UNIVERSIDADE FEDERAL DO ESPÍRITO SANTO
CENTRO TECNOLÓGICO
DEPARTAMENTO DE ENGENHARIA ELÉTRICA
PROJETO DE GRADUAÇÃO

ÉMILIE SOUZA DE ALENCAR

SISTEMA DE AUTOMAÇÃO RESIDENCIAL PARA SEGURANÇA E MONITORAMENTO BASEADO NOS CONCEITOS DA INTERNET DAS COISAS COM GERENCIAMENTO E ACIONAMENTO REMOTO POR MEIO DE APLICAÇÃO WEB

VITÓRIA – ES
JULHO/2018

ÉMILIE SOUZA DE ALENCAR

SISTEMA DE AUTOMAÇÃO RESIDENCIAL PARA SEGURANÇA E MONITORAMENTO BASEADO NOS CONCEITOS DA INTERNET DAS COISAS COM GERENCIAMENTO E ACIONAMENTO REMOTO POR MEIO DE APLICAÇÃO WEB

Parte manuscrita do Projeto de Graduação da aluna **Émilie Souza de Alencar**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheira Eletricista.

Orientador: Prof. Dr. Anselmo Frizera Neto

VITÓRIA – ES
JULHO/2018

ÉMILIE SOUZA DE ALENCAR

SISTEMA DE AUTOMAÇÃO RESIDENCIAL PARA SEGURANÇA E MONITORAMENTO BASEADO NOS CONCEITOS DA INTERNET DAS COISAS COM GERENCIAMENTO E ACIONAMENTO REMOTO POR MEIO DE APLICAÇÃO WEB

Parte manuscrita do Projeto de Graduação da aluna **Émilie Souza de Alencar**, apresentado ao Departamento de Engenharia Elétrica do Centro Tecnológico da Universidade Federal do Espírito Santo, como requisito parcial para obtenção do grau de Engenheira Eletricista.

COMISSÃO EXAMINADORA:

_____

Prof. Dr. Anselmo Frizera Neto
Universidade Federal do Espírito Santo - Ufes
Orientador

_____

Prof. Dr. Klaus Fabian Côco
Universidade Federal do Espírito Santo - Ufes
Examinador

_____

Prof. Dr. Edson de Paula Ferreira
Universidade Federal do Espírito Santo - Ufes
Examinador

RESUMO

Automação residencial é uma área em expansão no mercado e a aplicação de suas tecnologias na vida diária acrescenta segurança, conforto e economia para os usuários. Avanços na inovação de redes, sensores, atuadores e sistemas embarcados atrelados à Internet tornaram possível o desenvolvimento de novas soluções que dão suporte e assistência ao atendimentos das necessidades da sociedade contemporânea de maneira mais eficiente e menos custosa. A tecnologia pode ser uma grande aliada no auxílio às pessoas a se sentirem mais seguras e em controle sobre o que acontece em seus lares, mesmo quando deixam suas casas para realizar suas atividades diárias ou quando estão ausentes por longos períodos. Sendo assim, desenvolveu-se um sistema de segurança e monitoramento residencial o qual apresenta conhecimentos sobre automação, baseado nos conceitos da Internet das Coisas (IoT). Este trabalho apresenta o desenvolvimento e implementação de um sistema que possui uma plataforma de interface Web para visualização e controle local ou remoto das informações do ambiente doméstico, permite acionar e desativar alarme, monitoramento e captura de imagens no ambiente residencial. O conjunto deve reagir de forma ágil, alertando ao usuário através de sinais sonoros e SMS (S*hort Message Service*) sobre possíveis invasores em sua residência no período que o alarme estiver acionado, permitindo a antecipação do acionamento da ação das autoridades competentes, assegurando maior segurança e facilidade ao usuário.

LISTA DE FIGURAS

Figura 1-Comparação de unidades vendidas para equipamentos de casas inteligentes em 2015 com tendências para 2020. 14
Figura 2 –Relação entre o número de dispositivos e pessoas conectadas à Internet. 16
Figura 3 – Pirâmide de construção da sabedoria a partir de dados. 17
Figura 4 - Representação visual do projeto. 19
Figura 5 - Arquitetura conceitual do sistema de segurança residencial. 21
Figura 6-Modelo de organização de camadas da arquitetura de projetos IoT proposta por Wu et al. 22
Figura 7-Raspberry Pi 2 e os componentes para ajuste de seu funcionamento 24
Figura 8-Apresentação do módulo de câmera conectada ao Raspberry Pi 26
Figura 9-Imagens do sensor piroelétrico PIR vista de várias posições de forma a expor seus componentes. 27
Figura 10- Diagrama de bloco de funcionamento do sensor PIR. 28
Figura 11 - Buzzer ativo utilizado como alarme sonoro. 29
Figura 12-Visão geral do funcionamento do Software no lado do cliente. 30
Figura 13-Arquitetura do sistema no lado do servidor. 32
Figura 14-Diagrama representativo do funcionamento das requisições feitas à API Twilio. 34
Figura 15-Telas que compõem a Interface da aplicação. 36
Figura 16 - Fluxo de funcionamento da Interface do usuário. 37
Figura 17-Tela de alarme sonoro e seus três estados 38
Figura 18-Esquemático desenvolvido para desenvolvimento do circuito do protótipo 39
Figura 19-Lógica de controle com alarme ativo 40
Figura 20-Imagem da aplicação web sendo utilizada para monitoramento através do Macbook e iPad. 44
Figura 21- Imagem da tela do celular acessando a tela de monitoramento da aplicação. 44
Figura 22-Procedimento de teste para determinação do tempo médio de latência do streaming de vídeo. 46
Figura 23-Verificação sobre o funcionamento de notificação por mensagem SMS. 48

LISTA DE ABREVIATURAS E SIGLAS

API	*Application Programming Interface*
AURESIDE	Associação Brasileira de Automação Residencial
CES	*Consumer Electronics Show*
CSI	*Camera Serial Interface*
CSS	*Cascade Style Sheet*
D2D	*Device to Device*
DOM	*Document Object Model*
GPIO	*General Purpose Input/Output*
HTML	*Hypertext Markup Language*
HTTP	*Hypertext Transfer Protocol*
IBSG	*Cisco Internet Business Solutions Group*
IEEE	Instituto de Engenheiros Eletricistas e Eletrônicos
IoT	*Internet of Things* (Internet das Coisas)
ITU	*International Telecommunication Union*
M2M	*Machine to Machine*
MIT	*Massachusetts Institute of Technology*
NFC	*Near Field Communication*
PCB	*Printed Circuit Board*
PHP	*Hypertext Preprocessor*
PIR	*Passive Infrared Sensor*
PLC	*Power Line Carrier*
REST	*Representational State Transfer*
RFID	*Radio-Frequency Identification*
SBC	*Single Board Computer*
SMS	*Short Message Service*
SO	Sistema Operacional
SPA	*Single Page Application*
UI	*User Interface*

SUMÁRIO

1 INTRODUÇÃO 7
1.1 Contexto 9
1.2 Objetivos 11
1.2.1 Objetivo Geral 11
1.2.2 Objetivos Específicos 11
1.3 Estrutura da Dissertação 12
2 FUNDAMENTAÇÃO TEÓRICA 13
2.1 Automação Residencial 13
2.2 Internet das Coisas (IoT) 15
3 SISTEMA DE SEGURANÇA E MONITORAMENTO 19
3.1 Visão geral do Sistema 19
3.2 Concepção Geral Sobre o Funcionamento 20
4 DESENVOLVIMENTO E IMPLEMENTAÇÃO 22
4.1 Materiais 23
4.1.1 Raspberry Pi 24
4.1.2 Câmera Raspberry Pi 25
4.1.3 Sensor de Presença Piroelétrico (PIR) 27
4.1.4 Alarme Sonoro (Buzzer) 29
4.2 Desenvolvimento do Software 29
4.2.1 Client-Side 29
4.2.2 Server-Side 31
5 IMPLEMENTAÇÃO 35
5.1 Interface do Usuário (UI) 35
5.2 Protótipo 39
5.3 Implementação da lógica do sistema 40
6 ANÁLISES E RESULTADOS 42
7 CONCLUSÕES 49
8 REFERÊNCIAS BIBLIOGRÁFICAS 51

1 INTRODUÇÃO

A automação residencial ou domótica, como também é chamada, tem sido tópico de interesse desde 1970, quando, por meio da rede elétrica residencial, módulos recebiam comandos baseados no conceito de PLC (*Power Line Communication*) para realização das ações do sistema automático (MURATORI e BÓ, 2005). As expectativas sobre as possibilidades desse modelo de tecnologia concomitante à popularização de produtos eletrônicos e da *Internet* no cotidiano das pessoas, reformulou todo o conceito e a forma com que a automação pode ser empregada nas residências (JOSE, MALEKIAN, *et al.*, 2016). A compreensão da automação residencial tem se unido à ideia de ambientes inteligentes, a qual surge a partir da possibilidade de dispositivos comunicarem entre si, compartilhando informações que permitirão um entendimento e gerenciamento mais eficiente dos sistemas. Com isso, a automação possibilita através da tecnologia, substituir o trabalho humano em comandar a execução de ações rotineiras ou mesmo que são possíveis de serem automatizadas, gerando praticidade à vida dos usuários, entre outros benefícios (WU, LU, *et al.*, 2010).

O crescimento da indústria da *Internet* aumentou os esforços para explorar esta tecnologia na inserção da inteligência a processos e sistemas (ALLAMEH, JOZAM, *et al.*, 2012). Essa ascensão no uso da *Internet* para desenvolvimento de novos sistemas, propiciou a implementação de técnicas de automação que permitem o desenvolvimento de serviços com menos componentes de *hardware*, minimizando os custo e possibilitando a atribuição de mais funcionalidades aos processos, constituindo a *Internet* como uma ferramenta de extrema relevância na elaboração e integração de projetos e com aplicações mais abrangentes e econômicas (HOU, WU, *et al.*, 2008).

A *Internet* das Coisas (IoT), que vem do termo *Internet of Things*, é o nome que se dá ao conceito que propõe utilizar a *Internet* na construção da comunicação M2M (*Machine to Machine*). O compartilhamento proposto pela IoT baseia-se na implementação de tecnologias de computação que permitem que informações e dados sejam coletados e transmitidos através da *web*, com o propósito de concluir tarefas e ser uma base de informações para construção de lógicas e comportamentos mais avançados (ISIKDAG, 2015). Este modelo inspirou entusiastas da domótica a criarem ferramentas, sistemas e produtos inteligentes que são capazes de interagirem entre si através da *Internet*. Gerando um novo padrão de criação e

desenvolvimento de dispositivos e uma revolução nos conceitos sobre o uso da *Internet* (LISBOA e CRUZ, 2014).

O aprimoramento e a expansão da tecnologia tem criado novos horizontes de possibilidades de implementação de tecnologias na vida diária das pessoas. (BRUCKNER, 2013). Como consequência há uma incessante reestruturação nas definições sobre as necessidades básicas da sociedade. Nos dias atuais, considera-se como preceito indispensável a segurança, o conforto e a praticidade.

Uma estatística preocupante é o índice de casos de assaltos a residências (HOU, WU, *et al.*, 2008). O que frequentemente delonga a ação das autoridades quando há ocorrências de furtos e roubos em residências, é a falta ou demora na percepção de que o crime está em andamento. A existência de um sistema de monitoramento e segurança possibilita alertar o morador sobre o ocorrido, permitindo a antecipação no acionamento da ação policial e facilitando a identificação e detenção de criminosos que outrora escapariam impunes levando os bens das vítimas (FOLHA DIÁRIA, 2017).

Um dos pilares que sustenta a cidadania são os serviços básicos oferecidos e apoiados pelo Estado, como os serviços de emergência que são prestados aos cidadãos em momentos de crise, que são direitos atestados pela Constituição Federal brasileira (CARDOSO, 2015). No entanto, muitos casos de invasão residencial acontecem quando os moradores não estão presentes por curtos e às vezes grandes períodos de tempo. A ação criminosa muitas vezes passa despercebida pelos moradores, vizinhos, porteiros ou vigilantes sendo descoberta somente após o ocorrido.

Com o aumento na incidência de roubos e furtos em residências depreende-se ser primordial que medidas sejam tomadas para diminuir indicadores de criminalidade em residências. A tecnologia nesse sentido se torna uma grande aliada no combate preventivo, sendo um facilitador para que os serviços de atendimento a emergências sejam acionados, dando assistência e suporte aos moradores em momentos críticos. Considera-se que um sistema de automação residencial moderno deve prover suporte ao usuário na evidência de qualquer intrusão ou tentativa de intrusão, notificando e alarmando o proprietário, além de preservar

evidências da invasão para que o criminoso possa ser conduzido à justiça (JOSE, MALEKIAN, *et al.*, 2016).

Levando em consideração a relevância dos conceitos apresentados e a necessidade de uma forma de prevenção a crimes contra o patrimônio residencial, o desenvolvimento deste projeto propõe a construção de um sistema de automação residencial, que permita por meio de dados compartilhados via *web*, gerenciar ações e dispositivos além de monitorar um sistema de segurança residencial local e remotamente. As ações de interação e visualização do ambiente por parte dos usuários serão realizadas através de uma aplicação *web*. A interface de acesso do usuário deve ser responsiva para que assim esta possa ser acessada a partir de qualquer tipo de dispositivo, possibilitando ao usuário o gerenciamento e monitoramento de sua residência de qualquer lugar desde que esteja conectado à *Internet*, contribuindo para a segurança e comodidade dentro e fora de suas residências.

1.1 Contexto

O número de crimes contra o patrimônio, que incluem roubos e furtos em residências, tem crescido em várias regiões do Brasil (SANTOS, 2014). Segundo o Instituto de Segurança Pública (ISP) houve intensificação no número de casos em 70% no RJ, a comparação foi feita entre os meses de janeiro e fevereiro de 2017 com os de 2018 e não inclui as ocorrências de furtos (quando não há presença de vítimas no local do crime) (HOU, WU, *et al.*, 2008). No segundo semestre de 2017 foi registrado em São Paulo que a cada uma hora uma casa é invadida e em 3 anos este é o maior número de roubo a residências já registrado (CARVALHO e TOMAZELA, 2017). Em Brasília o número de moradias violadas foi de em média três por dia no ano de 2017, e esse ano foi apontada como sendo a modalidade de crime que mais aumentou em comparação com o mesmo período no ano anterior (FOLHA DIÁRIA, 2017).

Em épocas especiais e/ou comemorativas a situação se agrava ainda mais. Durante feriados prolongados, como o de carnaval, as ocorrências deste tipo de delito podem aumentar em até 15%. Do mesmo modo, as casas de veraneio sofrem com um acréscimo na ocorrência de furtos em cerca de 10% durante o verão (DINO, 2018). O índice de roubos e furtos no Estado do Espírito Santo aumentam expressivamente no período de férias, quando o índice de turistas

visitando as cidades litorâneas aumenta, o que oferece grande ameaça não só aos capixabas mas também aos visitantes (MELO, 2017).

Os criminosos não têm encontrado dificuldades para entrar nas residências (CARVALHO e TOMAZELA, 2017). Ações criminosas dessa natureza são facilitadas, pois, diferentemente de um furto em que o ladrão depende da rapidez para cometer o crime, esta possibilita livre acesso aos bens das vítimas por um período longo de tempo por existir um risco menor de serem descobertos. Alguns salteadores rondam bairros em busca de moradores saindo de seus lares para os abordar e invadir suas residências (ANDRADE, 2017). Além disso, os métodos para realizar os furtos estão se sofisticando, empregando meios como a mídia social para identificar a ausência de moradores nas casas e apartamentos, o que tem sido cada vez mais comum devido à grande exposição das pessoas nas redes sociais (CARVALHO e TOMAZELA, 2017).

A preocupação com a segurança é algo recorrente por parte dos cidadãos de bem e do poder público. Antônio Carlos Vieira, titular da Sejusp, afirma que roubo a residências deve ganhar atenção especial depois de constatado um aumento de 9% nos crimes dessa natureza em Mato Grosso do Sul de 2016 para 2017. Apesar de o Mato Grosso do Sul ser o quinto Estado mais seguro no Brasil, o governo já investiu R$ 90 milhões nos últimos três anos e vai investir mais R$ 30 milhões neste ano no programa MS Mais Seguro, diz a governadora em exercício (SANTOS e KASPARY, 2018).

As empresas especializadas em segurança privada já sentem na prática o que os números expõem (CARVALHO e TOMAZELA, 2017). Cada vez mais, moradores têm investido em proteção, entretanto, a melhor forma de combater este problema e não fazer parte dessas estatísticas é através da prevenção (DINO, 2018). Anualmente são investidos muitos recursos no desenvolvimento de novas tecnologias para prover mais segurança às pessoas. O incentivo ao avanço tecnológico, assim como a expansão dos mercados mundiais auxilia a preparação e a composição de um sistema de segurança que ofereça benefícios, como, notificar a vítima rapidamente caso haja violação da propriedade, podendo agilizar o acionamento e consequente intervenção da polícia, oferecendo mecanismo adicional de proteção ao patrimônio residencial (FOLHA DIÁRIA, 2017).

1.2 Objetivos

Esta seção apresenta o objetivo geral e os objetivos específicos deste trabalho.

1.2.1 Objetivo Geral

O objetivo geral deste trabalho é o de assegurar ao usuário acesso local e remoto a um sistema de automação residencial integrado à uma interface *Web*, a qual permite o acionamento de alarmes, envio de notificação de intrusos através de SMS e alarme sonoro e a visualização de *streaming* de vídeo do ambiente na residência em que o protótipo está inserido. Este sistema deve agir de forma ágil, notificando o usuário sobre anormalidades caso o alarme esteja acionado, promovendo uma diminuição no tempo entre o acontecimento de uma invasão e o conhecimento do usuário sobre o ocorrido, assim como permitir o constante monitoramento de sua residência.

1.2.2 Objetivos Específicos

Os objetivos específicos que baseiam a realização deste projeto são:

- construir protótipo de automação residencial para controlar os sensores e atuadores com ações autônomas predefinidas e configuradas;
- desenvolver interface *web* responsiva de gerenciamento do protótipo e monitoramento da residência;
- permitir captura de imagens a partir do sistema de monitoramento;
- notificar via SMS;
- desenvolver servidor *web* que permita gerenciar a interface e as ações de controle;
- disponibilizar *streaming* de vídeo disponível 24h ao usuário;
- acessar e controlar do sistema localmente e remotamente.

1.3 Estrutura da Dissertação

O trabalho está organizado em sete capítulos. O Capítulo 1 apresenta uma introdução geral, seguida por uma contextualização do motivo da pesquisa e objetivos do projeto. O Capítulo 2 é destinado à exposição do embasamento teórico necessário para fundamentação do projeto. Por sua vez, no Capítulo 3, realiza-se uma descrição da visão geral do sistema e suas funcionalidades. O Capítulo 4 seguinte informa com maior detalhamento o desenvolvimento do projeto e as partes constituintes do mesmo. O Capítulo 5 apresenta como a interface, o protótipo e a lógica do sistema foram implementados. No sexto, são abordadas análises e resultados obtidos. O Capítulo 7, apresenta as conclusões e considerações finais acerca do projeto. Por fim, o último capítulo apresenta as referências bibliográficas utilizadas neste trabalho.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 Automação Residencial

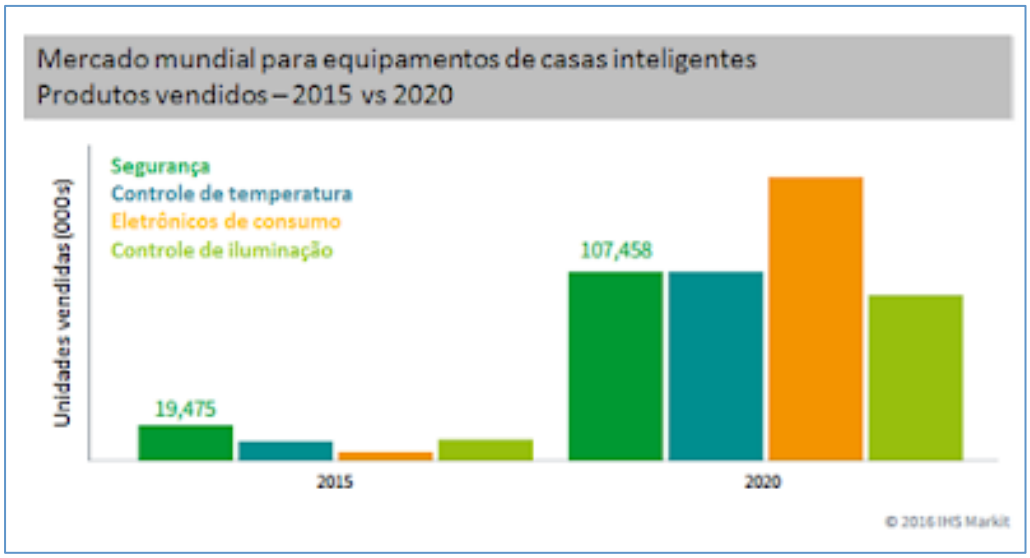
A Automação é um conceito que pode ser definido como a habilidade de sistemas realizarem atividades com pouca ou nenhuma interferência humana, seguindo instruções predeterminadas e programadas. Sistemas automáticos são desenvolvidos quando deseja-se executar atividades normalmente repetitivas ou em resposta a eventos específicos, substituindo o trabalho e a lógica humana pela tecnologia (PEREIRA, 2008). Com isso, a concepção da automação residencial referencia-se à utilização de processos automatizados em casas, apartamentos ou qualquer ambiente residencial no qual pretende-se que dispositivos realizem tarefas de maneira autônoma em benefício da segurança, conforto, comodidade e economia dos usuários. Neste contexto, os dispositivos e equipamentos são capazes de comunicar-se e seguir instruções programadas seguindo o desejo do morador da residência (WORTMEYER, FREITAS e CARDOSO, 2005).

Historicamente, a automação foi primeiramente desenvolvida no âmbito industrial e comercial. A automação residencial surge posteriormente, devido a este segmento não propiciar retorno imediato a investimentos, como seus similares. Primeiramente surgiu a automação ligada ao controle e automação industrial seguida pela automatização de áreas patrimoniais e institucionais de edifícios comerciais. A aceitação de tecnologias voltadas para a automação residencial começou a ser instigada a partir da disseminação e anuência do uso de dispositivos móveis e da Internet no cotidiano da população em geral, como consequência, a produção e o desenvolvimento de novas tecnologias de automação residencial passaram a ser mais requeridas (MURATORI e BÓ, 2005).

Segundo à Associação Brasileira de Automação Residencial (AURESIDE), o mercado de automação residencial possui fortes tendências de desenvolvimento e expansão. Em nível mundial, as pesquisas apontam para um crescimento de 11,36% entre 2014 e 2020, o qual considera-se ter sido estimulado devido a cinco fatores principais: o aumento no uso de dispositivos móveis, nova forma de mercado de construção civil, desejo de consumidores de terem mais praticidade no dia a dia, busca pela eficiência no sentido de preservar o meio ambiente e a melhoria das tecnologias sem fio. O gráfico a seguir apresenta dados do

Financial Times comparando o mercado em 2015 com as tendências para 2020 (AURESIDE, 2015).

Figura 1-Comparação de unidades vendidas para equipamentos de casas inteligentes em 2015 com tendências para 2020.



Fonte: AURESIDE, (2015).

Conclui-se a partir da Figura 1 que a relação do número de equipamentos vendidos para casas inteligentes em 2020 será imensamente maior do que em 2015, em especial os equipamentos voltados à segurança, serão vendidos 5 vezes mais unidades em 2020. Esses dados são um reflexo dos avanços tecnológicos voltados a automação residencial atendendo os novos apelos da sociedade.

De forma a atender os desejos do indivíduo que a utiliza, uma casa inteligente possui 3 partes essenciais: *design*, tecnologia e o ambiente. A associação dessas partes originam diversas habilidades, como capacidade de percepção, cognição, análise, raciocínio e antecipação tornando o sistema apto a tomar as ações devidas (ALLAMEH, JOZAM, *et al.*, 2012). A aplicação dessas partes permite a criação de sistemas de controle de energia, temperatura, segurança, acesso, iluminação, aparelhos domésticos, recursos áudio visuais, aparelhos de assistência médica dentre outros (BRUCKNER, 2013).

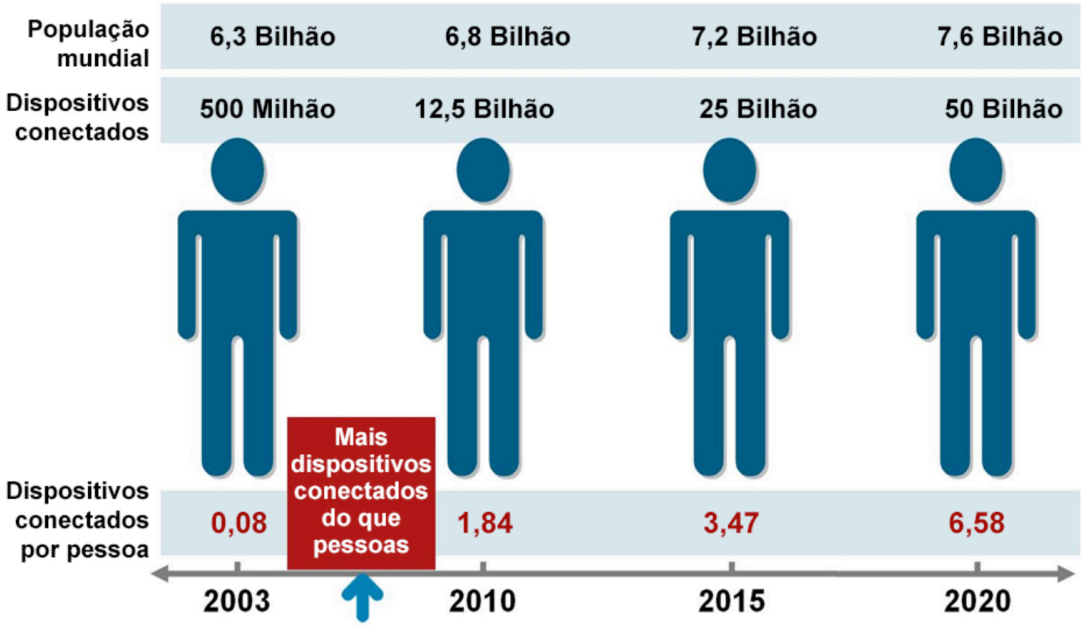
2.2 Internet das Coisas (IoT)

A IoT é um conceito e sua aplicação varia de acordo com o desejo, necessidade, recurso financeiro e forma de utilização (MINERVA, BIRU e ROTONDI, 2015). O interesse dessa ideia de tecnologia não está no dispositivo, mas na informação contida nele, transmitindo dados em tempo real sobre o ambiente no qual está inserido. Dessa forma, esses objetos permitem unir o mundo digital ao mundo real, impactando na vida profissional, pessoal e nos relacionamentos sociais da sociedade (EVANS, 2011).

De acordo com o ITU (*International Telecommunication Union*), o surgimento da IoT é considerado uma revolução tecnológica e representa o futuro da computação e comunicação, onde o termo "sociedade de rede onipresente" traduz a tendência dessa nova tecnologia (ITU, 2005). Assim sendo, o seu desenvolvimento permite automaticamente estabelecer a implantação de outros conceitos proveitosos, como os de ambientes inteligentes e computação pervasiva, termo utilizado para descrever a onipresença da informática na vida diária das pessoas (CARVALHO e TOMAZELA, 2017). Os avanços da *Internet* tem apontado rumo ao desenvolvimento e otimização, todavia, ainda realiza fundamentalmente o mesmo papel ao qual foi projetada. O IoT neste sentido, representa a primeira evolução autêntica da *Internet* (EVANS, 2011).

Segundo o *Cisco Internet Business Solutions Group* (IBSG) o exato momento do nascimento do IoT foi o momento em que se existiam mais 'coisas' ou objetos conectados à *Internet* do que pessoas (EVANS, 2011).

Figura 2 –Relação entre o número de dispositivos e pessoas conectadas à *Internet*.



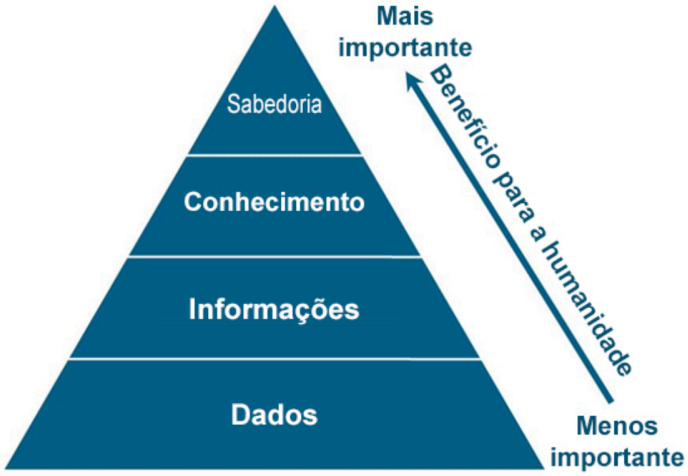
Fonte: EVANS, (2011).

Segundo sugere a Figura 2, estima-se que o surgimento do IoT ocorreu entre os anos de 2003 e 2010. Segundo a Cisco IBSG este fato ocorreu mais precisamente entre 2008 e 2009 (EVANS, 2011).

A abstração de conectar dispositivos vem desde 1991 quando Bill Joy, co-fundador da Sun Mycrosystems, elaborou a ideia da conexão D2D (*Device to Device*). As raízes do IoT são rastreadas até o MIT (*Massachusetts Institute of Technology*), onde em 1999 Kevin Ashton concebeu o termo 'Internet das Coisas' enquanto trabalhava no campo de identificação de frequência de rádio em rede (*Radio-Frequency Identification*) e tecnologias de sensores. Aproximadamente 10 anos depois da concepção deste conceito, Ashton escreveu o artigo "A coisa da Internet das Coisas" para o RFID Journal (OLIVEIRA e GIGLIO, 2017).

A importância do princípio de compartilhamento de informações do IoT pode ser melhor compreendida através da Figura 3.

Figura 3 – Pirâmide de construção da sabedoria a partir de dados.



Fonte: EVANS, (2011).

A pirâmide acima propõe a forma que os humanos processam os dados. Os dados são a matéria-prima para a constituição da informação sobre o sistema, esta por conseguinte propicia o desenvolvimento de conhecimento. A sabedoria emerge da soma do conhecimento com a experiência. Logo, os dados individuais sozinhos podem não ser tão úteis, mas em grande quantidade podem identificar tendências e padrões e por fim possibilitar a formação da sabedoria, evidenciando o valor do princípio de compartilhamento proposto pelo IoT (EVANS, 2011).

Conectividade inteligente e computação pervasiva podem ser alcançadas no IoT a partir de 3 bases fundamentais. A primeira é o compartilhamento das informações do usuário e dos dispositivos. Para transmitir a informação para o contexto no qual esta é importante, arquitetura de *software* e redes de comunicação difusa se fazem necessárias e formam a segunda base. Por último, ferramentas analíticas que objetivam o comportamento autônomo e inteligente da rede compõe a terceira base (BRUCKNER, 2013).

O Instituto de Engenheiros Eletricistas e Eletrônicos (Institute of Electrial and Electronics Engineers) explica que um sistema baseado no conceito de IoT pode ser definido como um sistema autoconfigurável e adaptável que consiste de uma rede de sensores e objetos inteligentes cujo propósito comum é a interconexão de todas as ‘coisas’, sejam estas usadas no dia a dia ou usados pela indústria, de forma a fazê-los ‘inteligentes’, programáveis e mais capazes de interagir com humanos (ISIKDAG, 2015). Todavia, não existe um padrão na

construção da arquitetura de sistemas fundamentados no IoT. A falta desse padrão contribui para a confusão sobre o que é o conceito de IoT. Entretanto, pesquisadores apresentam os componentes mínimos derivados a partir dos quesitos que o sistema deve satisfazer. Os componentes de *Hardware* incluem:

- sensores/atuadores;
- unidades de processamento como microprocessadores e SBCs (*Single Board Computers*);
- unidades de comunicação como RFID, NFC, Wi-Fi e etc;
- unidades de armazenamento (MINERVA, BIRU e ROTONDI, 2015).

Para fomentar a interligação e comandos nos componentes de *Hardware* faz-se necessário a inclusão do *Software*. Este é formado de diferentes camadas, estas são:

1. sistema operacional (advinda do *Hardware*);
2. protocolos de comunicação (permitem a comunicação entre dispositivos com diferentes camadas de *Software*);
3. componentes de *middleware* (facilitam a comunicação e o compartilhamento de informações entre dispositivos);
4. portais de integração (para combinar a integração da informação obtida de múltiplos dispositivos e apresentar essa informação ao usuário) (ISIKDAG, 2015).

3 SISTEMA DE SEGURANÇA E MONITORAMENTO

3.1 Visão geral do Sistema

Considerando a motivação para a construção da proposta e os benefícios apresentados sobre a automação residencial e a IoT, esta baseia-se na implementação de um sistema de segurança capaz de gerenciar e monitorar um ambiente central de uma residência, localmente e remotamente. Sensores e atuadores são conectados a um controlador principal que é interligado a uma plataforma *web*. O controlador é onde são centralizadas as informações e permite o gerenciamento das funcionalidades por parte do usuário através da plataforma, além de interagir com o sistema/ambiente de forma autônoma a partir de funções de controle predefinidas. Uma visão geral simplificada da estrutura pode ser observada na figura a seguir.

Figura 4 - Representação visual do projeto.

Fonte: Produção do próprio autor.

O projeto realizado pode ser subdividido em duas partes constituintes básicas: A interface do usuário e o protótipo. Como visto na Figura 4, o controle do sistema pode ser realizado através de comandos advindos do usuário, os quais são interpretados pelo sistema que então executa uma ação. O sistema de segurança, que se encontra instalado na residência, é

interligado à plataforma gráfica através da conexão com Internet realizada via WiFi. Nesta proposta, as interações assim como a visualização do monitoramento por imagens do ambiente feita pelo morador são realizadas a partir de uma aplicação *web*.

O protótipo inclui um segmento com base eletrônica que servirá de amparo para a composição dos componentes de *hardware*, e outro formado por um conjunto de códigos computacionais que serão responsáveis pela coleta, análise e realização das demandas. O *software* constitui o elo principal no processo entre recebimento das informações de controle pelo usuário e/ou pelo sistema (automáticas) e a concretização das requisições.

A alternativa de estratégia de segurança disposta, é um recurso que objetiva ser uma plataforma de fácil utilização com ferramentas que possuem a finalidade de conceder uma experiência positiva ao usuário. O sistema propõe-se a responder prontamente a incidentes de alarme, comandando os recursos de forma automática e inteligente, sendo o maior benefício é a segurança e concomitante facilidade no uso.

3.2 Concepção Geral Sobre o Funcionamento

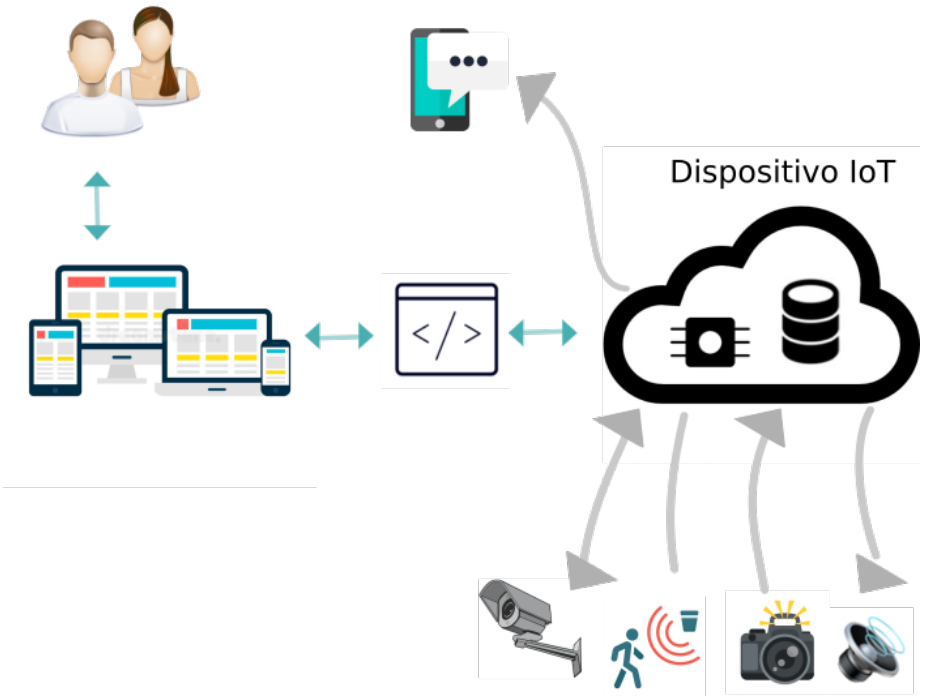
O alicerce de comunicação da *Internet* das Coisas é constituído por quatro componentes principais: sensores e atuadores na residência, controlador, servidor e a interface do usuário (ROCHA, 2017). O protótipo foi desenvolvido para efetuar o controle e gerenciamento de dados no controlador, servidor e dispositivos (sensores e atuadores) e compor a interação do sistema com a interface por meio de uma rede sem fio.

Para atingir o objetivo proposto, a estrutura é composta de uma solução integrada de *hardware* e *software* a qual abrange as etapas de coordenação e configuração dos seguintes itens:

- dispositivo IoT;
- sistema de transmissão de vídeo via *streaming*;
- sistema de captura de imagens;
- sistema de detecção de presença;
- sistema de alarme sonoro;
- sistema de notificação.

Essas partes conectadas em sua devida ordem irão conceber o sistema de monitoramento residencial utilizando os conceitos apresentados no capítulo anterior. A Figura 5 mostra a estrutura organizacional utilizada neste trabalho, incluindo a representação dos componentes que fazem parte do sistema.

Figura 5 - Arquitetura conceitual do sistema de segurança residencial.



Fonte: Produção do próprio autor.

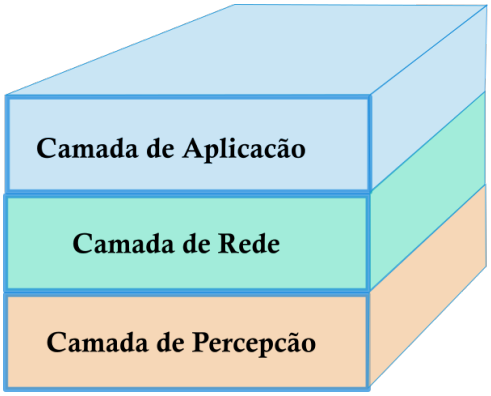
A Figura 5 ilustra o aspecto geral sobre a estrutura do sistema de segurança. É possível observar que o usuário, que conta com uma aplicação acessada através de um navegador *web*, constitui uma parte do sistema que é conectada ao dispositivo IoT por meio de uma página no navegador. O dispositivo IoT é o componente principal do protótipo de automação residencial, este atua como um servidor e controlador de dispositivos, sendo o elo central responsável por receber requisições e transmitir informações e exigências do sistema, tanto para a interface apresentada como para os componentes do protótipo.

4 DESENVOLVIMENTO E IMPLEMENTAÇÃO

Segundo Ogata (1997), três entidades básicas compõe o modelo de um controle automático: controlador, sensores e atuadores. O fluxo de funcionamento do sistema inicializa por meio de leitura de dados através dos sensores que, por conseguinte, enviam as informações para o controlador a cada alteração na leitura. As informações são centralizadas no controlador o qual possui parâmetros de controle predefinidos. Segundo a lógica estabelecida no controlador o mesmo pode realizar ações programadas de execução e/ou ativação de atuadores.

No contexto de controle com bases na *Internet* das Coisas os elementos da arquitetura básica de controle permanecem, no entanto, são acrescidos de novos elementos de lógica e comportamento de sistema. WU, LU, *et al* (2010), faz a previsão de uma arquitetura básica que reflete os componentes lógicos e estruturais básicos de uma proposta de sistema baseado na IoT é que se assemelha ao pretendido neste trabalho. A arquitetura possui três camadas, tais são representadas no diagrama da Figura 6.

Figura 6-Modelo de organização de camadas da arquitetura de projetos IoT proposta por Wu et al.



Fonte: Produção do próprio autor.

A partir da Figura 6 pode-se observar que a base da arquitetura proposta é a chamada camada de percepção, responsável pela percepção de informações do ambiente, obtenção de informação e controle de componentes de atuação no ambiente a partir de comandos da camada de aplicação. Segundo WU, LU, *et al* (2010), nesse conceito de arquitetura a camada central é a de rede e representa o centro de processamento de dados, informações e onde a inteligência do sistema reside. Por fim, a camada de aplicação representa a parte do projeto

onde as aplicações computacionais localizam-se. Esta camada recebe dados, processa ações e envia comandos à camada de rede para que então essa acione as ações na camada de percepção.

O protótipo servirá de amparo para construção dos elementos de *hardware* coletando e analisando as demandas do usuário e do sistema (camada de rede e percepção) e a interface será responsável por incorporar as requisições do usuário no sistema (camada de aplicação).

Este capítulo baseia-se no estudo o desenvolvimento do *hardware* e do *software* do sistema. Os componentes de *hardware* são os dispositivos físicos como os sensores, câmera e controlador. O *software* é composto pela parte lógica, ou seja, o conjunto de comandos e instruções que circulam no sistema que podem ser passados ou não para os componentes físicos (*hardware*) para que esse possa executar determinada tarefa.

4.1 Materiais

A fim de cumprir as especificações deste projeto foi determinado a utilização dos seguintes componentes de *Hardware*:

- Raspberry Pi;
- 8GB SD card;
- WiFi dongle;
- Câmera Raspberry Pi;
- Sensor piroelétrico (PIR);
- Buzzer;
- LED.

A seguir, serão apresentadas as principais especificações técnicas dos dispositivos que compõe a parte física do projeto. Será alisada as características e detalhadas as especificações e funcionalidades dos componentes principais, assim como o fundamento para estabelecer essa escolha.

4.1.1 Raspberry Pi

A plataforma computacional definida para realizar a função de módulo central do sistema foi a *Raspberry Pi*. Esse microcomputador foi criado pela *Raspberry Pi Foundation* no Reino Unido em 2006, a idéia principal da fundação é a de desenvolver um produto com preço acessível, com o intuito de incentivar pessoas de todas as idades a aprender a programar, estimulando a produção digital. O dispositivo, considerado como um computador em miniatura, é compatível com vários sistemas operacionais (SO) diferentes em sua plataforma inclusive o novo sistema Windows 10 poderá operá-lo. Apesar de possuir o tamanho de um cartão de crédito, o *Raspberry Pi* pode ser aplicado a inúmeras finalidades, como projetos de eletrônica, reprodução de vídeo de alta definição, além de muitas atividades que um computador convencional executa (RASPBERRY PI FOUNDATION).

O modelo de placa escolhido foi o *Raspberry Pi* 2 e ele foi utilizado em conjunto com um cartão de memória micro SD e um Dongle WiFi. O cartão SD foi pré-instalado com o sistema operacional oficial do *Raspberry Pi*, *Raspbian Jessie*, que é uma distribuição baseada no SO Linux (Debian). O adaptador USB *Wi-Fi dongle* permite que o microcomputador possa ser conectado à *Internet* via *Wi-Fi*, que é uma das especificações do projeto. A Figura 7 apresenta o Raspberry e os componentes complementares utilizados para compor a plataforma computacional.

Figura 7-*Raspberry* Pi 2 e os componentes para ajuste de seu funcionamento

Fonte: DEPHACE, (2017).

As principais características do *Raspberry* Pi 2 e suas especificações são apresentados a seguir.

Quadro 1-Características e especificações do *Raspberry* Pi2.

Característica	Especificação
SoC	Broadcom BCM2836 (CPU,GPU,DSP,SDRAM)
CPU	900MHz quad-core ARM Cortex-A7
GPU	VideoCore IV 3D
Memória (SDRAM)	1GB compartilhada com a GPU
Ethernet	10/100MB – interface RJ-45
Portas USB	4 x USB 2.0
Saída de vídeo	HDMI e DSI
Saída de áudio	HDMI e AV via 3.5mm conector de áudio
GPIO	40 pinos
Alimentação	5V - porta micro USB

Fonte: Produção do próprio autor.

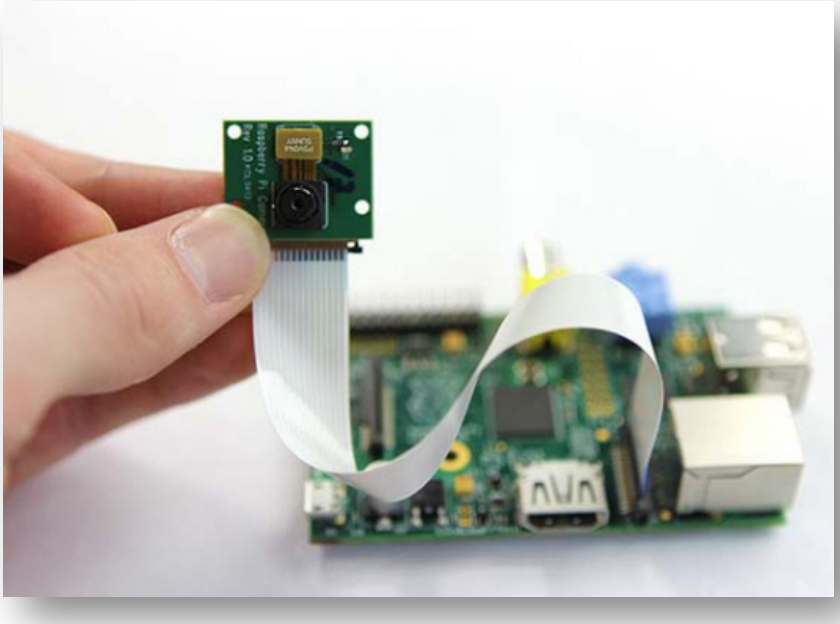
A partir do Quadro 1, é possível inferir que a plataforma pode ser utilizada como módulo servidor responsável pelo processamento dos dados obtidos. Este possui capacidade suficiente de processamento para realizar as tarefas do projeto proposto, além de facilitar a integração de outros componentes necessários para o desenvolvimento do mesmo através de suas interfaces como a interface de câmera e GPIO (*General Purpose Input/Output*). O *Raspberry Pi* é capaz de realizar todas as funções requeridas pela IoT e pode facilmente ser conectado a periféricos permitindo realizar controle de E/S, possibilitando leitura de sensores e saídas para controlar dispositivos, possibilitando a conexão com a *Internet* e permitindo o desenvolvimento de código de programação para emular inteligência (RASPBERRY PI FOUNDATION).

4.1.2 Câmera Raspberry Pi

A Câmera *Raspberry Pi* foi criada com o intuito de expandir o contato das pessoas com a plataforma *Raspberry Pi* e foi desenvolvida pensando nos entusiastas de projetos e plataformas de prototipagem. Trata-se de uma pequena placa de circuito impresso agregada a uma câmera digital capaz de capturar imagens e realizar filmagens com alta qualidade e

definição de forma simples. A PCB (Placa de Circuito Impresso) é conectada a uma fita plástica que interliga o circuito na plataforma *Raspberry Pi*. Este, possui interface CSI (*Camera Serial Interface*) dedicada para conexão de câmeras poupando o uso da porta USB (*Universal Serial Bus*) na conexão. A Figura 8 a seguir apresenta o módulo de câmera digital em associação com o *Raspberry Pi*. (RASPBERRY PI FOUNDATION).

Figura 8-Apresentação do módulo de câmera conectada ao *Raspberry Pi*



Fonte: WILLIAMS, (2013).

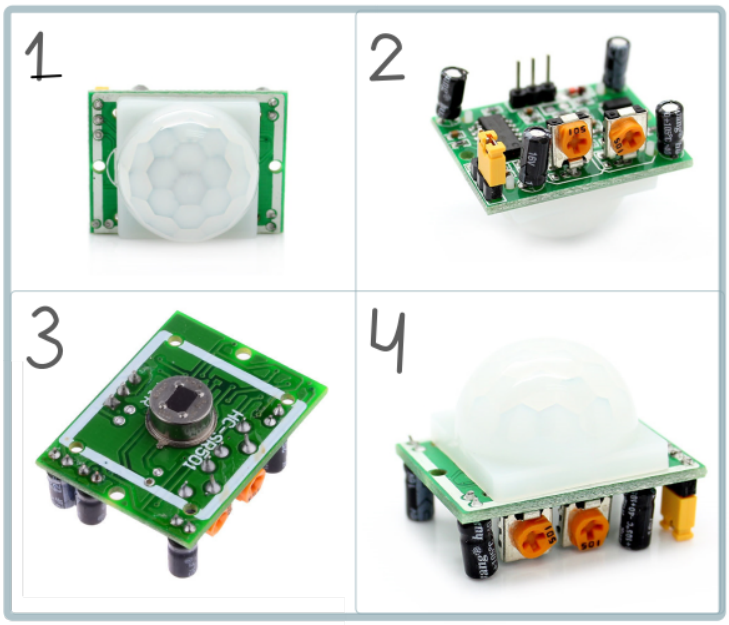
O *design* da câmera foi personalizado para que esta seja alimentada e comandada pelo *Raspberry Pi*. Todo o *software* da câmera é suportado pela última versão do sistema operacional *Raspbian*. O modelo definido foi a *Raspberry Pi Camera Module* v2 que é a última versão lançada pelos criadores e possui as seguintes características básicas: 8 *megapixels* de resolução, sensor de imagem SonyIMX219, lente de foco fixo e pesa apenas 3g (RASPBERRY PI FOUNDATION).

Como pode-se observar da Figura 8 e das características do componente, a placa de circuito é leve e pequena, ou seja, além dos benefícios em termos de desempenho que o módulo oferece, este ocupa pouco espaço o que facilita seu emprego em sistemas de segurança baseados em plataforma de prototipagem.

4.1.3 Sensor de Presença Piroelétrico (PIR)

Os sensores são utilizados para monitorar e coletar dados sobre o espaço físico no qual estão inseridos e transformá-los em sinais elétricos que são posteriormente processados e interpretados. O sensor piroelétrico PIR (*Passive Infrared Sensor*) é chamado de sensor passivo pois este recebe a luz infravermelha emitida por corpos que emitem calor e envia sinal elétrico ao controlador, informando a movimentação no ambiente. A imagem a seguir apresenta esse sensor visto de várias posições.

Figura 9-Imagens do sensor piroelétrico PIR vista de várias posições de forma a expor seus componentes.



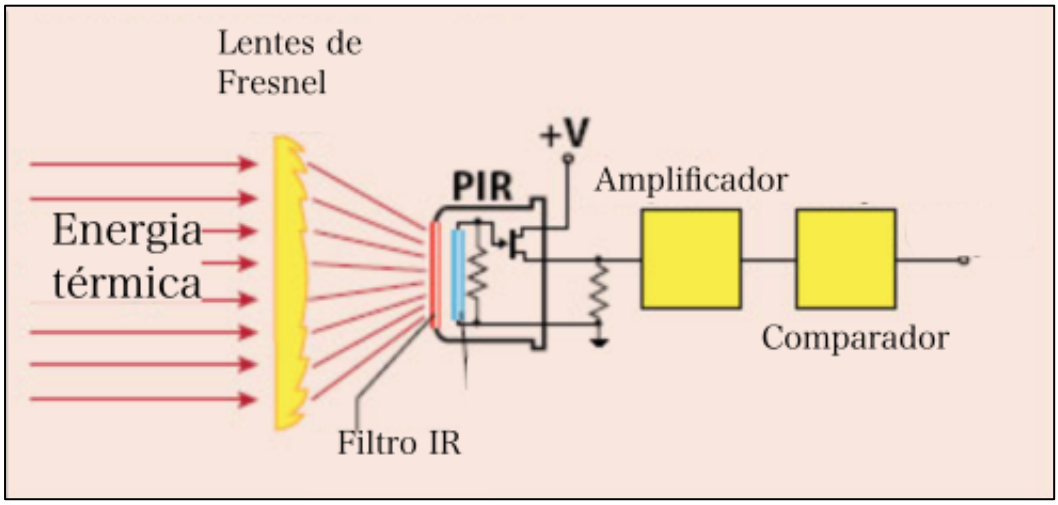
Fonte: Produção do próprio autor.

O PIR é composto pelas lentes de Fresnel as quais podem ser observada na parte 1 da Figura 9, estas lentes servem para condensar a radiação de modo a gerar uma quantidade maior de raios infravermelhos para o sensor, mostrado mais claramente na parte 3. O circuito com um *chip* embutido, que pode ser visto na parte 2 da Figura 9, atua como amplificador e comparador. Ajustes de tempo de saída e sensibilidade podem ser realizados neste sensor a partir dos pequenos parafusos laranjas mostrados na parte 4 da Figura 9. O ajuste de tempo é realizado para determinar o período que a saída lógica do sensor se mantém alta e o ajuste de

sensibilidade estabelece a distância ao PIR que a área de detecção pode alcançar. O módulo detectará movimento dentro de um ângulo de 110 graus. (EVER PI, 2015).

As etapas do funcionamento composta pelos componentes do sensor citadas podem ser melhor compreendidas a partir da Figura 10.

Figura 10- Diagrama de bloco de funcionamento do sensor PIR.



Fonte: Adaptado Ever Pi, (2015).

Após o movimento ter sido percebido pelo sensor o comparador ajusta a saída para que lógica de sinal permaneça alta por um período de tempo regulável no sensor, chamado de tempo de saída, que pode variar de 3 a 5 minutos. Depois de passados o tempo de saída o sensor permanece por 3 segundos com a lógica baixa na saída (*time off*) até outro ciclo de detecção poder ser iniciado. Durante esse tempo a detecção de presença fica bloqueada.

4.1.4 Alarme Sonoro (Buzzer)

O *Buzzer* ativo é uma pequena estrutura simplificada e integrada de transdutores utilizada como auto-falante destinado a emitir sinais a partir da alimentação DC. O sinal sonoro de alerta de intrusos é reproduzido por este componente conectado ao *Raspberry PI* de forma a alertar moradores e talvez afugentar possíveis invasores. A imagem do *buzzer* pode ser visualizada a seguir.

Figura 11 - *Buzzer* ativo utilizado como alarme sonoro.

Fonte: Produção do próprio autor.

4.2 Desenvolvimento do *Software*

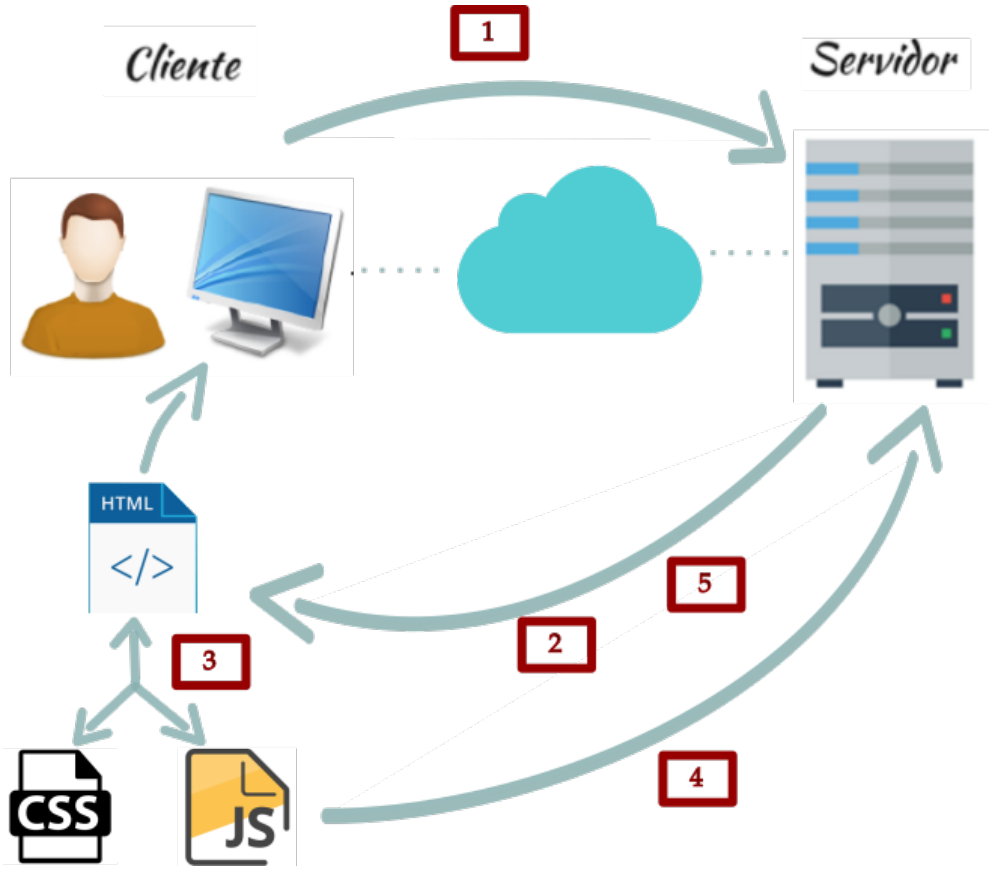
Aplicações *web* devem ser hospedadas em um servidor, este, capaz de atender às solicitações de vários clientes que possam acessar a aplicação (RASPBERRY PI FOUNDATION). Sendo assim, o modelo de desenvolvimento foi baseado no conceito de cliente-servidor, que é uma estrutura distribuída de aplicação. Este formato de arquitetura é baseado no modelo OSI (*Open System Interconnection*), em que as tarefas são distribuídas em camadas lógicas, facilitando a performance dos sistemas com um ou mais usuários. O cliente e o servidor podem residir num mesmo computador, entretanto, o cliente apenas solicita dados e funções, mas não compartilha seus recursos com o servidor. Neste modelo, o cliente é representado pela interface e suas requisições e o servidor pelo *Raspberry Pi* que gerencia todos os recursos. O *software* é essencial para o projeto, sendo este o responsável pela integração do sistema.

4.2.1 Client-Side

O lado do cliente (*client-side*) foi implementado utilizando as linguagens HTML (*Hypertext Markup Language*), JavaScript e CSS (*Cascade Style Sheet*). O HTML é uma linguagem básica da Internet e utilizada para desenvolver *websites*. Essa linguagem é considerada como

a estrutura ou ‘esqueleto’ do site, enquanto o CSS configura o visual da página e o JavaScript lida com a execução de funções no lado do cliente e faz requisições ao servidor. A figura a seguir ilustra como essas linguagens de programação atuam no *client-side*.

Figura 12-Visão geral do funcionamento do *Software* no lado do cliente.



Fonte: Produção do próprio autor.

A interface *web* projetada deve permitir que o usuário gerencie e monitore o sistema de segurança. Para tal, a interface é acessada por meio de um navegador *web*. A Figura 12 demonstra como esse acesso é executado e as solicitações que são feitas ao servidor. A partir da Figura 12, observa-se que os processos realizados no lado cliente foram divididos em 5 etapas principais básicas. Na primeira, o navegador efetua a requisição da página ao servidor *web*, onde a página está hospedada, através de uma requisição HTTP (*Hypertext Transfer Protocol)*. O protocolo HTTP é o meio pelo qual o padrão cliente-servidor se concretiza. O cliente faz a requisição (HTTP *request*) e a aplicação no servidor responde (HTTP *response*) à solicitação. Essa resposta é compreendida na etapa 2 do diagrama, onde o servidor retorna a página HTML para o cliente.

Após o recebimento da página HTML advinda do servidor, o processamento da página é realizado invocando os recursos dos arquivos JavaScript e CSS para alterar a página, como ilustrado na terceira etapa. O arquivo JavaScript possui toda lógica computacional que vai rodar no cliente. Quando o usuário realiza o chamado de alguma função do JavaScript, este responde a função ligada ao evento sem que a página seja recarregada. Caso a requisição seja feita do HTML para o servidor, o JavaScript funciona como intermediário, encaminhando a requisição (etapa 4), e o servidor responde ao HTML diretamente (etapa 5).

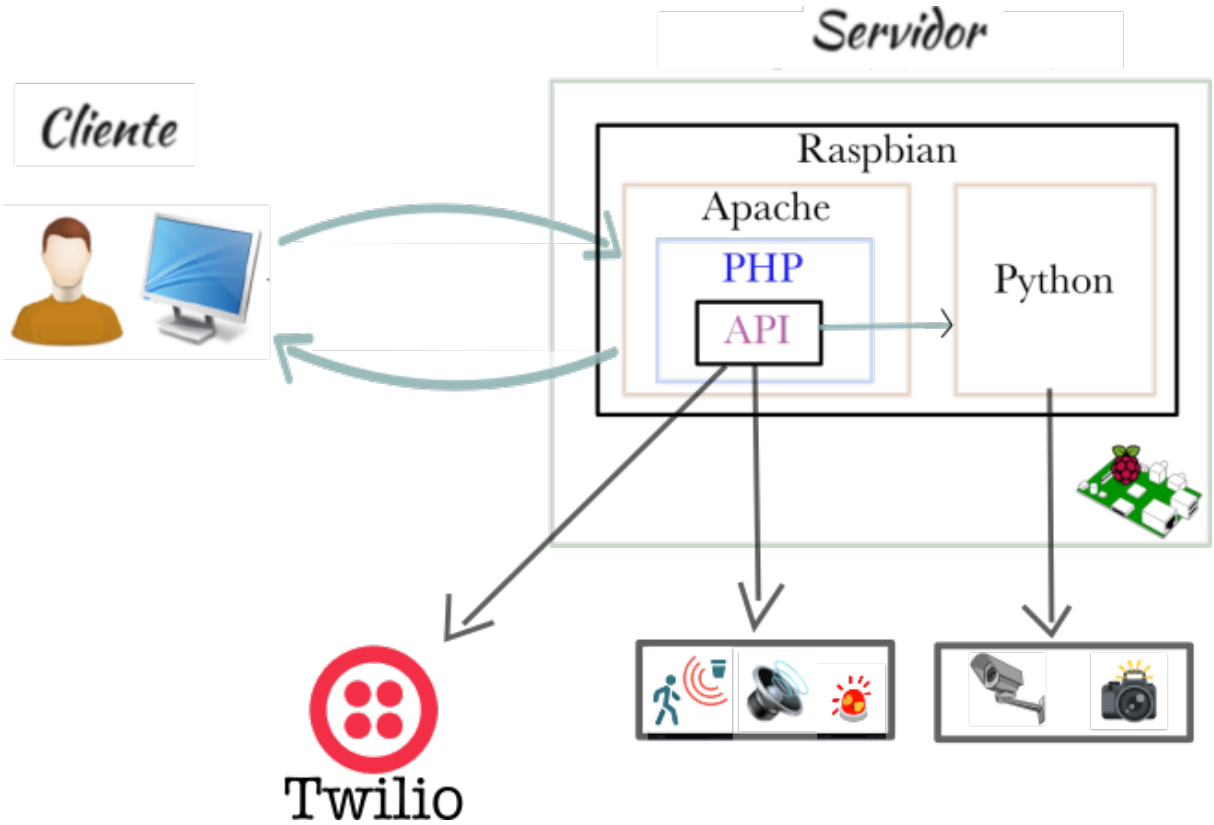
A aplicação desenvolvida é baseada no conceito de SPA (*Single Page Application*). O SPA é um modelo de desenvolvimento de aplicações em que todo conteúdo localiza-se em uma única página. Ou seja, quando o navegador fizer a requisição na etapa 1 da Figura 12, todos os templates e conteúdos da aplicação serão carregados no lado do cliente na primeira requisição. As requisições ao servidor que possam ser necessárias serão apenas de tráfego de dados entre o cliente e o servidor. Em modelos diferentes o usuário visita várias páginas diferentes, e para cada uma o servidor precisa processar uma requisição. Portanto, ao *client-side* será delegada mais lógica computacional e a aplicação estará rodando mais no navegador do que no servidor, gerando melhorias de desempenho no sistema. O *framework* escolhido para ser utilizado para implementação do modelo foi o Aurelia, o qual é um *framework* JavaScript que possui recursos para manipulação dinâmica do DOM (*Document Object Model*).

4.2.2 Server-Side

O *Raspberry* é o componente principal de controle do sistema. De forma geral, o microcomputador atua como o servidor do sistema onde é responsável por receber e transmitir as informações do cliente, assim com realizar a sincronização e gerenciamento de dados dos dispositivos associados a este. Os dados coletados são processados no servidor e utilizados para tomada de decisão através do uso das ferramentas de programação Python e PHP (*Hypertext Preprocessor*).

Existem diferentes possibilidades de tecnologias disponíveis que transformam um computador em um servidor *web* de forma simples. A definição de qual ferramenta utilizar está intimamente ligada à linguagem de programação que se deseja fazer uso para desenvolver as funções no servidor. Dentre as possibilidades, destaca-se o Apache, que é uma implementação de *software* que executa códigos em PHP. A arquitetura no lado do servidor pode ser melhor entendida através da Figura 13 a seguir.

Figura 13-Arquitetura do sistema no lado do servidor.



Fonte: Produção do próprio autor.

A Figura 13, representa o relacionamento de cada parte do servidor e como este é estruturado. Conforme a arquitetura planejada para o *Raspberry Pi*, foi escolhido o sistema operacional *Raspian* em conjunto com um servidor Apache instalado. O Apache é um gerenciador HTTP que roda na porta 80 e possui um módulo PHP instalado. No momento em que o cliente faz uma requisição para o endereço da aplicação, o Apache recebe a requisição HTTP e aciona o módulo PHP para tratar as rotas da requisição. O PHP é o programa que gerencia a API

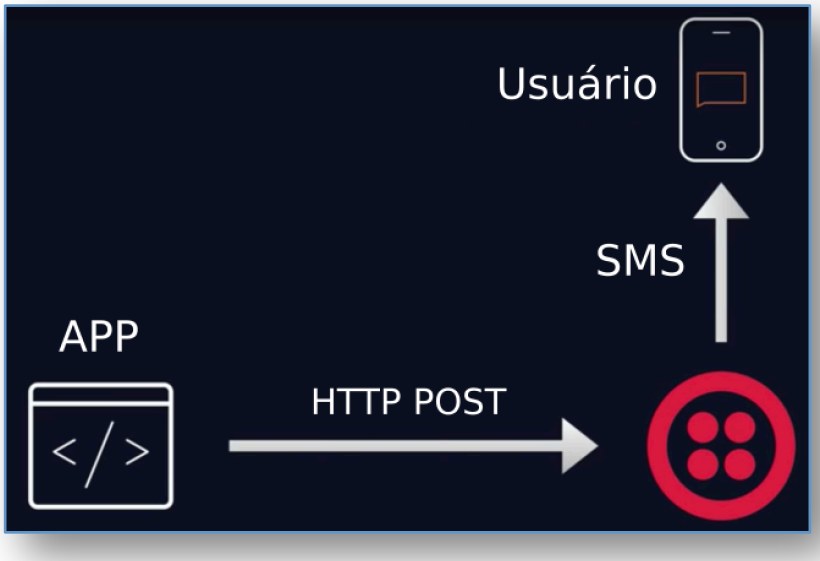
(*Application Programming Interface*) do servidor *web*. Esse conjunto de tecnologias atua como servidor *web* conectado à *Internet*.

API são programas que dão suporte ou promovem a interação entre sistemas mesmo que estes possuam linguagens completamente distintas, permitindo que criadores de *software* desenvolvam produtos associados a diferentes serviços sem que se conheça as informações intrínsecas do *software* desenvolvido na API, logo a aplicação não é exposta somente suas funcionalidades que acessadas através de requisições. Para simplificar a forma com que sistemas se comunicam com a API, foi criada a arquitetura REST, que se utiliza de verbos específicos nas rotas de requisição usando de padronizados para mapear estas, em ações HTTP de POST, GET, PUT e DELETE.

A lógica de programação na API foi decidida ser realizada em PHP e se resume a um conjunto de métodos e instruções que promovem o gerenciamento dos dispositivos de *hardware*, efetua requisições de serviço à API de envio de SMS, trata dos processos de correspondência entre o cliente e o servidor e interage com o código em Python, o qual aciona as funções de câmera. A API utiliza-se do padrão de arquitetura REST para realizar as requisições HTTP do entre servidor e cliente e a API Twilio de envio de mensagem. Quando o usuário faz uma requisição através da interface para o servidor, um código em JavaScript é executado através dos métodos POST, GET, PUT e DELETE e em seguida os parâmetros são passados para o arquivo PHP que faz a verificação da ação que deverá ser executada.

Para executar o envio de SMS alertando o morador sobre uma possível invasão, foi escolhida a comunicação entre a API em PHP e outra chamada Twilio através do método POST. O Twilio é um serviço que possui toda sua estrutura baseada num servidor em nuvem e permite que desenvolvedores possam realizar a integração de voz, SMS, vídeo, entre outras, em suas próprias aplicações, possibilitando eficaz comunicação entre dispositivos móveis, aplicações e serviços (LEONEL, 2015). A Figura 14 ilustra a forma como é feita a requisição a partir da aplicação em PHP até o envio do SMS ao dispositivo.

Figura 14-Diagrama representativo do funcionamento das requisições feitas à API Twilio.



Fonte: LEONEL, (2015).

5 IMPLEMENTAÇÃO

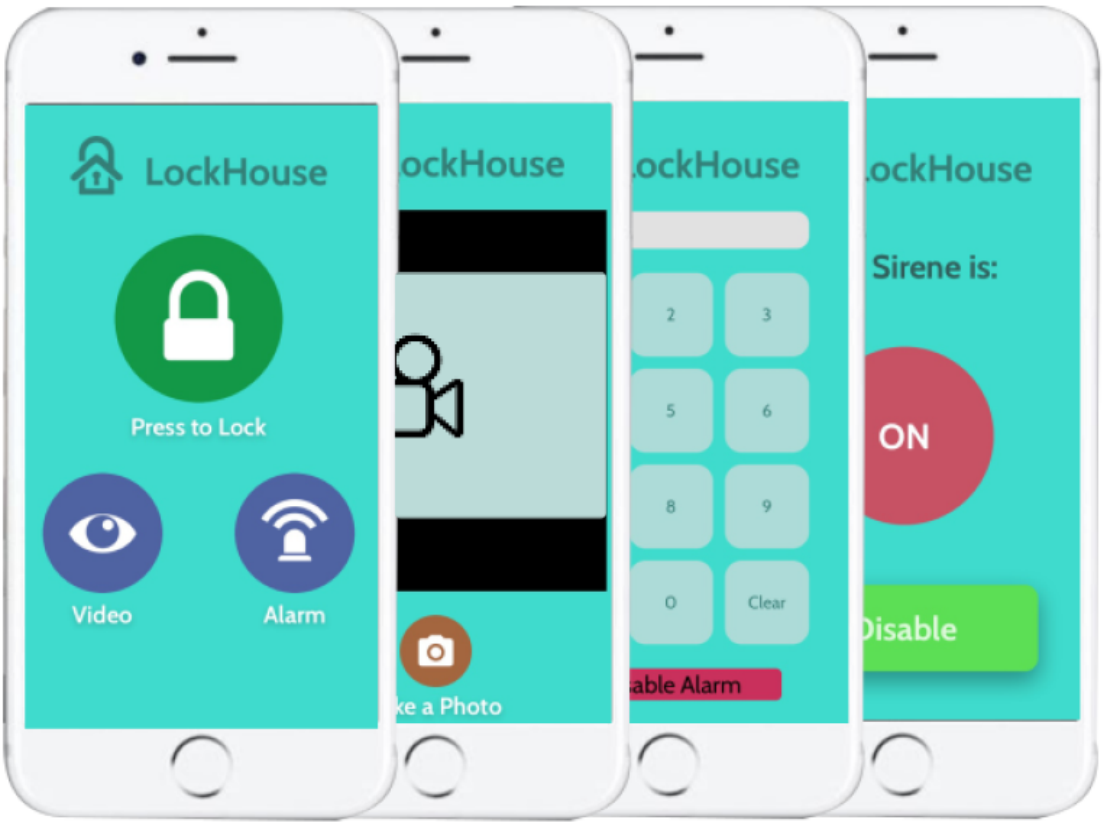
A abordagem a este projeto baseia-se na construção das partes componentes do mesmo. O protótipo servirá de amparo para construção dos elementos de *hardware* coletando e analisando as demandas do usuário e do sistema e a interface será responsável por incorporar as requisições do usuário no sistema. O *Design* do *hardware* e do *software* e a integração destes no sistema são expostos em detalhes no transcorrer deste capítulo.

5.1 Interface do Usuário (UI)

A criação de novos aplicativos para auxiliar as pessoas nas inúmeras atividades tem sido impulsionada por uma vasta utilização de dispositivos móveis como celulares e *tablets* (SILVA, 2014). Todavia, não existe um padrão de concepção e criação de aplicações para esses aparelhos, dependendo do sistema operacional e tipo do item as aplicações podem ser construídas a partir de uma variedade de linguagens de programação e utilizando configurações distintas dependendo do dispositivo. Por este motivo, foi desenvolvido o *LockHouse*, uma aplicação *web* para controle e gerenciamento do sistema de automação residencial. Esta ferramenta computacional foi projetada para ser acessada e utilizada através de um navegador, de forma responsiva, ou seja, possibilitando a adaptação a vários dispositivo que tenha acesso à *Internet*, e não necessitando de um sistema operacional ou navegador específico para utilização do sistema.

A interface do usuário é a componente onde se constrói visualmente a parte tangível do projeto, possibilitando a interação do usuário com o sistema. Logo, essa plataforma precisa ser capaz de transmitir claramente todas as suas funções sem muitos elementos de UI (*User Interface*), assim posto a interface seja atrativa e fácil de ser utilizada. O *LockHouse* foi criado especificamente para permitir a interação de usuários com o protótipo, consumindo dados do servidor *web* e ativando funções neste. O aplicativo e seus respectivos *templates* podem ser visualizados na Figura 15.

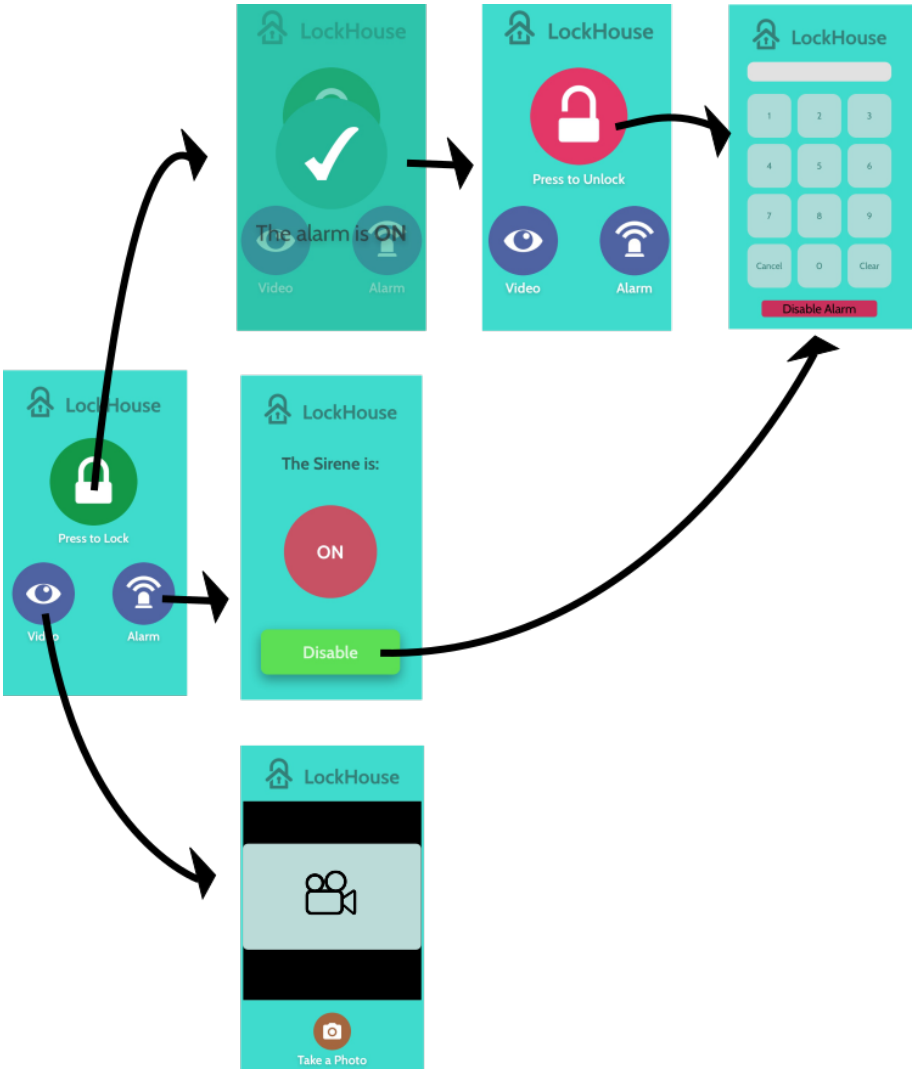
Figura 15-Telas que compõem a Interface da aplicação.



Fonte: Produção do próprio autor.

Para auxiliar a compreensão da aplicabilidade da interface o diagrama abaixo mostra o fluxo de apresentação dos *templates* de acordo com ações escolhidas pelo usuário.

Figura 16 - Fluxo de funcionamento da Interface do usuário.



Fonte: Produção do próprio autor.

Seguindo as imagens da Figura 16 a primeira tela a ser apresentada é o *template* principal, esta mostra as funções básicas da interface, as quais, são funções de ativação/desativação do sistema de alarme, câmera e alarme sonoro. A tela principal informa o usuário sobre o *status* do alarme apresentando um símbolo de cadeado fechado ou aberto se o alarme estiver desativado ou ativado, respectivamente.

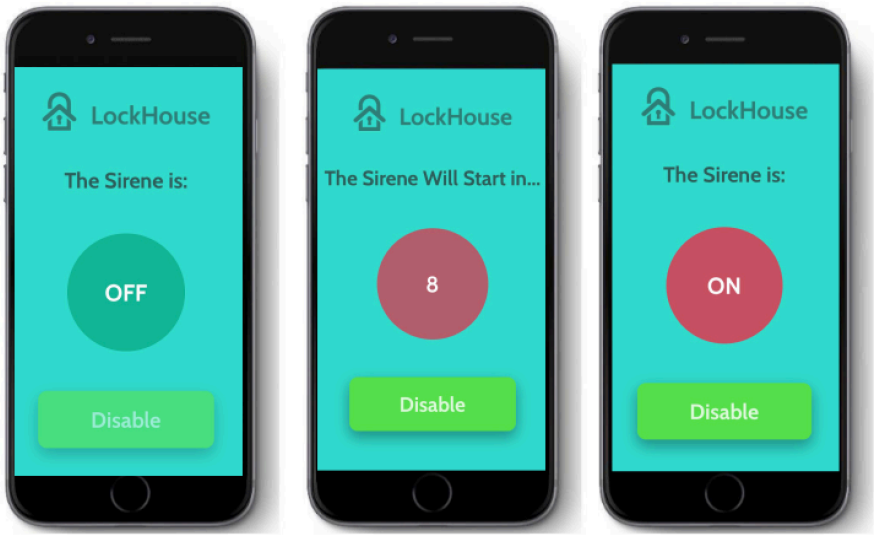
Supondo que o alarme esteja desativado no momento do acesso do usuário à interface, este pode acionar o alarme apenas clicando no símbolo que representa um cadeado fechado, uma tela de confirmação será apresentada e então o usuário pode prosseguir e acessar outras

funcionalidades do sistema se desejar. Caso o alarme esteja ativado, a desativação pode ser executada através do botão com símbolo de cadeado fechado que será apresentado na interface principal direcionando-o para a página de desativação de alarme, onde será necessário informar uma senha para que o sistema de alarme seja desativado.

A transmissão de *streamings* é realizada 24h por dia, porém a captura de imagens através da câmera demandam o acionamento por parte do morador para ser iniciada. O monitoramento pode ser concretizado através da tela de câmera e a obtenção de imagens do ambiente pode ser realizada a qualquer momento que o usuário achar necessário, bastando clicar no botão de captura de imagem na tela de câmera. A interface então envia um comando para o servidor notificando que a imagem deve ser capturada e armazenada.

A tela de alarme sonoro apresenta três templates diferentes como mostra a imagem a seguir.

Figura 17-Tela de alarme sonoro e seus três estados



Fonte: Produção do próprio autor.

A partir da Figura 17 observa-se que a tela de alarme sonoro possui três estados. O primeiro estado representa o momento em que o alarme está desativado, logo se o usuário acessar esta tela, observará que esta indicará que o sinal sonoro está desligado (OFF). Supondo que o alarme esteja ativado, ao constatar que a residência foi invadida o sistema inicia um contador para que ao final da contagem a ativação do alarme sonoro seja efetuada. A tela de alarme sonoro então apresentará o contador ou indicará que o sinal sonoro está ligado (ON). Em

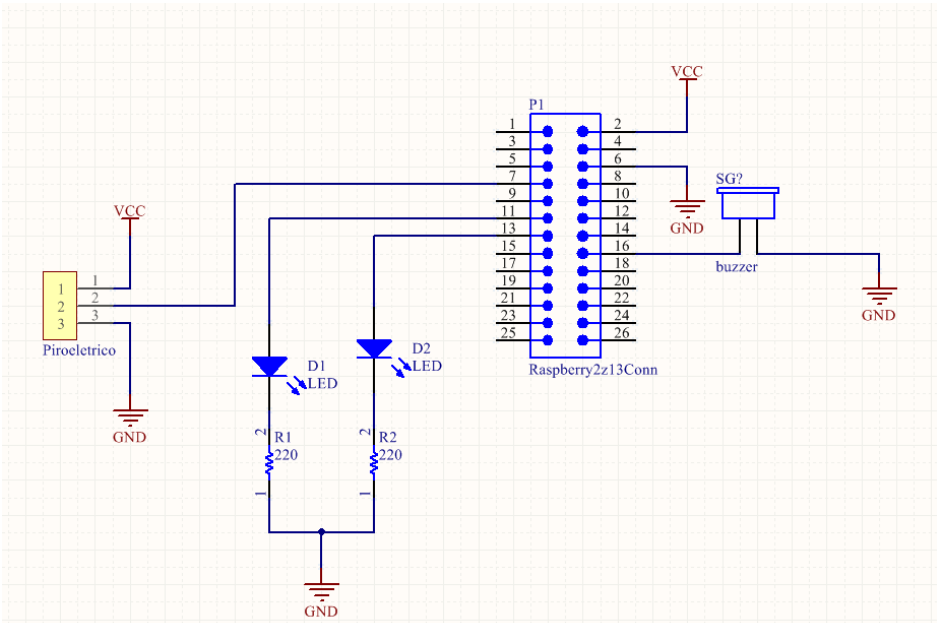
qualquer desses dois estados será apresentado na interface um botão de desativação de alarme que redireciona a aplicação para a tela desativação do alarme. Ou seja, caso o usuário perceba que o intruso não oferece risco para seu patrimônio, o mesmo pode desativar o sistema a qualquer momento, antes mesmo que o alarme sonoro seja ativado. Se o intruso oferecer risco ao usuário, este pode então proceder com o acionamento das autoridades e tomar as devidas providências para que a ação criminosa seja interrompida. Além disso, o mesmo pode efetuar a captura de imagens a qualquer momento se achar necessário.

Dessa forma, sempre que o cliente desejar realizar alguma ação, a informação é enviada para o servidor que manipula esses dados e faz a requisição para que o controlador a realize. Semelhantemente, a interface também recebe a informação do servidor caso haja violação na residência.

5.2 Protótipo

A Figura 18 a seguir, mostra o esquemático dos componentes *Hardware* implementados apresentando as conexões físicas de todos os componentes ao *Raspberry Pi*, com exceção da câmera. O circuito mostra o *Buzzer* e o sensor piroelétrico PIR, já detalhados anteriormente, além de *LED* e resistores, os quais são utilizados para indicar o acionamento do alarme pelo usuário e do *Buzzer* pelo controlador.

Figura 18-Esquemático desenvolvido para desenvolvimento do circuito do protótipo

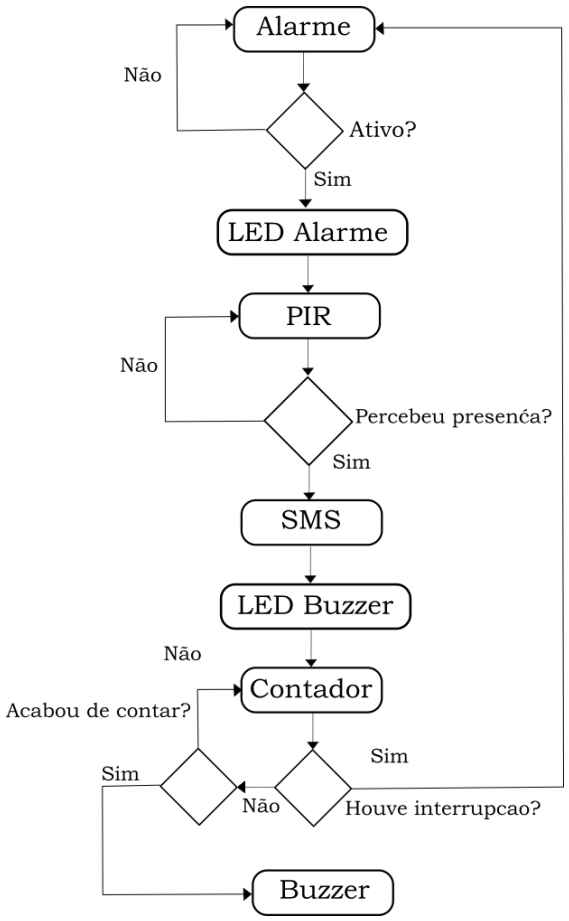


Fonte: Produção do próprio autor.

5.3 Implementação da lógica do sistema

A inicialização do sistema independe da intervenção do usuário para ser instaurada. O item de transmissão de vídeo via *streaming* é inicializado assim que o protótipo for conectado à energia. Todavia, as funções específicas de detecção de intrusos constituídos pelos itens de notificação, alarme sonoro e captura de imagens são iniciadas apenas a partir do momento em que o morador entende que qualquer movimentação no ambiente é considerada uma ameaça e/ou deve ser notificada. A lógica de programação de *software* realizada nesta etapa pode ser visualizada no fluxograma da Figura 19.

Figura 19-Lógica de controle com alarme ativo.



Fonte: Produção do próprio autor.

A partir do fluxograma da Figura 19, é possível observar que quando o alarme for acionado pela interface, o sistema inicializa o modo de detecção de intrusos que constitui um modo de atuação autônoma do sistema. Ao receber o comando de acionamento de alarme o controlador

gerenciará o desencadeamento de outras ações. Essas ações seguem o seguinte padrão de chamada caso não haja intervenção no sistema por parte do usuário:

- reconhece o comando de acionamento do alarme;
- aciona o sistema de detecção de presença no ambiente;
- chama a função para envio de notificação;
- acionamento do LED de sinalização de intruso;
- chama a função para iniciar o contador do alarme sonoro;
- chama a função que aciona o alarme sonoro;

Ou seja, após o acionamento do alarme o controlador começa a verificar qualquer movimentação no ambiente. Na hipótese de isso acontecer o sistema entende que há uma possível ameaça a residência e a informação enviada para o dispositivo IoT que gerenciará o restante do desencadeamento e tomada de decisão.

Os componentes foram configurados de modo que, ao ser percebida movimentação no ambiente monitorado o sistema seja capaz de notificar rapidamente o usuário através do envio de mensagem SMS e o acionamento de um alarme sonoro. Caso não haja intervenção por parte do usuário no sentido de desativar o sistema, este procederá com o acionamento do alarme sonoro. Há também os LED, que servem para indicar o *status* de alarme ativo ou não e quando há detecção de intruso o LED BUZZER é ativado indicando que o alarme sonoro será acionado em instantes.

As informações seguem para o servidor que então disponibilizará o *status* do alarme sonoro para a aplicação *web*, ficando a cargo do usuário realizar a captura de imagens ou devida desativação das funcionalidades enquanto o sistema opera no modo de detecção de intrusos. Após quarenta minutos desde a ativação do alarme sonoro, o sistema desativará o alarme automaticamente caso o usuário não o tenha realizado.

O protótipo pode ser instalado em qualquer ambiente ou mesmo no ambiente principal da residência e possibilita que este seja monitorado de qualquer lugar através da *Internet*.

6 ANÁLISES E RESULTADOS

Este capítulo apresenta os resultado de testes realizados para analisar o funcionamento do sistema projetado como um todo, verificar se este atende às especificações e objetivos propostos e produzir uma avaliação qualitativa sobre o seu funcionamento geral. Os ensaios foram conduzidos baseados nas respostas de cinco partes importantes que compõe o sistema, tais são o protótipo, a interface do usuário, o *streaming* de vídeo e o envio de SMS.

6.1 Protótipo

A Figura mostra o sistema de monitoramento em sua fase final. Os equipamentos foram acomodados em uma *case* de *Raspberry Pi* com a finalidade de posicionar o sensor e a câmera adequadamente, além de manter o conjunto organizado. A Figura apresenta a visão frontal do protótipo

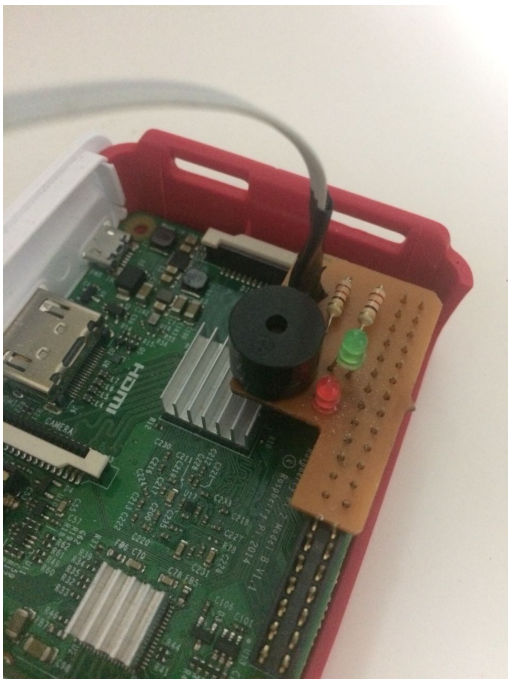
Figura 20- Visão frontal do protótipo do sistema de monitoramento e segurança.

Fonte: Produção do próprio autor.

A partir da figura é possível observar os LED de sinalização, a câmera e o sensor de presença PIR.

O circuito construído para colocação dos LED, *buzzer* e conexão do sensor de presença ao *Raspberry Pi* pode ser observado a seguir.

Figura 21- Imagem do circuito do protótipo final.



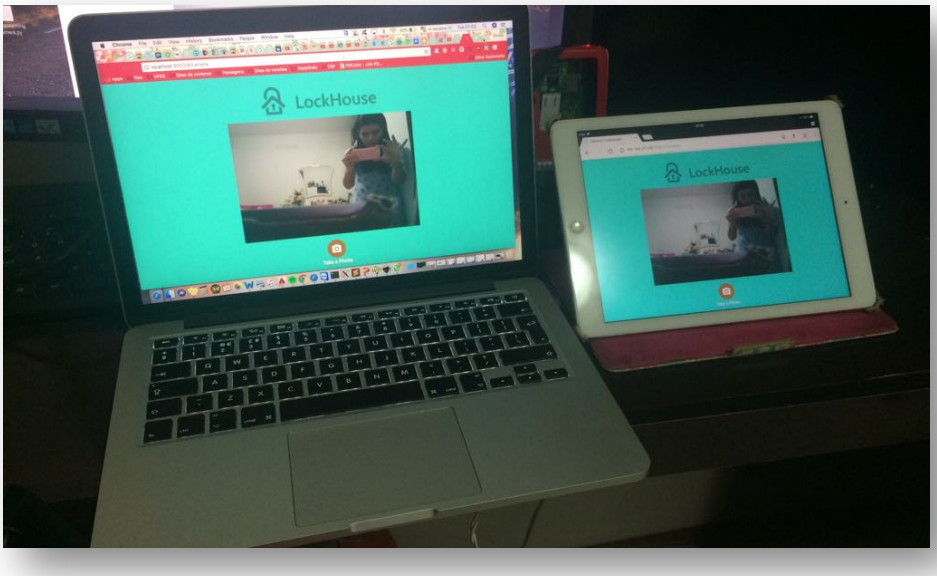
Fonte: Produção do próprio autor.

O teste do protótipo envolve a verificação do funcionamento do Raspberry, do sensor de presença, do buzzer e dos LED de sinalização. Para apuração dos resultados foi simulado uma intrusão para acionar o LED de sinalização do *Buzzer* , o envio de SMS e em seguida o ativação do *Buzzer*, indicando que o sensor de presença também havia funcionado. O resultado obtido foi satisfatório e o protótipo correspondeu às expectativas sobre seu funcionamento.

6.2 Interface

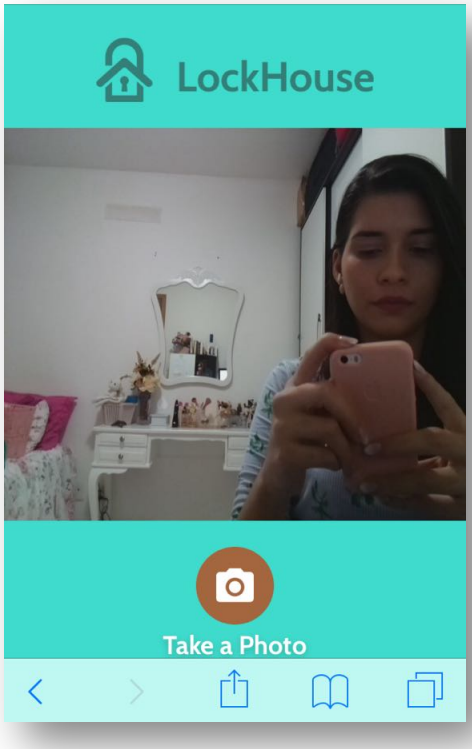
O primeiro teste realizado foi a averiguação sobre o funcionamento da interface quanto à responsividade, ou seja, que a aplicação possa ser acessada a partir de smartphones, tablets e computadores oferecendo sempre a mesma qualidade dos componentes da interface. A constatação foi concebida utilizando diversas plataformas de tecnologia, entre elas um *iPhone* do modelo 5S, um *iPad Air* e um *Macbook Pro Retina Display*, dos quais o resultado a seguir puderam ser obtidos.

Figura 22-Imagem da aplicação *web* sendo utilizada para monitoramento através do *Macbook* e *iPad*.



Fonte: Produção do próprio autor.

Figura 23- Imagem da tela do celular acessando a tela de monitoramento da aplicação.



Fonte: Produção do próprio autor.

Os resultados obtidos com respeito à responsividade das telas foi satisfatório, em particular a tela de monitoramento por câmera, pois como pode-se observar a partir das imagens das Figuras 22 e 23 a interface se adaptou à tela de cada dispositivo sem perder a qualidade da visualização do *streaming* e nem dos elementos UI.

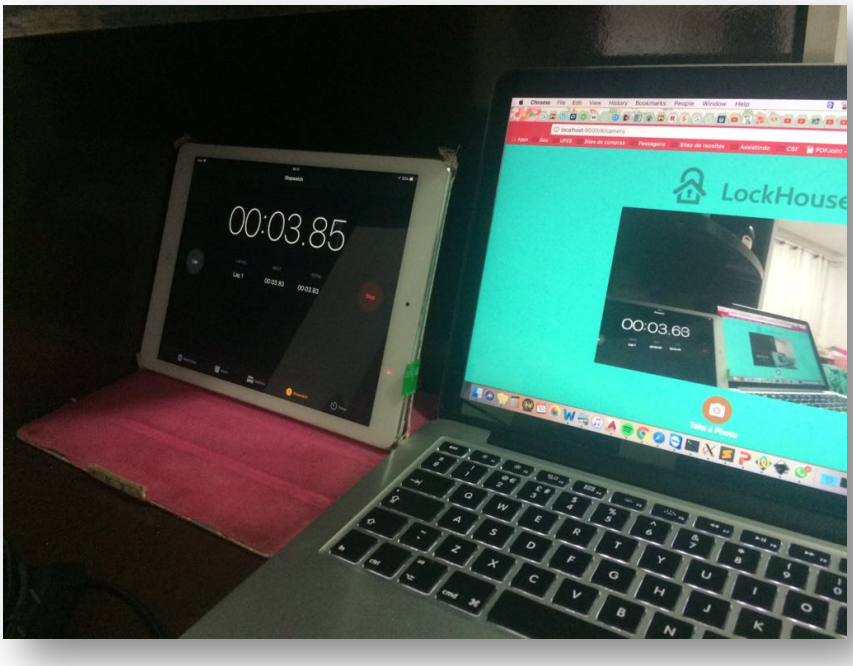
A validação da interface em navegadores diferentes também foi realizada, checando a compatibilidade da aplicação com diferentes navegadores. O resultados obtidos apontam que não houve problemas de compatibilidade entre o navegador e a interface, além disso, um parâmetro interessante foi observado: a latência do *streaming* de vídeo variou de acordo com o *browser* empregado. Os testes foram realizados nos navegadores Safari, Google Chrome e *Chromiun* do *Raspberry Pi* e, ainda que a qualidade do monitoramento fosse satisfatória, o navegador Safari apresentou resposta ao *streaming* inferior aos outros navegadores testados.

6.3 *Streaming* de vídeo

A primeira validação do *streaming* efetuada foi um teste de *stress* de *streaming* de vídeo, onde o mesmo permaneceu ligado por uma semana, período no qual foram realizados vários acessos ao sistema. Os resultados obtidos foram positivos pois não houve interrupção na transmissão durante esse tempo, no entanto, verificou-se que o *streaming* é afetado pela velocidade de conexão com a *Internet* o que fez com que a qualidade do *streaming* ocasionalmente oscilasse.

O testes de *streaming* constataram que a latência média do *streaming* no *browser* foi de meio segundo. Este foi realizado direcionando o sistemas de monitoramento para um contador e comparando o tempo real do contador com o tempo que a aplicação mostrava na imagem do contador vista pela interface. A Figura 24 apresenta o procedimento de teste.

Figura 24-Procedimento de teste para determinação do tempo médio de latência do *streaming* de vídeo.



Fonte: Produção do próprio autor.

Outro teste realizado foi quanto a resposta do *streaming* de vídeo a diferentes níveis de luminosidade. A imagem a seguir apresenta os resultados do teste.

Figura 25- Variação da imagem do *streaming* conforme a luminosidade.

É possível observar a partir da Figura 25 que quando a luminosidade é média ou baixa ainda é possível observar os objetos, entretanto na ausência de luminosidade ficaria inviável perceber a presença de intrusos. No entanto, o teste de baixa luminosidade demonstra que o ambiente

não necessita estar totalmente iluminado para reproduzir um *streaming* capaz de observar intrusos. Para tal comprovação foi acionado a lanterna do celular longe da câmera e obteve-se resultados satisfatórios. A imagem pode ser observada na Figura 26.

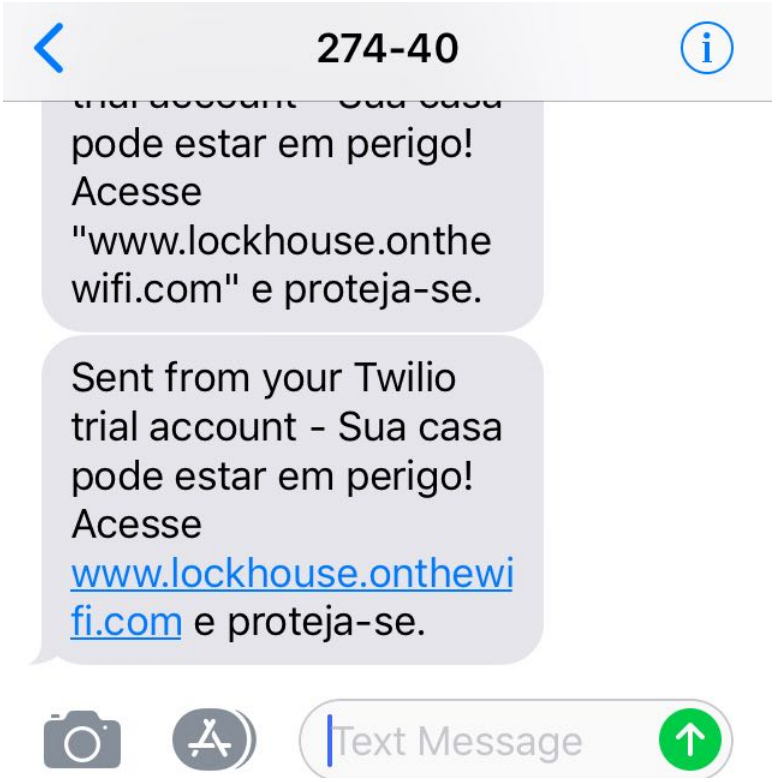
Figura 26- *Streaming* de vídeo na ausência de luz natural mas com pequeno foco de luz acionado.

Portanto, se faz necessário que o usuário acione qualquer objeto luminoso, por menor que seja, auxiliará o *streaming* de imagens a ter uma boa qualidade na ausência de luz.

6.4 Envio de SMS

Ao realizar a avaliação sobre o funcionamento no envio de SMS foi constatado o início de um evento de percepção de presença com o alarme acionado e o tempo de resposta até o recebimento da mensagem no celular. O tempo estimado foi em torno de 7 segundos. O que leva a crer que a resposta é rápida, cumprindo um importante papel entre as funcionalidades do sistema. A mensagem recebida pelo usuário pode ser visualizada na Figura 27.

Figura 27-Verificação sobre o funcionamento de notificação por mensagem SMS.



Fonte: Produção do próprio autor.

7 CONCLUSÕES

O principal objetivo deste trabalho é apresentar um protótipo de sistema de automação residencial construindo um sistema de segurança e monitoramento baseado no uso da Internet das Coisas (IoT), foi realizado através da construção do protótipo e do desenvolvimento da interface do usuário (UI) e obteve sucesso em seus resultados, assim como todas as premissas foram satisfeitas. O sistema foi elaborado através de levantamento de requisitos e estudo da estrutura a ser desenvolvida e análise detalhada do mesmo, que foi desdobrado em componentes físicos e de *software* do projeto, utilizando o entendimento sobre automação residencial e controle automático baseado nos conceitos da IoT e o conhecimento sobre eletrônica e composição de elementos de *software* o sistema foi implementado.

O desenvolvimento de um sistema de automação residencial integrado à uma interface *Web* promove o aumento de mecanismos de segurança para o patrimônio residencial. Com isso a ferramenta construída possibilita o acesso do usuário à informação sobre sua residência através do monitoramento e acionamento de alarme o que propiciou melhor controle sobre as condições de segurança do ambiente doméstico. Com o projeto e desenvolvimento da ferramenta *web* obtêm-se um sistema completo que age de forma inteligente tomando ações de acordo com a realidade do ambiente e a lógica estabelecida.

Através da realização dos testes foi possível constatar que o desenvolvimento do sistema de monitoramento residencial planejado obteve resultados satisfatórios, no entanto se faz necessário que uma fonte de luz, por mais que pequena, seja utilizada para visualização do *streaming* na ausência de luz.

Cabe aqui ressaltar que este trabalho poderá ser expandido, englobando a conexão com novos dispositivos para ambientes domésticos e evoluindo a interface com o usuário, criando novas funcionalidades. Para implementações futuras poderá ser avaliada a possibilidade de integração do sistema com os canais de emergência e também desenvolver outros mecanismos de armazenamento e apresentação das imagens capturadas pelo protótipo.

Por fim, conclui-se que o sistema desenvolvido mostrou-se capaz de reduzir o tempo de ciência do usuário a uma invasão domiciliar. O uso do sistema de segurança residencial tornará a resposta ao atendimento do usuário mais eficiente e eficaz, gerando assim maior segurança, satisfação, usabilidade e disponibilidade de acesso às informações do ambiente doméstico.

8 REFERÊNCIAS BIBLIOGRÁFICAS

WU, M. et al. **Research on the architecture of the Internet of things**. School of Economics and Management. Beijing, p. 484-487. 2010. (V5).

WILLIAMS, W. **Beta News**, 2013. Disponivel em: <https://betanews.com/2013/05/14/raspberry-pis-camera-board-now-available/>. Acesso em: 2018.

WORTMEYER, C.; FREITAS, F.; CARDOSO, L. **Automação Residencial: Busca de Tecnologias visando o Conforto, a Economia, a Praticidade e a Segurança do Usuário.**. Associação Educacional Dom Bosco – AEDB. Resende , p. 1064-1067. 2005.

XU, K. et al. Toward Software Defined Smart Home. **Green Communications and Computing Networks**, p. 116-122, Maio 2016.

AURESIDE. - Associação Brasileira de Automação Residencial e Predial. Todos os direitos reservados. **AURESIDE**, 2015. Disponivel em: <http://www.aureside.org.br/aureside-imprensa>. Acesso em: 2018.

ALLAMEH, E. et al. The Role of Smart Home in Smart Real Estate. **Journal of European Real Estate Research**, p. 156-170, 2012.

ANDRADE, A. **DM/Cotidiano**, 2017. Disponivel em: <https://www.dm.com.br/cotidiano/2017/01/cresce-numero-de-furtos-e-roubos-a-residencias-em-goiania.html>. Acesso em: 2018.

BRUCKNER, D. Building Automation, Amart Homes, and communities- Guest Editorial. **IEE transactions on industrial informatics**, v. 10, n. 1, p. 1-1, 2013.

CARVALHO, M. A. C.; TOMAZELA, J. M. T. A cada hora, uma casa é invadida em SP; o número de roubos a imóveis é o maior em 3 anos. **Uol notícias**, 2017. Disponivel em: <https://noticias.uol.com.br/ultimas-noticias/agencia-estado/2017/06/25/sp-tem-maior-n-de-ataques-a-residencias-em-3-anos.htm>. Acesso em: 2018.

CARDOSO, L. H. M. **PROPOSTA DE UM SISTEMA DE MONITORAMENTO RESIDENCIAL INTEGRADO AOS SERVIÇOS PÚBLICOS DE EMERGÊNCIA**. Universidade Federal de Santa Catarina. Araranguá. 2015.

EVANS, D. **Cisco**, Abril 2011. Disponivel em: <https://www.cisco.com/c/dam/en_us/about/ac79/docs/innov/IoT_IBSG_0411FINAL.pdf>. Acesso em: 2018.

EVER PI. **Como conectar o sensor PIR HC-SR501 ao Raspberry Pi**. Ever Pi. [S.l.]. 2015.
DEPHACE. **DEPHACE**, 2017. Disponivel em: <https://dephace.com/raspberry-pi-format-sd-card/raspberry-pi-format-sd-card-featured-image/>. Acesso em: Junho 2018.

DINO. Invasões a residências aumentam 15% durante Carnaval. **Exame**, 2018. Disponivel em: <https://exame.abril.com.br/negocios/dino/invasoes-a-residencias-aumentam-15-durante-carnaval/>. Acesso em: 2018.

FOLHA DIÁRIA. Allianz dá Dicas Para Evitar Roubos e Furtos Durante as Férias. **Folha Diária**, 2017. Disponivel em: <http://www.folhadiaria.com.br/materia/64/2865/l/p/allianz-da-dicas-para-evitar-roubos-e-furtos-durante-as-ferias#.W0Txji-ZNPN>. Acesso em: 2018.

GUBBI, J. Internet of things (IoT): A vision, architectural elements, and future directions. **Future generation computer systems**, n. Julho, p. 1645-1660, 01 Setembro 2013.

ISIKDAG, U. **Enhanced Building Information Models**. [S.l.]: Springer International Publishing, 2015.

ITU. ITU. **ITU Internet Report**, 2005. Disponivel em: <http://www.itu.int/osg/spu/publications/internetofthings/>. Acesso em: Junho 2018.

HOU, J. et al. **Research of Intelligent Home Security Surveillance System Based on ZigBee**. College of Information Science & Engineering. Liaoning , p. 554-557. 2008.

JOSE, A. C. et al. Improving Home Automation Security; Integrating Device Fingerprinting Into Smart Home. **IEEE Access**, p. 5777-5778, Setembro 2016.

LEONEL, J. S. **Twilio**, 2015. Disponivel em: <https://dicastwilio.wordpress.com/2015/01/24/o-que-eh-o-twilio/>. Acesso em: Julho 2018.
LISBOA, E. F.; CRUZ, A. A. WebHome – Automação residencial utilizando Raspberry PI.

Revista Ciência e Tecnologia , v. 17, n. 31, p. 35-43, Dezembro 2014. ISSN 2236-6733.
MURATORI, J. R.; BÓ, P. H. D. **Automação residencial:** histórico, definições e conceitos. [S.l.]: Atitude Editorial. 2005. p. 70-77.

MELO, M. **Metrópoles**, 2017. Disponivel em: <https://www.metropoles.com/distrito-federal/seguranca-df/brasilia-registra-tres-casos-de-roubos-a-residencias-por-dia>. Acesso em: 2018.

MILLENIUM, I. ROUBOS A RESIDÊNCIAS AUMENTAM 70% NO RIO, E MORADORES INVESTEM EM SEGURANÇA. **Instituto Millenium**, 2018. Disponivel em: <https://www.institutomillenium.org.br/blog/roubos-residencias-aumentam-70-no-rio-e-moradores-investem-em-seguranca/>. Acesso em: 2018.

MINERVA, R.; BIRU, A.; ROTONDI, D. Internet of Things. **IEEE**, 2015. Acesso em: 2018.
OGATA, K. **Modern Control Engineering**. 3. ed. Minnesota: [s.n.], 1997.

OLIVEIRA, T. R. D.; GIGLIO, G. P. D. M. **Análise de estudo de casos em abordagens pelo mundo da implementação de Internet das Coisas**. Centro de Ensino Superior de Juiz de Fora (CES/JF). Minas Gerais. 2017.

PEREIRA, L. A. D. M. **Automação Residencial: rumo a um futuro pleno de novas soluções**. Pontifícia Universidade Católica do Rio de Janeiro – PUC-Rio. Rio de Janeiro. 2008.

PINZ, L. Você está preparado para a Internet das Coisas? **ITFORUM 365**, Setembro 2014. Disponivel em: <http://itforum365.com.br/noticias/detalhe/113954/voce-esta-preparado-para-a-internet-das-coisas>. Acesso em: Novembro 2016.

SANTOS, A. D. S.; KASPARY, B. Estatística revela menos assassinatos e mais roubos a residências em MS. **Campo Grande News**, 2018. Disponivel em: <https://www.campograndenews.com.br/cidades/estatistica-revela-menos-assassinatos-e-mais-roubos-a-residencias-em-ms>. Acesso em: 2018.

SANTOS, F. C. L. **Jusbrasil**, 2014. Disponivel em: <https://fernandacls.jusbrasil.com.br/artigos/147323303/as-causas-de-aumento-dos-crimes-patrimoniais>. Acesso em: Junho 2018.

SILVA, B. R. S. D. **Sistema de automação residencial de baixo custo para redes sem fio**. UNIVERSIDADE FEDERAL DO RIO GRANDE DO SUL. Porto Alegre. 2014.

RASPBERRY PI FOUNDATION. Disponivel em: <https://www.raspberrypi.org/help/faqs/#topIntro>. Acesso em: 5 Julho 2018.

ROCHA, R. **Reengineering and development of IoT Systems for Home Automation**. CISTER Research Centre Polytechnic Institute of Porto. Porto. 2017.

TALGERI, A.; ABHEESH, K. B. A.; B, A. Domotics- A Cost Effective Smart Home Automation System. **Journal of Engineering Research and applications**, p. 52-55, 2014.

TEZA, V. R. **ALGUNS ASPECTOS SOBRE A AUTOMAÇÃO RESIDENCIAL - DOMÓTICA**. UNIVERSIDADE FEDERAL DE SANTA CATARINA. Florianópolis. 2002.

APÊNDICE A

A.1-Códigos Fonte- Protótipo

Streaming de Câmera

```
##
import io
import picamera
import logging
import socketserver
from threading import Condition
from http import server

PAGE="""\
<html>
<head>
<title>Raspberry Pi - Surveillance Camera</title>
</head>
<body>
<center><h1>Raspberry Pi - Surveillance Camera</h1></center>
<center><img src="stream.mjpg" width="640" height="480"></center>
</body>
</html>
"""

class StreamingOutput(object):
    def __init__(self):
        self.frame = None
        self.buffer = io.BytesIO()
        self.condition = Condition()

    def write(self, buf):
        if buf.startswith(b'\xff\xd8'):
            # New frame, copy the existing buffer's content and notify all
            # clients it's available
            self.buffer.truncate()
            with self.condition:
                self.frame = self.buffer.getvalue()
                self.condition.notify_all()
            self.buffer.seek(0)
        return self.buffer.write(buf)

class StreamingHandler(server.BaseHTTPRequestHandler):
    def do_GET(self):
        if self.path == '/':
            self.send_response(301)
            self.send_header('Location', '/index.html')
```

```
            self.end_headers()
        elif self.path == '/index.html':
            content = PAGE.encode('utf-8')
            self.send_response(200)
            self.send_header('Content-Type', 'text/html')
            self.send_header('Content-Length', len(content))
            self.end_headers()
            self.wfile.write(content)
        elif self.path == '/stream.mjpg':
            self.send_response(200)
            self.send_header('Age', 0)
            self.send_header('Cache-Control', 'no-cache, private')
            self.send_header('Pragma', 'no-cache')
            self.send_header('Content-Type', 'multipart/x-mixed-replace; boundary=FRAME')
            self.end_headers()
            try:
                while True:
                    with output.condition:
                        output.condition.wait()
                        frame = output.frame
                    self.wfile.write(b'--FRAME\r\n')
                    self.send_header('Content-Type', 'image/jpeg')
                    self.send_header('Content-Length', len(frame))
                    self.end_headers()
                    self.wfile.write(frame)
                    self.wfile.write(b'\r\n')
            except Exception as e:
                logging.warning(
                    'Removed streaming client %s: %s',
                    self.client_address, str(e))
        else:
            self.send_error(404)
            self.end_headers()

class StreamingServer(socketserver.ThreadingMixIn, server.HTTPServer):
    allow_reuse_address = True
    daemon_threads = True

with picamera.PiCamera(resolution='640x480', framerate=24) as camera:
    output = StreamingOutput()
    #Uncomment the next line to change your Pi's Camera rotation (in degrees)
    #camera.rotation = 180
    camera.start_recording(output, format='mjpeg')
    #camera.wait_recording(10)
    #for i in range(5):
    #   camera.capture('Pictures/picture%s.jpg' %i, use_video_port=True)
    try:
        address = ('', 8000)
        server = StreamingServer(address, StreamingHandler)
```

```
        server.serve_forever()
    finally:
        camera.stop_recording()
```

Controle de GPIO

```
<?php

error_reporting(E_ALL);
ini_set("display_errors", 1);

require 'vendor/autoload.php';

use \Psr\Http\Message\ServerRequestInterface as Request;
use \Psr\Http\Message\ResponseInterface as Response;
use PiPHP\GPIO\GPIO;
use PiPHP\GPIO\Pin\InputPinInterface;
use PiPHP\GPIO\Pin\OutputPinInterface;



$config['displayErrorDetails'] = true;
$config['addContentLengthHeader'] = false;
$config['db'] = require('./src/db/db-config.php');

$app = new \Slim\App(["settings" => $config]);

# container Slim
$container = $app->getContainer();

$container['db'] = function ($c) {
        $DC = new DataConnection($c['settings']['db']);
        return new DataManagement($DC::$connection);
};
```

```
$app->options('/{routes:.+}', function ($request, $response, $args) {
   return $response;
});


#CORS
$app->add(function ($req, $res, $next) {
   $response = $next($req, $res);
   return $response
      ->withHeader('Access-Control-Allow-Origin', '*')
      ->withHeader('Access-Control-Allow-Headers',  'X-Requested-With,  Content-Type,
Accept, Origin, Authorization')
      ->withHeader('Access-Control-Allow-Methods',  'GET,  POST,  PUT,  DELETE,
OPTIONS');
});

#ROUTES

$app->get('/', function ($request,  $response, $args = []) {
    $this->db;
    return $response->withStatus(200)->write('<h1>We are ready!</h1>');
});

$app->post('/camera', function($request, $response, $args) {
    $body = $request->getParsedBody();
    return $response->withJson($this->db->addPicture($body));
});

$app->post('/alarm', function($request, $response, $args) {
    $body = $request->getParsedBody();
    return $response->withJson($this->db->addAlarm($body));

});
```

```
// led-blink.php

require_once 'vendor/autoload.php';

//Funções de GPIO

// This GPIO object can be used to retrieve pins and create interrupt watchers
$gpio = new GPIO();


$ledPinGreen = $gpio->getOutputPin(27);
$ledPinRed = $gpio->getOutputPin(17);
$buzzerPin = $gpio->getOutputPin(23);

// Configure pin 3 as an input pin and retrieve an object that we can use to observe it
$pirPin = $gpio->getInputPin(4);

// Configure this pin to trigger interrupts when the voltage rises.
// ::EDGE_FALLING and ::EDGE_BOTH are also valid.


$ledPinGreen->setValue(OutputPinInterface::VALUE_HIGH);
$pirPin->setEdge(InputPinInterface::EDGE_RISING);


// Create an interrupt watcher (this is a type of event loop)
$interruptWatcher = $gpio->createWatcher();

// Register a callback for handling interrupts on the button pin
$interruptWatcher->register($pirPin, function ($pirPin, $value) use ($ledPinRed,$buzzerPin)
{
```

```
    //Envio SMS

        require ('sms.php');

    echo 'Blinking LED...' . PHP_EOL;
    for ($i = 0; $i < 20; $i++) {
                        $ledPinRed->setValue(OutputPinInterface::VALUE_HIGH);
                        usleep(100000);
                        $ledPinRed->setValue(OutputPinInterface::VALUE_LOW);
                        usleep(100000);
        }
        sleep(15);

    while (true){
                        $buzzerPin->setValue(OutputPinInterface::VALUE_HIGH);
                        $ledPinRed->setValue(OutputPinInterface::VALUE_HIGH);
                        usleep(200000);
                        $buzzerPin->setValue(OutputPinInterface::VALUE_LOW);
                        $ledPinRed->setValue(OutputPinInterface::VALUE_LOW);
                        usleep(200000);
        }


    // Returning false would cause the loop below to exit
    return true;
});

while(true){
        $interruptWatcher->watch(20);
}
```

Código para envio de SMS

```
<?php
require __DIR__ . '/vendor/autoload.php';
use Twilio\Rest\Client;

// Your Account SID and Auth Token from twilio.com/console
$account_sid = $_ENV["TWILIO_ACCOUNT_SID"]
$auth_token = $_ENV["TWILIO_ACCOUNT_TOKEN"]
// In production, these should be environment variables. E.g.:

// A Twilio number you own with SMS capabilities
$twilio_number = "+xxxxxxxxxxx”
$client = new Client($account_sid, $auth_token);

$client->messages->create(
   // Where to send a text message (your cell phone?)
   '+5527xxxxxxxxx',
   array(
      'from' => $twilio_number,
      'body' => 'Sua casa pode estar em perigo! Acesse LockHouse! '
   )
);
```

A.2-Códigos Fonte- Interface

Inicial

```
<template>
  <section class="activate" if.bind="enableScreen">
    <div class="active">
      <div>
```

```
      <img src="/src/styles/assets/check.png"></img>
    </div>
    <span>The alarm is <b>ON</b></span>
  </div>
</section>
<section class="home">
  <img src="/src/styles/assets/logo_header.png" alt="LockHouse"
class="logo">
  <div>
    <button
    if.bind="!lock"
    class="lockoff"
    click.delegate="check()">
      <span>Press to Lock</span>
    </button>
    <button
    if.bind="lock"
    class="lockon"
    click.delegate="check()">
      <span>Press to Unlock</span>
    </button>
    <button
    class="view"
    click.delegate="goScreen('camera')">
      <span>Video</span>
    </button>
    <button
    class="alarm"
```

```
    click.delegate="goScreen('alarm')">
      <span>Alarm</span>
    </button>
  </div>
 </section>
</template>

export class Inicial {

        constructor() {
                this.lock = false;
                this.enableScreen = false;
                this.count = 1;
        }

        activate(params, routeConfig, navigationInstruction) {
                this.params = params;
                this.router = navigationInstruction.router;
        }

        check(){
                if(!this.lock){
                        this.contagemRegressiva();
                }
                else{
                        this.goScreen('panel');
                }
        }
```

```
    goScreen(screen){
        this.router.navigateToRoute(screen);
    }

    contagemRegressiva() {
        this.enableScreen = true;
        setTimeout(() => {
        this.count--;
            // if (!/home/.test(document.location.hash)) {
                if (this.count <= 0) {
                    this.enableScreen = false;
                    this.lock = true;
                } else {
                    this.contagemRegressiva();
                }
            // }
        }, 1000);
    }
}
```

Câmera

```
<template>
    <section class="view">
        <img    src="/src/styles/assets/logo_header.png"    alt="LockHouse"
class="logo">
        <img                class                =                "stream"
src="http://192.168.25.177:8000/stream.mjpg"/>
        <div>
```

```
                <button class="foto" click.delegate ="takePicture()">
                        <span>Take a Photo</span>
                </button>
            </div>
        </section>
</template>
```

Alarme

```
<template>
        <section class="sirene">
                <img                src="/src/styles/assets/logo_header.png"
alt="LockHouse" class="logo">
                <div class="status">
                        <div class="phrase">${phrase}</div>
                        <div class="display">
                                <div class="on" if.bind = "buzzer && !counting"
id="on">${written}</div>
                                <div class="off" if.bind = "!buzzer &&
!counting">${written}</div>
                                <output if.bind = "counting" class="count
pulse">
                                        <div class="number">${display}</div>
                                </output>
                        </div>
                        <button class="disableBuzzer" class.bind="alarm? " :
'buttonOff' " click.delegate ="turnOffAlarm()" id="disable" >Disable</button>
                </div>
        </section>
```

```
</template>


export class Alarm {

	constructor() {
		this.display = '10'; //Pegar do BackEND
		this.counting = true; //Gerar do display
		this.alarm = true; //Pegar do BackEND -quando alarm e presenca são
detectados
		this.buzzer= false; //Valor inicial
	}
	activate(params, routeConfig, navigationInstruction) {
		this.params = params;
		this.router = navigationInstruction.router;
		if(this.counting){
			this.phrase = 'The Sirene Will Start in...';
			this.contagemRegressiva();
		}
		else{
			if(this.alarm){
				this.buzzer= true;
			}
			this.buzzerOnOff();
		}
	}
	turnOffAlarm(){
		this.buzzer = false; //Mandar pro BackEND interrupção
		this.counting = false;
		document.getElementById("disable").style.opacity = "0.6";
		document.getElementById('disable').disabled = true;
		this.buzzerOnOff();
		this.router.navigateToRoute('panel');
```

```
        }
        buzzerOnOff(){
                this.phrase = 'The Sirene is:';
                if(this.buzzer){
                        this.written = 'ON';
                }else{
                        this.written = 'OFF';
                }
        }
        contagemRegressiva() {
                setTimeout(() => {
                        this.display--;
                                if (this.display < 0) {
                                        if(this.counting){
                                                this.buzzer = true;
                                        }
                                        this.counting = false;
                                        this.buzzerOnOff();
                                } else {
                                        this.contagemRegressiva();
                                }
                }, 1000);
        }
}
```

Desarmar Alarme

```
<template>
  <section class="panel">
    <img src="/src/styles/assets/logo_header.png" alt="LockHouse" class="logo">
    <form>
      <div class="display">${ number }</div>
      <!-- //Limit size string
      //enable button
```

```
        //Create span
        //password protection -->
        <div class="keyboard">
            <button type="button" click.delegate = "addNumber('1')" >1</button>
            <button type="button" click.delegate = "addNumber('2')" >2</button>
            <button type="button" click.delegate = "addNumber('3')" >3</button>
            <button type="button" click.delegate = "addNumber('4')" >4</button>
            <button type="button" click.delegate = "addNumber('5')" >5</button>
            <button type="button" click.delegate = "addNumber('6')" >6</button>
            <button type="button" click.delegate = "addNumber('7')" >7</button>
            <button type="button" click.delegate = "addNumber('8')" > 8</button>
            <button type="button" click.delegate = "addNumber('9')" >9</button>
            <button type="button" click.delegate = "clearArray()" >Cancel</button>
            <button type="button" click.delegate = "addNumber('0')" >0</button>
            <button type="button" click.delegate = "clearLast()">Clear</button>
        </div>
        <button class="check" type="button" click.delegate = "checkPassword()"> Disable
Alarm</button>
        </form>
    </section>
</template>

import { inject } from 'aurelia-framework';
import { HttpClient} from 'aurelia-fetch-client';

@inject(HttpClient)
export class WebApi {



getVerboseUrl(url){

        return this.client.fetch(url)
            .then(loaded => {
```

```
            if (!loaded.success) {
                console.warn("Nenhum dado foi obtido de " + url + "\n" + loaded.data);
            }
        }).catch(error => {
            console.error(error);
        }
    );
  }
getUrl(url) {
    return this.getVerboseUrl(url, {method: "GET"});
  }

}
```

Código Main

```
// import {inject} from 'aurelia-framework';
// import {Dados} from './dados';
// import {WebApi} from './web-api';

// @inject (WebApi, Dados)
export class App {

        constructor(api, dados) {
                this.api = api;
                // this.config = window.productConfig;
        }
        configureRouter(config, router){
            config.title = 'LockHouse';
            config.map([
                        {
                                route: ['', 'inicial'],
                                moduleId: 'inicial',
```

```
                title: 'LockHouse',
                name: 'inicial'
            },
            {
                route: ['camera'],
                moduleId: 'camera',
                title: 'Camera',
                name: 'camera'
            },
            {
                route: ['alarm'],
                moduleId: 'alarm',
                title: 'Alarm',
                name: 'alarm'
            },
            {
                route: ['panel'],
                moduleId: 'panel',
                title: 'Panel',
                name: 'panel'
            }
        ]);
            config.mapUnknownRoutes(instruction => {
    return 'inicial';
    });
    }
}
```

Folha de estilos

```
@keyframes pulse {
    0%{
```

```
        background-color: rgba(255, 24, 56, 0.7);
        }
        100%{
        background-color: rgba(36, 181, 150, 0.8);
        }
}
*, ::before, ::after {
        font-family: 'Cabin', sans-serif;
        box-sizing: border-box;
}
body, html, section {
        margin: 0;
        height: 100%;
        background-color: #3fdbcc;
        font-size: 18px;
        color: rgba(43, 43, 43, 0.7);
}
section {
        display: flex;
        flex-flow: column;
        align-items: center;
        justify-content: center;
        position: relative;
}
a {
        color: #0e8379;
}

::-webkit-input-placeholder { color: #52a7a0; }
::-moz-placeholder { color: #52a7a0; }
:-ms-input-placeholder { color: #52a7a0; }
:-moz-placeholder { color: #52a7a0; }
```

```
.logo {
        width: 60%;
        max-width: 20em;
        opacity: 0.75;
        margin: 2em 0 1em 0;
}
.login .logo {
        width: 55%;
        margin: 0 0 2em 0;
        position: static;
        opacity: 1;
}
.login form {
        width: 55%;
        padding: 0;
        display: flex;
        flex-flow: column;
}
input {
        padding: 0.5em;
        font-size: 1.4em;
        text-align: center;
        margin: 0.2em 0;
        border: none;
        border-radius: 0.25em;
        background-color: #fff;
        opacity: 0.8;
        color: #007d73;
}
input:focus {
        opacity: 1;
        outline: none;
}
```

```
.login button {
        padding: 0.5em;
        font-size: 1.4em;
        text-align: center;
        margin: 0.85em 0 1.3em 0;
        border: none;
        border-radius: 0.25em;
        background-color: #0e8379;
        opacity: 0.8;
        color: #fff;
        transition: opacity 1s;
        height: auto;
        width: auto;
}
button:disabled {
        opacity: 0.3;
        transition: opacity 1s;
}
.login a {
        font-size: 1.1em;
        text-decoration: none;
}
button {
        width: 5em;
        height: 5em;
        border: none;
        border-radius: 50%;
        position: relative;
        font-size: inherit;
        background: transparent none center no-repeat / contain;
}
button span {
        color: #fff;
```

```
        font-weight: bold;
        position: absolute;
        white-space: nowrap;
        width: 120%;
        left: -10%;
        right: -10%;
        top: 105%;
        text-shadow: 0 0.15em 0.3em rgba(0, 0, 0, 0.2);
}
.activate{
        position: absolute;
        width: 100%;
        height: 100%;
        background-color:rgba(36, 181, 150, 0.6);
        z-index: 1;
}
.active{
        display: flex;
        flex-flow: column;
        align-items: center;
        justify-content: center;
}
.active div{
        border-radius: 50%;
        width: 8em;
        height: 8em;
        display: flex;
        align-items: center;
        justify-content: center;
        padding: 0;
        margin-bottom: 0.7em;
        background-color:rgba(36, 181, 150, 1);
}
```

```
.active img{
        width: 4em;
        height: 4em;
}
.active span{
        font-size: 1.6em;
        color: rgba(0, 0, 0, 1);
        text-shadow: 0 0.15em 0.3em rgba(0, 0, 0, 1);
}
.home div {
        display: grid;
        grid-template-columns: 1fr 1fr;
        grid-template-rows: 3fr 2fr;
        grid-gap: 2em 3em;
        align-items: center;
        justify-content: center;
        flex: 1;
        margin: 0 0 6em 0;
}
button.lockoff {
        grid-column: 1 / 3;
        width: 7em;
        height: 7em;
        justify-self: center;
        background-image: url('/src/styles/assets/locked.png');
}
button.lockon {
        grid-column: 1 / 3;
        width: 7em;
        height: 7em;
        justify-self: center;
        background-image: url('/src/styles/assets/unlocked.png');
```

```
}
button.view {
	background-image: url('/src/styles/assets/view.png');
}
button.alarm {
	background-image: url('/src/styles/assets/alarm.png');
}
button.foto {
	background-image: url('/src/styles/assets/foto.png');
}
.stream {
	background-color: #000;
	height: 60%;
	width: 100%;
	max-width: 580px;
	max-height: 480px;
	border-radius: 0.8em;
}
.view div {
	display: flex;
	align-items: center;
	flex-flow: column;
	justify-content: center;
	margin: 1em 0 2em 0;
	height: 10%;
	min-height: 3em;
}
.status {
	/*line-height: 1.3;*/
	flex: 1;
	display: flex;
	flex-flow: column;
	justify-content: space-around;
```

```
        align-items: center;
        font-weight: bold;
}
.phrase{
        font-size: 1.4em;
}
.on,.off,output{
        border-radius: 50%;
        width: 5em;
        height: 5em;
        display: flex;
        align-items: center;
        justify-content: center;
        padding: 0;
        margin:0;
        color:#fff;
        font-size: 1.5em;
        font-weight: bold;
}
.on{
        background-color: rgba(255, 24, 56, 0.7);
}
.off{
        background-color:rgb(36, 181, 150);
}
/*output {
        border-radius: 50%;
        width: 6em;
        height: 6em;
        display: flex;
        align-items: center;
        justify-content: center;
        padding: 0;
```

```
        margin:0;
}*/
.pulse {
        animation: pulse infinite 1s;
        margin:0;
}
.disableBuzzer{
        color: rgba(255, 255, 255, 0.8);
        font-size: 1.4em;
        width: 8em;
        height: 2.6em;
        font-weight: bold;
        border-radius: 0.4em;
        box-shadow: 0 8px 16px 0 rgba(0,0,0,0.2), 0 6px 20px 0 rgba(0,0,0,0.19);
        margin-bottom: 1em;
        background-color:rgba(112, 223, 6, 0.6);

}
.buttonOff{
        opacity: 0.6;
}
.keyboard {
        display: grid;
        grid-template-columns: 1fr 1fr 1fr;
        grid-gap: 0.5em;
        margin: 1em 0;
        margin-bottom: 1em;
}
.panel {
        height: 100%;
        width: 100%;
        display: flex;
        flex-flow: column;
```

```
        align-items: center;
}
.panel form {
        display: flex;
        flex-flow: column;
        align-items: center;
        justify-content: space-around;
   padding: 0 2em;
   flex: 1;
}
.keyboard button {
        background-color: rgba(220,218,220,0.7);
        border-radius: 1em;
        font-size: 0.7em;
        color: rgb(54,126,119);
}
.panel .display {
        padding: 0.2em;
        width: 100%;
        height: 1.8em;
        background-color: rgba(225,225,225,1);
        font-size: 1.5em;
        text-align: center;
        border-radius: 0.35em;
        color: rgb(54,126,119);
}
.check {
        width: 80%;
        height: 3em;
        font-size: 1em;
        border-radius: 0.25em;
        background-color: rgba(201,48,91,1);
        margin-bottom: 1.5em;
```

```
}
/*.toggle {
        background-image: url('/src/styles/assets/off.png');
        width: 10em;
        border-radius: 0;
        height: 5em;
        transition: all 0.3s;
}
.toggle.on {
        background-image: url('/src/styles/assets/on.png');
        transition: all 0.3s;
}
.toggle.off {
        background-image: url('/src/styles/assets/off.png');
}
.action {
        flex: 2;
}*/
```