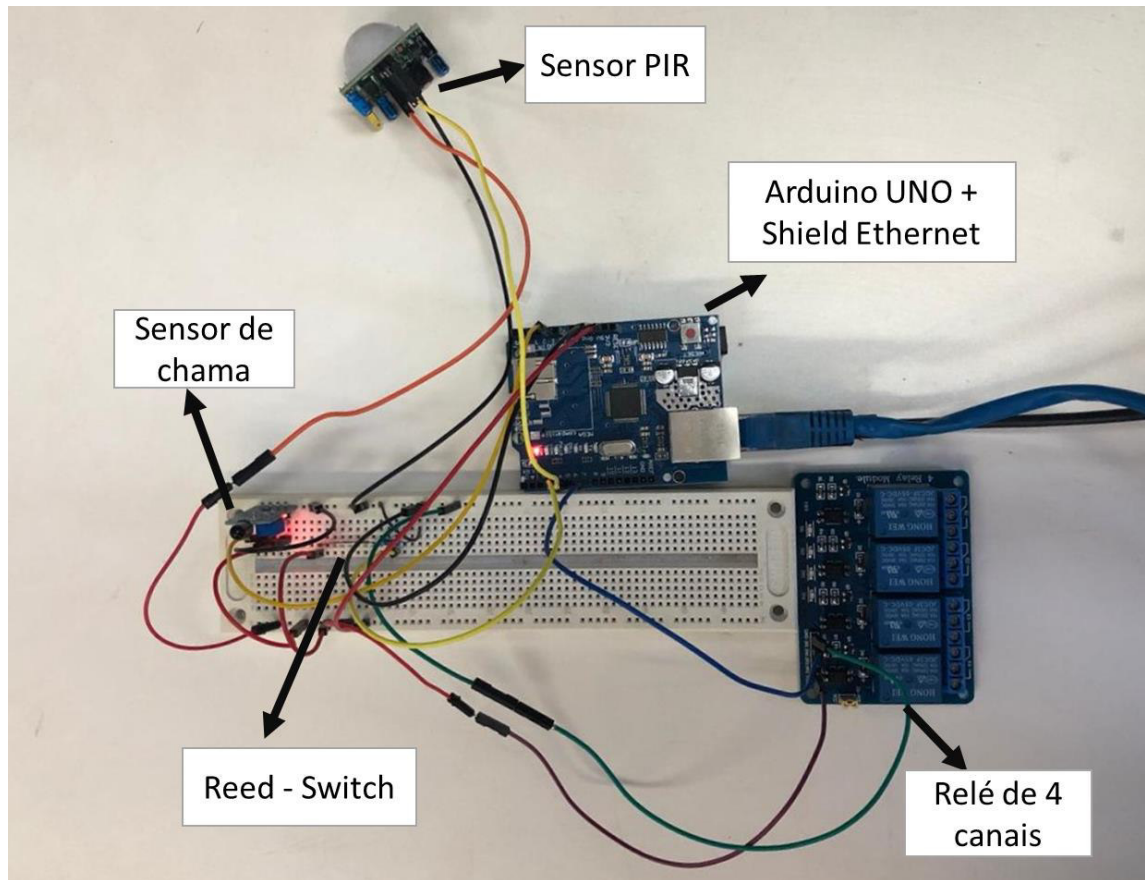
O sistema de monitoramento permite que o Arduino possa coletar informações quando há mudança no estado dos sensores, onde 0 (zero) significa ausência de evento e 1 (um) o seu acontecimento. No caso do Reed Switch, quando há a presença de campo magnético próximo ao sensor, a indicação é que a porta (ou janela) está fechada, pois o ímã preso à porta encontra-se próximo ao *Reed Switch* preso ao portal. Assim, quando a porta se abre e o ímã se afasta do sensor, o Arduino reconhece a alteração do estado e sinaliza na janela serial do Arduino IDE.

Com o sensor de detecção de chama o processo ocorre de forma semelhante. Antes de tudo é necessário calibrar sua sensibilidade, para que situações do ambiente não indiquem alarmes falsos, isso pelo fato do sensor detectar a intensidade da radiação infravermelha emitida pelos corpos quentes, assim, janelas de vidro voltadas para o sol próximas a esse sensor podem interferir na sua detecção. Como foi utilizada a saída analógica do sensor, essa calibração pôde ser feita através do programa, onde primeiro verifica-se a radiação detectada no ambiente a ser instalado e em seguida ajusta o código para sinalizar a presença de fogo caso esse valor seja ultrapassado. Assim, quando há a presença de fogo no seu campo de atuação, esse limite é superado e o Arduino sinaliza a mudança de estado.

O sensor de presença, além de indicar o estado onde há ou não movimento na sua faixa de detecção, está conectado a um módulo relé que permite com que algumas ações possam ser feitas na ocorrência do movimento. O sensor PIR também precisa ser calibrado, onde é ajustada manualmente a sensibilidade e o tempo de inatividade pós detecção. Dessa forma, quando ocorre movimento, o Arduino é sinalizado e o módulo relé executa uma ação, podendo ser o acionamento de uma lâmpada, registro fotográfico ou execução de mensagem sonora.

Assim como exposto na simulação da Figura 15, foi realizada montagem do protótipo em *proto-board*, visando garantir o correto funcionamento e realizando os ajustes analógicos nos sensores, como é o caso da sensibilidade e tempo de atraso do sensor PIR. Na Figura 16 (página seguinte) pode-se observar a montagem do protótipo, nela estão sinalizados os sensores, Arduino, módulo Ethernet, alimentação USB e conexão com a rede.

Figura 16: Montagem do sistema em protoboard

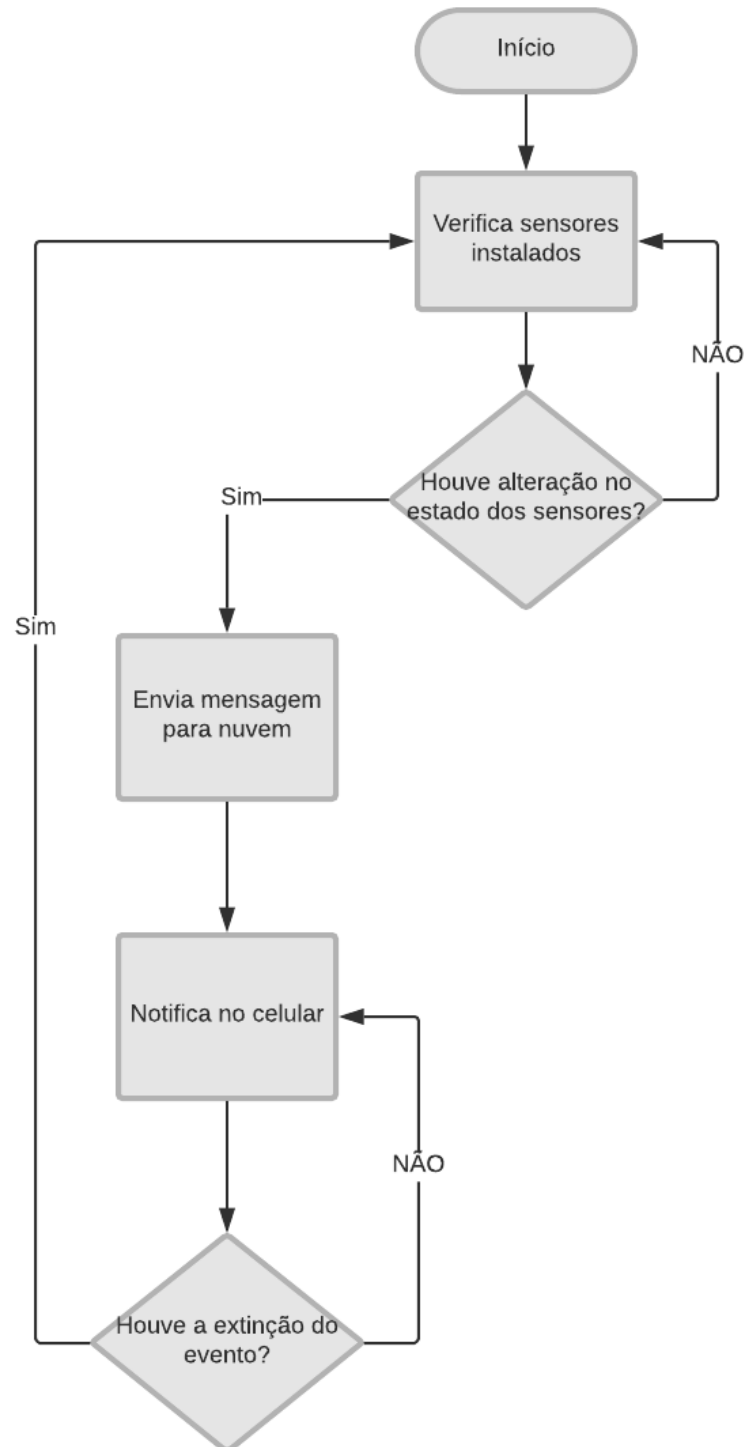


Fonte: (Autor, 2019)

4.2.2 SOFTWARE

O Arduino é responsável por todo processamento deste sistema embarcado e associado a ele está o módulo Ethernet, responsável por conectá-lo a rede de computadores. O código segue o esquema demonstrado no fluxograma da Figura 17 (página a seguir). O algoritmo referente a essa lógica encontra-se no Apêndice deste trabalho.

Figura 17: Fluxograma do código do sistema de monitoramento residencial



Fonte: (Autor, 2019)

No código é incluída a biblioteca “Ethernet.h”, responsável por trabalhar com o *Ethernet Shield* ou outros dispositivos com finalidades semelhantes, inserindo algumas funções para permitir que o Arduino se conecte à Internet.

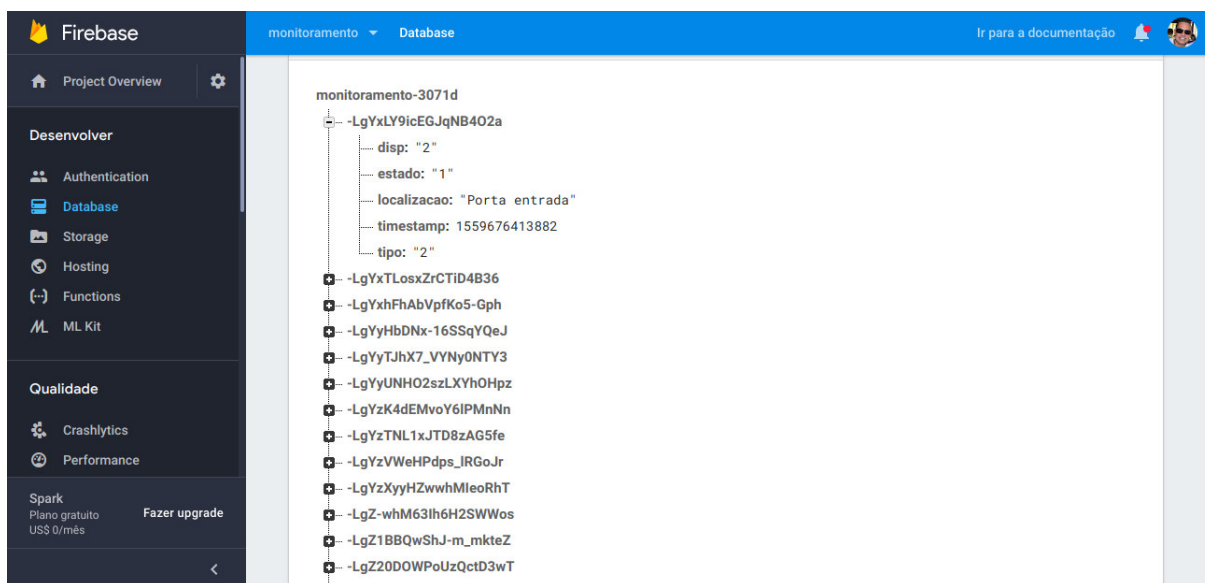
A primeira etapa da execução do código define o MAC, endereço físico associado à interface de comunicação, que conecta o dispositivo à rede. Posteriormente, define-se o servidor e um endereço de IP fixo, caso a conexão DHCP falhe. A conexão com o servidor é então estabelecida, dando seguimento para o envio para a nuvem dos dados oriundos da leitura dos sensores, estes iniciados depois que a conexão for estabelecida, mantendo ininterruptamente o monitoramento.

Para a leitura dos sensores é feita uma condicional SE, onde indica a existência ou ausência do evento monitorado, enviando para a nuvem quando o estado do sensor for alto (1) e quando voltar ao estado inicial (0). Contudo, como o sensor de chama está conectado em porta analógica, deve ser verificado a leitura média no ambiente em que ele for instalado, para que só seja sinalizado a existência do evento quando esse limite for seguramente ultrapassado, indicando presença de fogo no local.

4.2.3 COMPUTAÇÃO EM NUVEM

Ao passo que os dados da leitura dos sensores são processados pelo Arduino, eles são enviados para nuvem, através da conexão com a Internet, especificamente para o módulo *Realtime Database* do Firebase. Todos os parâmetros enviados encontram-se disponíveis na página do Firebase, após a identificação com *login* e senha, conforme na Figura 18, na página seguinte.

Figura 18: Dados no Firebase



Fonte: (Autor, 2019)

4.2.4 APLICAÇÃO PARA ANÁLISE DE DADOS

O Aplicativo que irá receber os dados do módulo Arduino através do Firebase foi desenvolvido na plataforma Android Studio do Google (ANDROID STUDIO, 2019). Esta plataforma é um ambiente de desenvolvimento integrado (IDE, *Integrated Development Environment*) que permite o desenvolvimento de aplicativos para a plataforma Android e é disponibilizado gratuitamente sob a Licença Apache 2.0 (APACHE, 2019). A ferramenta está disponível em multiplataformas, permitindo assim seu uso em sistemas operacionais como Windows, OS X/Mac OS e Linux.

O Android Studio permite o desenvolvimento de aplicativos para a plataforma Android que podem ser concebidos nas linguagens orientadas a objeto Java, Kotlin, e C++ usando o Android NDK. A Java foi inicialmente desenvolvida pela Sun Microsystems em 1995, e é usado em múltiplas plataformas de hardware, sendo uma das mais populares linguagens de programação usadas atualmente (JAVA, 2019). O código Java é executado por uma máquina virtual presentes nos dispositivos Android. A Kotlin gera códigos que também são executados na mesma máquina virtual e possui como principais vantagens requerer menos código e tratar alguns erros de forma automática, o que facilita a programação (KOTLIN, 2019). A C++ é executada

nativamente nos dispositivos, que oferece mais controle sobre o hardware e um desempenho melhorado (DEVELOPERS, 2019).

A linguagem escolhida para o desenvolvimento da aplicação foi a linguagem Java. O App, que foi nomeado Monitoramento nas Nuvens, é um aplicativo simples e foi desenvolvido apenas como prova de conceito para consumir os dados disponíveis na nuvem. Por ser um produto de Computação em nuvem, assim como é esperado, a integração do App com o *Realtime Database* é simples e feita em poucos passos (GOOGLE, 2019b). A seguir descreve-se estes passos:

- 1) Configurar através do console do Firebase o seu Aplicativo e suas credenciais, salvando na raiz do código do projeto Android. Este processo é feito através da IDE Android Studio e todas as informações são salvas de forma simplificada, abstraindo toda a complexidade da configuração.

- 2) Instalar o SDK do Firebase

Para instalar a SDK, basta adicionar a dependência do *Realtime Database* ao arquivo de configuração *build.gradle* no nível de aplicativo:

```
implementation 'com.google.firebase:firebase-database:16.1.0'
```

- 3) Recuperar as instâncias dos dados armazenados através de uma referência ao banco:

```
FirebaseDatabase bancodedados = FirebaseDatabase.getInstance();
DatabaseReference minhaReferencia = bancodedados.getReference("/");
```

- 4) Ler o banco de dados

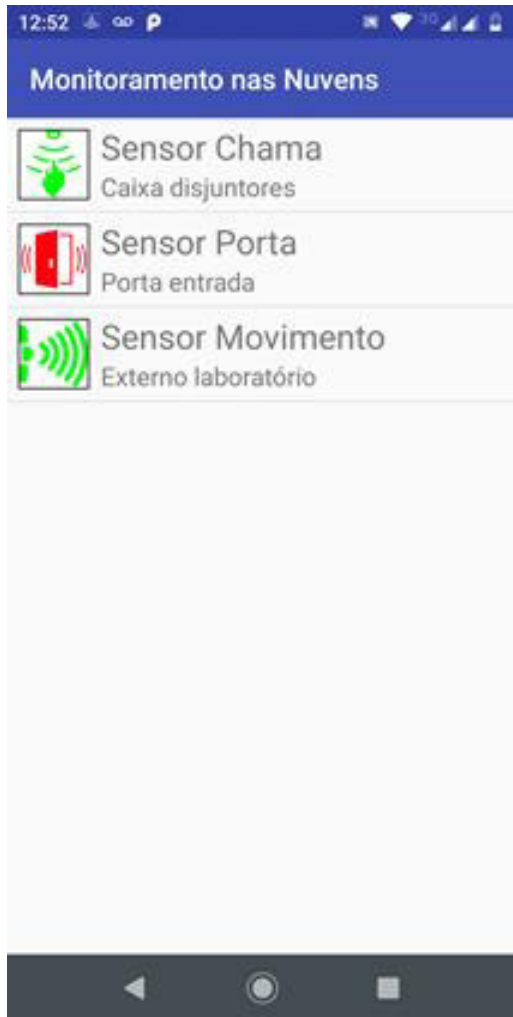
Os dados no *Realtime Database* são disponíveis assim que são gerados. Para que os dados do aplicativo sejam atualizados em tempo real, adiciona-se um “ValueEventListener” à referência criada anteriormente. Esta estrutura segue o Padrão de projeto “Observer”, onde um programa se inscreve para receber atualizações de um dado específico. Para obter informações das modificações ocorridas no banco de dados, o método “onDataChange()” é estendido e sempre que houver alteração, refletirá nos dados apresentados pelo App. Segue o código exemplo:

```
// Ler do banco de dados
myRef.addValueEventListener(new ValueEventListener() {
    @Override
```

```
public void onDataChange(DataSnapshot dataSnapshot) {  
    String jsonValue = dataSnapshot.getValue(String.class);  
    atualizaDados(jsonValue)  
}  
  
@Override  
public void onCancelled(DatabaseError error) {  
    Log.w(TAG, "Ocorreu falha ao ler os dados enviados.",  
error.toException());  
}  
});
```

Após estes passos serem executados, toda e qualquer modificação será atualizada em tempo real pelo aplicativo. Na Figura 19 (página 51) tem-se a tela inicial do aplicativo Monitoramento nas Nuvens, preenchida com os sensores disponíveis e os locais de monitoramento. Os atuadores do sistema embarcado possuem uma representação gráfica no aplicativo, caso esteja sinalizado na cor verde, não há nenhuma ocorrência e a cor vermelha representa o contrário (Figura 19). Quando selecionado o sensor, o usuário pode verificar o histórico de ocorrências, conforme Figura 20 (página 51). Além disso, o usuário será notificado com data e hora caso ocorra alguma alteração de estado, conforme Figura 21 (página 52).

Figura 19: Tela inicial do aplicativo



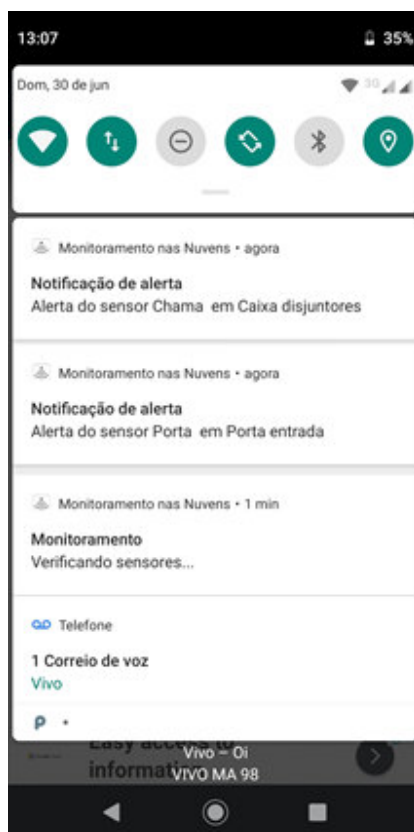
Fonte: (Autor, 2019)

Figura 20: Histórico de ocorrências



Fonte: (Autor, 2019)

Figura 21: Tela de notificações

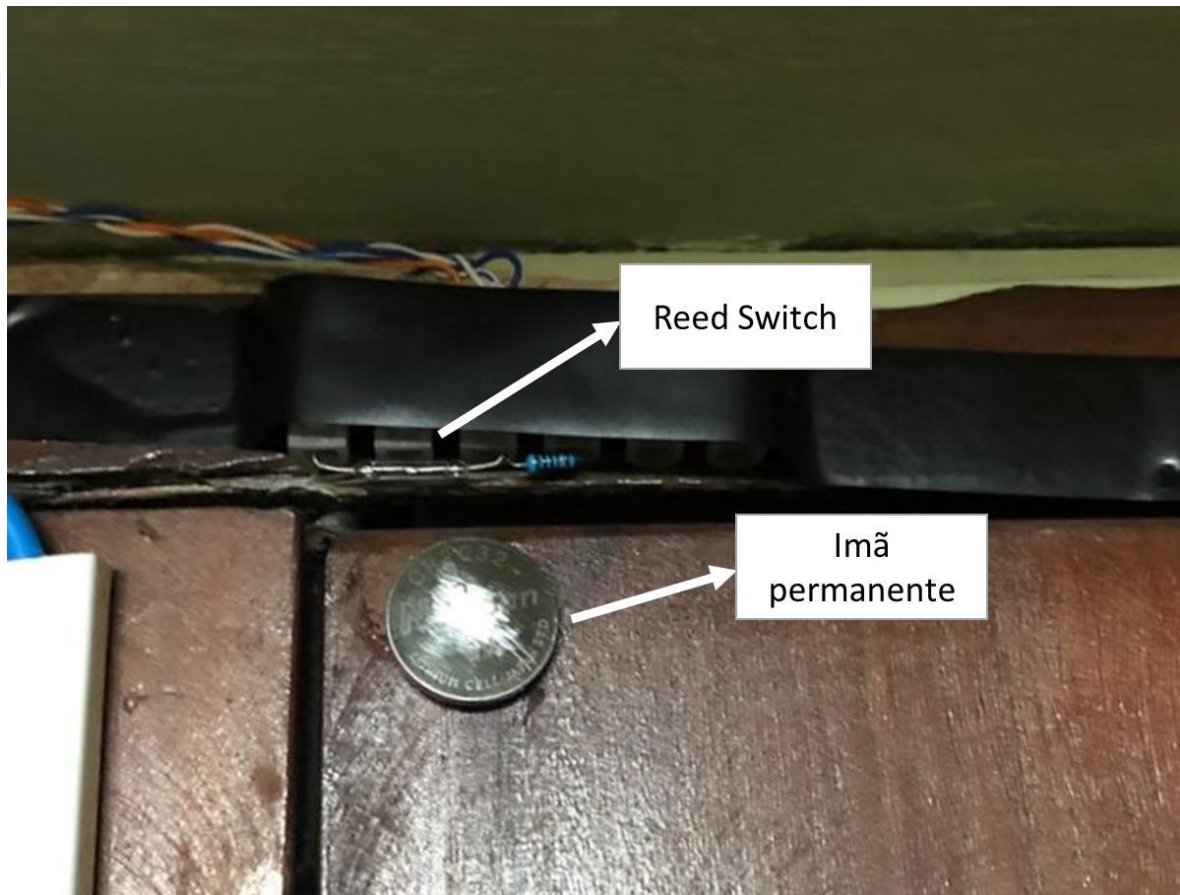


Fonte: (Autor, 2019)

4.3 Implementação no LABGEA

O Laboratório do Grupo de Eletrônica Aplicada (LABGEA) da UFMA é um dos mais importantes e antigos centros de pesquisa do curso de Engenharia Elétrica, local onde este projeto de monografia foi desenvolvido. Visando tornar esse modelo executável e o laboratório mais moderno, foram utilizados para o monitoramento do mesmo um reed switch na porta principal (Figura 22, página 52), um sensor de chama sob o quadro geral de energia (Figura 23, página 53) e o sensor de presença conectado ao relé no lado externo à porta (Figura 24, página 54).

Figura 22: Reed Switch na porta principal do LABGEA



Fonte: (Autor, 2019)

Figura 23: Sensor de chama sobre o quadro elétrico do LABGEA



Fonte: (Autor, 2019)

Figura 24: Sensor de presença



Fonte: (Autor, 2019)

5 CONCLUSÃO

A inspiração do tema deste projeto surgiu após a constatação de que a segurança residencial tem se tornado cada vez mais uma prioridade do brasileiro – veja-se as notícias frequentes e os boletins sobre assaltos à casas e acidentes relacionados ao vazamento de gás e curto-circuito. Diante dessa perspectiva, esta proposta de baixo custo serve como alternativa para o monitoramento de residências e salas.

Tendo instalado o sistema aqui proposto, o usuário pode visualizar à distância e em tempo real a situação do seu local monitorado através da Internet, seja pelo uso do aplicativo para celular ou pelo banco de dados do Firebase. Assim, em caso de detecção por algum dos sensores instalados, o usuário pode remotamente saber qual cômodo e que tipo de sinistro houve, podendo tomar atitudes de imediato, como acionar a polícia ou o corpo de bombeiros. Além disso, o sistema prevê o armazenamento das informações, possibilitando que o usuário tenha disponível o histórico de ocorrências de cada sensor.

A implementação realizada no LABGEA, no ato da confecção deste trabalho, mostrou estabilidade no sistema, tornando-o passível de ser implementado em outros lugares. Além disso, o aplicativo pode ser acessado por mais de um aparelho celular, podendo, portanto, ser disponibilizado aos integrantes do laboratório.

A plataforma Arduino do sistema proposto é totalmente *open source* tanto em *hardware*, quanto em *software*, possibilitando o desenvolvimento dos algoritmos, isso implica em um baixo custo na aquisição do equipamento. Os sensores possuem baixo custo, principalmente se forem adquiridos em quantidade relevante e o módulo para conexão com a Internet custa em média o mesmo preço da placa Arduino. O projeto todo implementado no LABGEA custou cerca de R\$ 100,00.

Ao fim da realização deste trabalho foi possível atingir os objetivos propostos, realizando estudos e colaborando para a área da segurança doméstica. Foram utilizados ferramentas livres e sensores para que os objetivos fossem atingidos. Assim, depois de implementado e testado, o sistema apresentou resultados satisfatórios.

5.1 Trabalhos futuros

Tendo em vista a maior eficiência dos produtos para o usuário final, pode-se realizar alguns ajustes para aperfeiçoar o projeto. A experiência pode ser aprimorada com uma interface mais intuitiva com o usuário, mostrando a planta da casa onde os sensores estão instalados. Além disso, no aplicativo, poderia existir um campo onde fosse possível alertar a polícia ou o corpo de bombeiros quando houvesse confirmação de um sinistro.

Para que o preço do serviço seja ainda mais atrativo, pode-se procurar por alternativas menos dispendiosas que substituam, com os mesmos recursos, o Arduino. Além disso, para uma automação residencial totalmente completa, pode-se adicionar ao sistema de monitoramento um controle de iluminação e equipamentos, integrando-os em uma só aplicação.

A conexão com a Internet neste projeto foi estabelecida através do *shield* Ethernet, contudo, para uma maior facilidade de conexão e economia de fios, pode-se usar um módulo que estabeleça comunicação remota, tanto com a rede de computadores, como com os sensores. Esse tipo de configuração permite, inclusive, uma maior flexibilidade da localização dos sensores.

REFERÊNCIAS

ABESE. **ABESE - Pesquisa ABESE mapeia mercado de segurança eletrônica.**

Disponível em: <<https://abese.org.br/index.php/412-pesquisa-abese-mapeia-mercado-de-seguranca-eletronica>>. Acesso em: 23 fev. 2019.

ABI RESEARCH. **1.5 Million Home Automation Systems Installed in the US This**

Year | ABI Research. Disponível em: <<https://www.abiresearch.com/press/15-million-home-automation-systems-installed-in-th/>>. Acesso em: 2 maio. 2019.

ADAFRUIT. PIR Motion Sensor Tutorial. 2010.

ANDROID STUDIO. **Download Android Studio and SDK tools.** Disponível em:

<<https://developer.android.com/studio>>. Acesso em: 1 jul. 2019.

APACHE. **Apache License, Version 2.0.** Disponível em:

<<http://www.apache.org/licenses/LICENSE-2.0>>. Acesso em: 1 jul. 2019.

ARDUÍNO. **Arduíno.** Disponível em: <<https://www.arduino.cc/>>. Acesso em: 1 jul.

2019.

BARRET, S. **Arduino Microcontroller Processing for Everyone!** 3 ed. ed. [s.l:

s.n.].

BOLANAKIS, D. E. **Microcontroller Education.** [s.l: s.n.].

BRAGA, N. C. Como funciona o Reed-Switches (MEC098). 2015.

CABALLERO, D. C. **How to use the Arduino Ethernet Shield – Scidle.** Disponível

em: <<https://scidle.com/how-to-use-the-arduino-ethernet-shield/>>. Acesso em: 10 jun. 2019.

DEVELOPERS. **Android NDK.** Disponível em: <<https://developer.android.com/ndk>>.

Acesso em: 1 jul. 2019.

DU BUF, H.; BAYER, M. M. Introduction To Adiac and This Book. p. 1–8, 2010.

ELETRÔNICA, R. S. **Segurança: quanto custa para o Brasil e para o brasileiro?** | **Revista Segurança Eletrônica**. Disponível em:

<<http://revistasegurancaeletronica.com.br/seguranca-quanto-custa-para-o-brasil-e-para-o-brasileiro/>>. Acesso em: 23 fev. 2019.

FARR, C. **Firestore's scalable backend makes it “10 times easier” to build apps** | **VentureBeat**. Disponível em: <<https://venturebeat.com/2013/02/13/firebases-backend-makes-it-ten-times-easier-to-build-apps/>>. Acesso em: 10 jun. 2019.

FELIPEFLOP. **Sensor de Movimento Presença PIR - FilipeFlop**. Disponível em: <<https://www.filipeflop.com/produto/sensor-de-movimento-presenca-pir/>>. Acesso em: 22 maio. 2019.

FILHO, R. DE G. N. **Introdução ao Computador**. Disponível em: <<http://www.di.ufpb.br/raimundo/ArqDI/Arq2.htm>>. Acesso em: 13 maio. 2019.
GALLUP. 2018 Global Law and Order. 2018.

GEARBEST BRASIL. **KEYES KT0012 37 em 1 Kit Sensor** | **Gearbest Brasil**. Disponível em: <https://br.gearbest.com/kits/pp_226897.html>. Acesso em: 20 maio. 2019.

GFK BRAZIL. **Tecnologias smart home são tão relevantes quanto carro conectado, na visão do brasileiro**. Disponível em: <<https://www.gfk.com/pt-br/insights/press-release/tecnologias-smart-home-sao- tao-relevantes-quanto-carro-conectado-na-visao-do-brasileiro/>>. Acesso em: 2 maio. 2019.

GOOGLE. **Firestore**. Disponível em: <<https://firebase.google.com/>>. Acesso em: 25 fev. 2019a.

GOOGLE. **Firestore**. Disponível em: <<https://firebase.google.com/>>. Acesso em: 10 jun. 2019b.

IBGE. IBGE | Agência de Notícias | Insegurança aumenta, restringe direitos e ameaça liberdade no país. Disponível em:

<<https://agenciadenoticias.ibge.gov.br/agencia-noticias/2012-agencia-de-noticias/noticias/21586-inseguranca-aumenta-restringe-direitos-e-ameaca-liberdade-no-pais>>. Acesso em: 23 fev. 2019.

JAVA. java.com: JAVA + Você. Disponível em: <https://www.java.com/pt_BR/>.

Acesso em: 1 jul. 2019.

KOTLIN. Kotlin for Android - Kotlin Programming Language. Disponível em:

<<https://kotlinlang.org/docs/reference/android-overview.html>>. Acesso em: 1 jul. 2019.

MA, F. The Firebase Blog: What's new at Firebase Summit 2018. Disponível em:

<<https://firebase.googleblog.com/2018/10/whats-new-at-firebase-summit-2018.html>>. Acesso em: 10 jun. 2019.

MANDULA, K. et al. Mobile based home automation using Internet of Things(IoT).

2015 International Conference on Control Instrumentation Communication and Computational Technologies, ICCICCT 2015, p. 340–343, 2016.

MCROBERTS, M.; JOICE, T.; COSTA, E. Arduino Básico. In: NOVATEC (Ed.). .

Arduino Básico. 2 nd. ed. [s.l: s.n.].

NASIR, S. Z. Introduction to ATmega328 - The Engineering Projects. Disponível

em: <<https://www.theengineeringprojects.com/2017/08/introduction-to-atmega328.html>>. Acesso em: 29 maio. 2019.

SANTOS, B. P. et al. Internet das Coisas: da Teoria à Prática. Departamento de

Ciência da Computação Universidade Federal de Minas Gerais (UFMG) Belo Horizonte, MG, Brasil, v. Cap. 1, p. 1–50, 2016.

STANLEY, M.; LEE, J. Sensor Analysis for the Internet of Things. [s.l: s.n.]. v. 9

THOMSEN, A. **Tutorial Sensor de Chama com Arduino - FilipeFlop**. Disponível em: <<https://www.filipeflop.com/blog/sensor-de-chama-com-arduino/>>. Acesso em: 21 maio. 2019.

VAQUERO, L. M. et al. A Break in the Clouds : Towards a Cloud Definition. v. 39, n. 1, p. 50–55, 2009.

WIKIPÉDIA. **Wikipédia, a enciclopédia livre**. Disponível em: <https://pt.wikipedia.org/wiki/Wikipédia:Página_principal>. Acesso em: 24 jun. 2019.

WIZNET. W5100 Datasheet. p. 71, 2008.

APÊNDICE

```
//MONOGRAFIA LUCAS - TODOS OS SENSORES
```

```
#include <Ethernet.h>
```

```
//DECLARAÇÕES DE FUNÇÕES
```

```
void Fogo();
```

```
void Presenca();
```

```
void Porta ();
```

```
//SENSOR DE CHAMA
```

```
int pino_A0 = A1;
```

```
int estadoChama = 0;
```

```
int valor_a = 0;
```

```
int valor_d = 0;
```

```
unsigned long startMillis1;
```

```
unsigned long currentMillis1;
```

```
const unsigned long periodo1 = 50;
```

```
//SENSOR MAGNÉTICO
```

```
const int SMa = 3;
```

```
const int LEDa = 8;
```

```
int estadoPA = 0;
```

```
const int SMb = 6;
```

```
const int LEDb = 9;
```

```
int estadoPB = 0;
```

```
//SENSOR DE PRESENÇA
```

```
const int pinorele = 7; // pino de comando do modulo relé de estado sólido
```

```
const int pinopir = 5; // pino de leitura digital do sensor de presença
```

```
int estadoMov = 0;
```

```

unsigned long startMillis1;
unsigned long currentMillis1;

//Ethernet
// Enter a MAC address for your controller below.
// Newer Ethernet shields have a MAC address printed on a sticker on the shield
byte mac[] = { 0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED };

// if you don't want to use DNS (and reduce your sketch size)
// use the numeric IP instead of the name for the server:
//IPAddress server(74,125,232,128); // numeric IP for Google (no DNS)
char server[] = "www.bragaburgerq.com.br"; // name address for Google (using
DNS)

// Set the static IP address to use if the DHCP fails to assign
IPAddress ip(192, 168, 0, 177);
IPAddress myDns(192, 168, 0, 1);

// Initialize the Ethernet client library
// with the IP address and port of the server
// that you want to connect to (port 80 is default for HTTP):
EthernetClient client;

void setup()
{

//SENSOR DE CHAMA
Serial.begin(9600);
pinMode(pino_A0, INPUT);
startMillis1 = millis();

//SENSOR MAGNÉTICO
pinMode(SMa, INPUT);

```



```

pinMode(SMb, INPUT);
pinMode(LEDa, OUTPUT);
pinMode(LEDb, OUTPUT);

//SENSOR DE PRESENÇA
// define o pino que aciona o relé como saída
pinMode(pinorele, OUTPUT);
// define os pinos dos sensores como entrada
pinMode(pinopir, INPUT); //Define pino sensor de presença como entrada

//-----
// start the Ethernet connection:
Serial.println("Inicializa Ethernet com DHCP:");
if (Ethernet.begin(mac) == 0) {
  Serial.println("Falha ao conectar Ethernet usando DHCP");
  // Check for Ethernet hardware present
  if (Ethernet.hardwareStatus() == EthernetNoHardware) {
    Serial.println("Ethernet shield não encontrado. :(");
    while (true) {
      delay(1); // do nothing, no point running without Ethernet hardware
    }
  }
  if (Ethernet.linkStatus() == LinkOFF) {
    Serial.println("Cabo Ethernet não encontrado.");
  }
  // try to configure using IP address instead of DHCP:
  Ethernet.begin(mac, ip, myDns);
} else {
  Serial.print(" DHCP IP recebido");
  Serial.println(Ethernet.localIP());
}
// give the Ethernet shield a second to initialize:
delay(1000);
Serial.print("Conectando no servidor: ");

```

```
Serial.print(server);
Serial.println("...");
//-----
}

void loop()
{
  Fogo();

  //SENSOR MAGNÉTICO

  if (!digitalRead(SMa)) {
    digitalWrite(LEDa, HIGH);

    if (estadoPA == 0) {
      enviaFirebase(2, 2, 1);
      estadoPA = 1;
      Serial.println("Porta A Aberta");
    }

  } else {
    digitalWrite(LEDa, LOW);

    if (estadoPA == 1) {
      enviaFirebase(2, 2, 0);
      estadoPA = 0;
      Serial.println("Porta A Fechada");
    }
  }

  if (!digitalRead(SMb)) {
    digitalWrite(LEDb, HIGH);

    if (estadoPB == 0) {
```

```
    enviaFirebase(3, 2, 1);
    estadoPB = 1;
    Serial.println("Porta B Aberta");
}

} else {
    digitalWrite(LEDb, LOW);

    if (estadoPB == 1) {
        enviaFirebase(3, 2, 0);
        estadoPB = 0;
        Serial.println("Porta B Fechada");
    }
}

//SENSOR DE PRESENÇA
Presenca();
delay(10);

}

void Fogo()
{
    //SENSOR DE CHAMA
    currentMillis1 = millis();
    if ((currentMillis1 - startMillis1) >= periodo1)
    {
        int valor_a = analogRead(pino_A0);

        Serial.println("Porta analogica: ");
        Serial.print(valor_a);

        if (valor_a < 200) {
```

```

    if (estadoChama == 0) {
        enviaFirebase(1, 1, 1);
        estadoChama = 1;
        Serial.println("Fogo detectado !!!");
    }
} else {
    if (estadoChama == 1) {
        enviaFirebase(1, 1, 0);
        estadoChama = 0;
        Serial.println("SEM FOGO !!!");
    }
}
startMillis1 = currentMillis1;
}
}

void Presenca()
{
    // lê os estados do sensores e os armazena nas variáveis luz e movimento
    int movimento = digitalRead(pinopir);

    if (movimento == HIGH) // Se não houver luz E (&&) houver movimento aciona o
    relé
    {
        digitalWrite(pinorele, LOW); // aciona relé (este relé é acionado em nivel LOW)
        // Serial.println("Presença!!");
        // enviaFirebase(4,3, 1);
        if (estadoMov == 0) {
            enviaFirebase(4,3, 1);
            estadoMov = 1;
            Serial.println("Movimento na porta do LABGEA");
        }
    }
}
}

```

```

else // Caso contrário, desliga o relé
{
    digitalWrite(pinorele, HIGH);// desliga o relé
    if (estadoMov == 1) {
        enviaFirebase(4,3, 0);
        estadoMov = 0;
        Serial.println("Sem movimento");
    }
    // enviaFirebase(4,3, 0);

}

}

// Funcao que envia os dados para a nuvem FireBase
void enviaFirebase(int disp, int tipo, int estado) {

    char url[150];

    sprintf(url, "GET
/arduino/firebasepost.php?PROJETO=monitoramento&disp=%d&estado=%d&tipo=
%d HTTP/1.1", disp, estado, tipo);
    Serial.println(url);
    if (client.connect(server, 80)) {
        Serial.print("conectado em ");
        Serial.println(client.remoteIP());
        // Make a HTTP request:
        // Enviando as variáveis para a NUVEM
        // PROJETO sinais ou seguranca
        // PALTA=120 medida da pressão alta 120
        client.println(url);
        client.println("Host: www.bragaburgerq.com.br");
        client.println("Connection: close");
        client.println();
    }
}

```

```
    client.stop();
  } else {
    // if you didn't get a connection to the server:
    Serial.println("connection failed");
  }

}
/*
int movimento = digitalRead(pinopir);

if (movimento == HIGH) // Se não houver luz E (&&) houver movimento aciona o
relé
{
  digitalWrite(pinorele, LOW); // aciona relé (este relé é acionado em nível LOW)
}
```