

UNIVERSIDADE FEDERAL DE SANTA MARIA
COLÉGIO TÉCNICO INDUSTRIAL DE SANTA MARIA
CURSO SUPERIOR DE TECNOLOGIA EM REDES DE
COMPUTADORES

Lucas Cardoso Viero

**SISTEMA DE AUTOMAÇÃO RESIDENCIAL UTILIZANDO
REDE WIFI COM ARDUINO E WEBSITE**

Santa Maria, RS
2016

Lucas Cardoso Viero

**SISTEMA DE AUTOMAÇÃO RESIDENCIAL UTILIZANDO REDE WIFI COM
ARDUINO E WEBSITE**

Trabalho de Conclusão de Curso (TCC)
do Curso Superior de Tecnologia em
Redes de Computadores, da
Universidade Federal de Santa Maria
(UFSM, RS), como requisito parcial para
obtenção do grau de **Tecnólogo em
Redes de Computadores.**

Orientador: Prof. Dr. Claiton Pereira Colvero

Santa Maria, RS
2016

Lucas Cardoso Viero

**SISTEMA DE AUTOMAÇÃO RESIDENCIAL UTILIZANDO REDE WIFI COM
ARDUINO E WEBSITE**

Trabalho de Conclusão de Curso (TCC)
do Curso Superior de Tecnologia em
Redes de Computadores, da
Universidade Federal de Santa Maria
(UFSM, RS), como requisito parcial para
obtenção do grau de **Tecnólogo em
Redes de Computadores.**

Aprovado em 5 de julho de 2016:

Claiton Pereira Colvero, Dr. (UFSM)
(Orientador)

Viviane Köhler, Dra. (UFSM)

Miguel Augusto Bauermann Brasil, Msc. (UFSM)

Santa Maria, RS
2016

DEDICATÓRIA

Dedico este trabalho a minha namorada Juliana Marchesan, que durante esses longos anos que estamos juntos, sempre me apoiou e incentivou desde quando comecei os estudos no curso até o momento de hoje. E que é uma pessoa muito importante na minha vida.

AGRADECIMENTOS

A minha família pelo apoio e suporte para que esse momento acontecesse.

Ao meu orientador, professor Dr. Claiton Pereira Colvero, pela confiança transmitida desde o começo do trabalho, pelas ideias, motivações e pelo seu tempo dedicado que independentemente do horário sempre esteve disponível para trocar informações.

Aos professores do curso que transmitiram seus conhecimentos durante as aulas.

Aos professores da banca que aceitaram ceder parte de seu tempo para avaliar este trabalho.

RESUMO

SISTEMA DE AUTOMAÇÃO RESIDENCIAL UTILIZANDO REDE WIFI COM ARDUINO E WEBSITE

AUTOR: Lucas Cardoso Viero
ORIENTADOR: Claiton Pereira Colvero

Neste trabalho foi desenvolvido um sistema de automação residencial de baixo custo que utiliza os próprios recursos da residência para a operação. Para a interação com o *hardware* da residência foi utilizado um módulo microprocessado Arduino® também de baixo custo e uma página *web* especialmente desenvolvida para o acesso do usuário através de seu dispositivo de comunicação *mobile* pessoal, em modo local. Como a interação com dispositivos da residência pode ser alvo de invasões não autorizadas e causar falhas de segurança física, para este projeto foram desenvolvidos diferentes níveis de credenciais de usuários para diferentes tipos de ações que podem ser realizadas no sistema. Em um nível hierárquico mais baixo, o usuário é automaticamente identificado ao entrar na rede de cobertura do *hotspot* disponibilizado na residência. Como esse artifício utiliza o endereço MAC da placa de rede do dispositivo para credencial de acesso, a interação com o sistema proporciona apenas o acendimento das luzes externas para avaliação da segurança do perímetro. Se o usuário estiver cadastrado no sistema, essa identificação permite que ele acesse o próximo nível de credencial de acesso depois de observar atentamente o ambiente iluminado no entorno da residência, onde através de uma senha criptografada é possível acessar outras funcionalidades mais invasivas, como acionar a iluminação interna, controlar portas e portões, eletrodomésticos, desligar o alarme, entre outros. As funcionalidades do sistema foram amplamente testadas após a implementação prática e os resultados corresponderam aos esperados no projeto original.

Palavras-chave: Automação Residencial. Arduino. Website. Wi-Fi. Segurança.

ABSTRACT

HOME AUTOMATION SYSTEM BASED ON NETWORK WIFI USING AN ARDUINO MODULE AND WEBSITE

AUTHOR: LUCAS CARDOSO VIERO
ADVISOR: CLAITON PEREIRA COLVERO

In this work we developed a home automation system that uses low-cost residence of own resources for the operation. For the interaction with the hardware of the residence was used a microprocessor module Arduino® low cost and also a web page specifically designed for user access through their personal mobile communication device in local mode. How the interaction with home devices may be subject to unauthorized intrusions and cause physical security holes, for this project have been developed different levels of user credentials for different types of actions that can be performed on the system. At a lower hierarchical level, the user is automatically identified when entering the hotspot network coverage available in the residence. As this device uses the MAC address of the device network card to access credential, the interaction with the system only provides the lighting of the external lights for evaluation of perimeter security. If the user is registered in the system, this identification allows it to access the next level of access credential after observing carefully lit environment surrounding the residence, where through an encrypted password is possible to access other more invasive features such as trigger indoor lighting, control gates and doors, appliances, turn off the alarm, among others. The system features have been amply tested after the practical implementation and the results corresponded to those expected in the original project.

Keywords: Home Automation. Arduino. Website. Wi-Fi. Security.

LISTA DE FIGURAS

Figura 1 – Parte da pilha de protocolos do 802.11	16
Figura 2 – Camadas (e protocolos) para usuários domésticos com SSL	23
Figura 3 – Modelos dos módulos Arduino® Uno R3	25
Figura 4 – Estrutura básica do relé de dois estados	27
Figura 5 – Definição dos diretórios do site	29
Figura 6 – Adicionando nome do site no arquivo hosts	29
Figura 7 – Configuração do HTTPS	30
Figura 8 – Configurando o HTTPS no Apache	30
Figura 9 – Modelo Entidade Relacionamento.....	31
Figura 10 – Diagrama Entidade Relacionamento.....	31
Figura 11 – Tabela do banco de dados.....	32
Figura 12 – Fluxograma do login.....	34
Figura 13 – Código de bloqueio de usuário e acionamento do alarme	34
Figura 14 – Fluxograma do painel de controle	35
Figura 15 – Código de envio de comandos para o Arduino®.....	36
Figura 16 – Fluxograma das configurações de cadastro.....	37
Figura 17 – Código da função excluir.....	37
Figura 18 – Código de senha fracas	38
Figura 19 – Código que verifica login iguais.....	39
Figura 20 – Código de campos preenchidos.....	39
Figura 21 – Fluxograma do cadastro.....	40
Figura 22 – Código que salva o login na SESSION	41
Figura 23 – Código para destruir a SESSION.....	41
Figura 24 – Criptografia da senha.....	42
Figura 25 – Funcionamento do hotspot.....	42
Figura 26 – Arquivo de configuração do <i>Ap-Hotspot</i>	43
Figura 27 – Script para ativar o Ap-Hotspot	43
Figura 28 – Arduino® Uno R3	44
Figura 29 – Código armazenado no módulo Arduino®	45
Figura 30 – Módulo de relés utilizado no projeto.....	46
Figura 31 – Script para verificação automática de um usuário.....	47
Figura 32 – Tela de login da interface web	49
Figura 33 – Tela de apresentação do painel de controle	50
Figura 34 – Configurações de usuário na página web	51
Figura 35 – Cadastro de um novo usuário	52
Figura 36 – Editar dados de um usuário	53
Figura 37 – Mensagem de alerta ao excluir um usuário cadastrado	53
Figura 38 – Senhas criptografadas no sistema	54
Figura 39 – Rede do hotspot.....	55
Figura 40 – Script para procurar endereço MAC.....	55
Figura 41 – Ensaios em laboratório com protoboard.....	56
Figura 42 – Ensaios finais com dispositivos elétricos de maior corrente.....	57
Figura 43 – Deformação da apresentação gráfica em telas diferentes	58
Figura 44 – Comando executado no reconhecimento do usuário	60
Figura 45 – Comando para habilitar porta serial no Linux.....	61

LISTA DE TABELAS

Tabela 1 – Desenvolvedores de servidores web: Mercado de sites ativos	19
Tabela 2 – Especificações do Arduino® Uno	26

LISTA DE ABREVIATURAS E SIGLAS

CERN	<i>Conseil Européen pour la Recherche Nucléaire</i>
CSMA/CA	<i>Carrier Sense Multiple Access with Collision Avoidance</i>
GNU-GPL	<i>General Public License</i>
HTML	<i>Hyper Text Markup Language</i>
HTTP	<i>Hyper Text Transfer Protocol</i>
HTTPS	<i>Hyper Text Transfer Protocol Secure</i>
IDE	<i>Integrated Development Environment</i>
IEEE	<i>Institute of Electrical and Electronics Engineers</i>
IETF	<i>Internet Engineering Task Force</i>
LLC	<i>Logical Link Control</i>
MAC	<i>Media Access Control</i>
NCSA	<i>National Center for Supercomputing Applications</i>
PHP	<i>Personal Home Page</i>
SGBD	<i>Sistema de Gerenciamento de Banco de Dados</i>
SSL	<i>Secure Socket Layer</i>
TCP	<i>Transmission Control Protocol</i>
TLS	<i>Transport Layer Security</i>
USB	<i>Universal Serial Bus</i>

SUMÁRIO

1.	INTRODUÇÃO	12
1.1.	OBJETIVOS	13
1.1.1.	Objetivo Geral	13
1.1.2.	Objetivos Específicos	13
1.2.	JUSTIFICATIVA	13
1.3.	ESTRUTURA DO TRABALHO	14
2.	REFERENCIAL TEÓRICO	15
2.1.	REDE WI-FI	15
2.2.	SERVIDOR WEB	17
2.2.1.	Servidor Apache	17
2.2.2.	Linguagem HTML	19
2.2.3.	Linguagem PHP	20
2.3.	BANCO DE DADOS	20
2.3.1.	MySql	21
2.4.	SEGURANÇA NA WEB	22
2.4.1.	Secure Socket Layer (SSL)	22
2.4.2.	Transport Layer Security (TLS)	23
2.5.	ARDUINO	24
2.5.1.	Arduino Uno R3	25
2.5.2.	Relé de Dois Estados	26
3.	MATERIAIS E MÉTODOS	28
3.1.	SERVIDOR WEB	28
3.1.1.	Configuração do Servidor Apache	28
3.2.	CRIAÇÃO DO BANCO DE DADOS	31
3.3.	DESENVOLVIMENTO DO WEBSITE	32
3.3.1.	Identificação Automática de Usuário	33
3.3.2.	Login de Usuário	33
3.3.3.	Painel de controle	35
3.3.4.	Configurações do Cadastro	36
3.3.5.	Cadastro de Usuários	38
3.3.6.	Sair	40
3.3.7.	Criptografia das senhas	41
3.4.	HOTSPOT	42
3.5.	MÓDULO ARDUINO	44
3.5.1.	Relé	45
3.6.	VERIFICAÇÃO DE USUÁRIOS NA REDE	46
4.	RESULTADOS E DISCUSSÃO	48
4.1.	ENSAIOS COM O WEBSITE	48
4.2.	CRIPTOGRAFIA DE SENHA	54
4.3.	SCRIPT	54
4.4.	REALIZAÇÃO DOS ENSAIOS	56
4.4.1.	Ensaio Final de Operação	57
4.5.	PROBLEMAS ENCONTRADOS	58
4.5.1.	Diferentes Resoluções de Telas	58
4.5.2.	Número de Tentativas de Acesso para Bloqueio	59
4.5.3.	Captura do Endereço MAC dos Dispositivos Mobile	59
4.5.4.	Envio de Dados para o Arduino	60
5.	CONCLUSÃO	62

5.1. TRABALHOS FUTUROS.....	63
REFERÊNCIAS BIBLIOGRÁFICAS	64

1. INTRODUÇÃO

Sistemas de automação residencial têm sido frequentemente idealizados pela grande maioria dos usuários como um artifício que possa proporcionar grande conforto e comodidade ao dia a dia. Desenhos animados e filmes antigos já demonstravam estas funcionalidades tão desejadas ao usuário comum com maior simplicidade do que se apresenta na vida real. Na prática, com a chegada dos computadores pessoais, internet, a explosão da telefonia móvel e outras tecnologias que entraram no mundo pessoal dos consumidores, a aceitação das tecnologias de automação residencial passou a ganhar mais força.

É de conhecimento que modernas técnicas de automação envolvem investimentos relativamente grandes para operar com eficiência e segurança contra agentes externos do sistema. É importante observar também que sistemas de certificação de usuários complexos podem oferecer grande proteção contra invasão e operação de agentes externos, mas também oneram maior tempo de entrada de dados, o que pode comprometer a segurança física do usuário pelo maior período de exposição no ambiente externo da residência.

O sistema desenvolvido neste projeto teve como objetivo simplificar um sistema de automação residencial convencional através da utilização de recursos já existentes no próprio ambiente, fornecendo uma interface de usuário de baixo custo, simples e intuitiva, mas oferecendo um modelo robusto de segurança hierárquica. Nesta configuração, inicialmente o usuário é reconhecido em um nível de segurança básico e restrito através da identificação de seu dispositivo móvel já cadastrado, e desta forma possibilita que o sistema forneça uma interação direta apenas pela aproximação de sua residência. Essa identificação é automática e fornece apenas interações simples, como o acender de uma luz externa ou uma sinalização de reconhecimento positivo.

Em níveis de segurança mais avançados, como para abrir um portão ou desligar o sistema de alarme, ele requer que o usuário realize uma certificação mais segura com seu *smartphone*. Através da rede wireless é disponibilizada uma página *web* em modo local, juntamente com uma exigência de senhas criptografadas. Persistindo uma certificação incorreta por três vezes, o alarme da residência é acionado e o usuário só poderá entrar na residência com as chaves ou digitando uma senha cadastrada exatamente para essa situação.

1.1.OBJETIVOS

1.1.1. Objetivo Geral

O objetivo geral deste trabalho foi a implementação de um sistema de automação residencial de baixo custo, utilizando a rede sem fio do próprio ambiente juntamente com as funcionalidades de um módulo Arduino® e um *website* especialmente desenvolvido, para controlar diferentes módulos de segurança de uma residência através de um dispositivo móvel.

1.1.2. Objetivos Específicos

- Realização de uma pesquisa de referencial teórico detalhado sobre as ferramentas utilizadas no projeto.
- Descoberta dos recursos mínimos necessários para o bom funcionamento do projeto.
- Promover a comunicação do dispositivo móvel com o servidor, que é responsável pelo processamento das informações.
- Desenvolver um sistema *web* para realizar a comunicação do dispositivo móvel com o módulo Arduino®.
- Analisar o sistema em operação dentro de um ambiente de testes.

1.2.JUSTIFICATIVA

Com a grande expansão de uso dos recursos oriundos dos computadores pessoais, da internet para o compartilhamento de informações, da aceleração das tecnologias de comunicações móveis, entre outras que entraram no mundo dos consumidores de perfil comum, a aceitação das modernas técnicas para fornecer sistemas de automação de processos passou a ganhar força. Entre essas tecnologias têm-se explorado em larga escala a automação residencial, que vem conquistando seu espaço no mercado gradativamente, pois proporciona conforto, comodidade e segurança aos seus utilizadores, além de ser facilmente adaptado a qualquer utilidade doméstica.

Estes sistemas de automação e controle utilizados para o conforto doméstico também podem eventualmente ser utilizados para segurança e praticidade do dia a dia de todos. Ao contrário de um controle remoto de um aparelho de TV ou ar condicionado, que apenas poupam que o usuário tenha que se deslocar até o equipamento para interagir com o mesmo, a utilização de processos de automação, controle e acesso remoto a determinados sistemas de segurança e gerenciamento residencial requerem alto controle de acesso de usuários, evitando expor os mesmos aos riscos de invasões. De forma análoga, procedimentos de segurança complexos para garantir o acesso seguro aos sistemas podem afastar os usuários devido à falta de praticidade ou tempo de acesso, dificultando a proliferação de uso em larga escala.

Desse modo, diante da demanda mapeada anteriormente para diversos produtos de automação residencial comerciais, este projeto propôs elaborar um moderno sistema de credenciais de usuários autorizados através de níveis de hierarquia, que iniciam com uma simples identificação do dispositivo móvel do usuário e possibilitam que este controle ambientes restritos ao aproximar-se de sua residência. Quando o dispositivo móvel for identificado pelo sistema, por exemplo, as luzes da residência se acenderão automaticamente por um determinado período de tempo. Em níveis mais avançados o usuário deverá, através da rede *Wi-Fi* da residência, acessar uma página *web* desenvolvida e digitar seu *login* criptografado para ter acesso as opções que desejar, como ligar ou apagar outras luzes, trancar ou destrancar portas e ligar ou desligar o alarme. Caso o usuário tente por três vezes efetuar o *login* de forma incorreta, a sua conta será bloqueada, o alarme da residência será acionado e o usuário só poderá entrar na residência com as chaves manuais ou através da digitação de uma senha cadastrada exatamente para esse momento.

1.3. ESTRUTURA DO TRABALHO

A descrição detalhada do desenvolvimento deste projeto está apresentada na forma de capítulos neste documento, iniciando com o referencial teórico (Capítulo 2), após com os materiais e métodos (Capítulo 3), os resultados obtidos (Capítulo 4) e finalizando, a conclusão e trabalhos futuros (Capítulo 5).

2. REFERENCIAL TEÓRICO

O termo automação atualmente é utilizado para qualquer sistema que substitua o trabalho humano ou processos repetitivos, oferecendo segurança, qualidade dos serviços, rapidez na produção ou na redução de custos. Estes procedimentos de automação em geral envolvem desde redes de comunicações até a implantação de sistemas interligados e assistidos. Sistemas de operação supervisória e interfaces de interação homem-máquina auxiliam os operadores e usuários na supervisão ou análise dos problemas que eventualmente ocorram. A vantagem de utilizar estes sistemas é a possibilidade da expansão quase que ilimitada dos recursos e funcionalidades do sistema através de mecanismos de fácil acesso, como os controladores lógicos programáveis, que fazem da automação uma realidade (CASTRUCCI e MORAES, 2007).

Conforme observado anteriormente, os processos de automação envolvem diversos recursos de *hardware* e *software*, assim como tecnologias da informação e comunicação específicas (TIC). A escolha destes recursos de forma eficiente permite que se possa implementar um sistema de automação que atenda aos requisitos mínimos de operação com investimentos de acordo com o orçamento disponível. Desta forma, neste referencial teórico buscou-se fornecer subsídios para as escolhas das tecnologias que mais se adequassem as necessidades do projeto proposto, priorizando a utilização de recursos já disponíveis no ambiente e dispositivos de baixo custo.

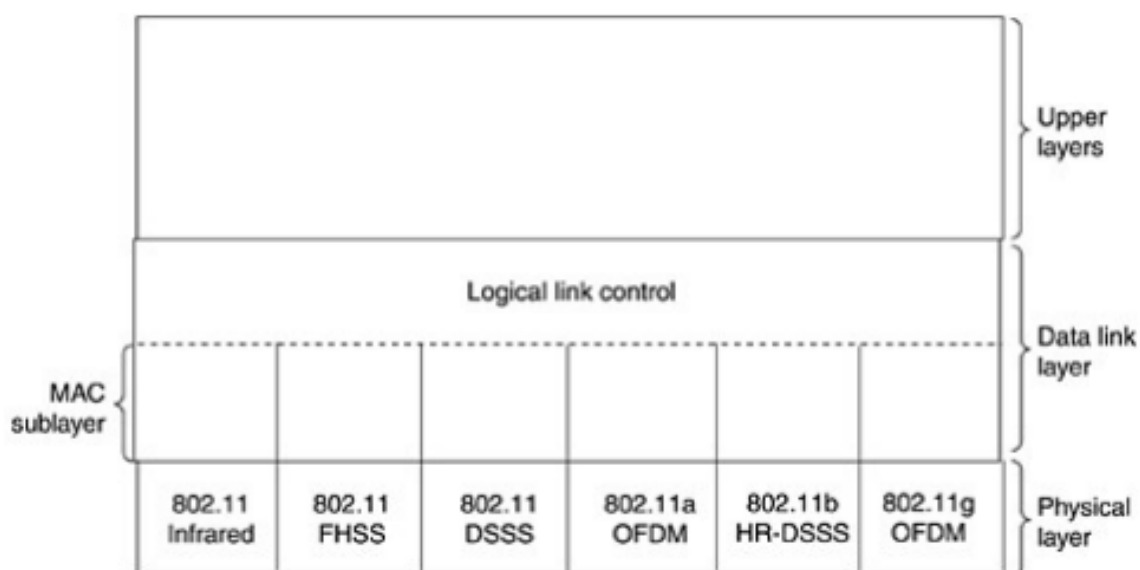
2.1. REDE WI-FI

Quando surgiram os primeiros esforços para implementação das redes *Wi-Fi*, as empresas começaram a criar suas próprias conexões sem fio sem um padrão estabelecido. Na prática, um determinado equipamento de uma marca não era compatível com o da outra marca, e com isso não podiam se comunicar. Para resolver esse problema, a indústria decidiu adotar um único padrão de rede sem fio. O comitê da IEEE (*Institute of Electric and Electronic Engineers*) ficou responsável por elaborar um padrão de rede sem fio, que recebeu o nome de 118802.11 ou 802.11, conhecido como *Wi-Fi* (TANENBAUM, 2003).

Atualmente existem diversos padrões 802.11 estabelecidos, entre eles o 802.11b, 802.11a e 802.11g. Estes três padrões do exemplo utilizam o mesmo protocolo CSMA/CA (*Carrier Sense Multiple Access with Collision Detect*), que evita o envio de pacotes simultaneamente. Eles usam a mesma estrutura de quadros para seus *frames* de camada de enlace, onde possuem a capacidade de reduzir a taxa de transmissão para alcançar distâncias maiores, e permitem o modo de infraestrutura e o modo *ad-hoc*. O modo com infraestrutura é a comunicação por meio de um roteador sem fio, e o modo *ad-hoc* representa a comunicação diretamente com o outro dispositivo (KUROSE e ROSS, 2006).

Segundo Tanenbaum (2003), os protocolos usados por todas os diferentes padrões 802 têm certas características comuns em sua estrutura. A camada física corresponde à camada física do modelo OSI, mas a camada de enlace de dados se divide em duas ou mais subcamadas. No 802.11, a subcamada MAC (*Medium Access Control*) determina quem terá a oportunidade de transmitir em seguida. A subcamada LLC (*Logical Link Control*), apresenta a função de ocultar as diferenças entre as diversas variações do 802 e torná-las indistinguíveis no que se refere à camada de rede. A Figura 1 demonstra uma visão parcial da pilha de protocolos destes padrões de comunicação sem fio.

Figura 1 – Parte da pilha de protocolos do 802.11



Fonte: TANENBAUM (2003).

Ainda não se sabe como será o futuro das redes *Wi-Fi*, porém, sabe-se que estão surgindo no mercado dispositivos com mais funcionalidades e consumindo maior largura de banda. A necessidade de querer usar dispositivos móveis inteligentes, corporativos ou próprios como se estivessem em casa, fazem com que surjam novas tecnologias. Um exemplo é a Internet das Coisas (*Internet of Things*), onde o objetivo é conectar qualquer produto à internet, podendo gerenciá-lo de qualquer lugar. Esse tipo de tecnologia irá mudar drasticamente a implementação de técnicas com base na taxa de transmissão e cobertura da *Wi-Fi* (VERT, 2016).

2.2. SERVIDOR WEB

Um servidor *web* é um computador que hospeda um ou mais *sites*. "Hospedagem" significa que todas as páginas da *web* e seus arquivos de suporte estão disponíveis neste servidor remoto, sendo que este por sua vez, envia qualquer página do *site* que está hospedado para o navegador de acesso ao servidor, por solicitação do próprio usuário (MOZILLA, 2016).

Quando um *site* da internet é acessado pelo navegador do usuário, é realizada uma troca de informações entre o navegador e o servidor. Essa troca utiliza um protocolo que é chamado de HTTP (*HyperText Transfer Protocol*). O objetivo desse protocolo é mostrar o conteúdo do servidor *web* nos navegadores dos usuários de forma padronizada (HOAG, 2002).

2.2.1. Servidor Apache

Em fevereiro de 1995, o serviço HTTP, que é um dos *softwares* de servidores mais popular da *web* foi desenvolvido por Rob McCool no *National Center for Supercomputing Applications* (NCSA) da Universidade de Illinois. No entanto, o desenvolvimento desse servidor *httpd* (*HyperText Transfer Protocol Daemon*) havia estagnado após Rob deixar o NCSA em 1994. Porém, após um período, muitos *webmasters* desenvolveram suas próprias extensões e correções de *bugs* e perceberam que havia a necessidade de uma padronização. Com isso, um pequeno grupo desses *webmasters* reuniu-se com a finalidade de coordenar as suas alterações na forma de *patches*. Brian Behlendorf e Cliff

Skolnick montaram um fórum de troca de conhecimentos para os desenvolvedores do núcleo em uma máquina na área da Baía Califórnia, com largura de banda doado por *HotWired* (APACHE).

Usando o NCSA *httpd* 1.3 como base, eles adicionaram todas as correções publicadas e melhorias enviadas, depois testaram, e o resultado foi o lançamento oficial do primeiro servidor Apache, em abril de 1995. O início do servidor Apache foi um grande sucesso, mas todos sabiam que a base de código precisaria de uma revisão geral em breve.

Desse modo, durante os meses de maio e junho de 1995, projetaram uma nova arquitetura de servidor que incluiu uma estrutura modular, chamada de API (*Application Programming Interface*), para obter uma melhor capacidade de extensão e alocação de memória. Após extensos testes, o Apache 1.0 foi lançado em 01 de dezembro de 1995. Menos de um ano após a formação do grupo, o servidor Apache *httpd* saiu do NCSA como o servidor número um na Internet e, de acordo com a pesquisa da Netcraft, mantém essa posição até hoje. Em 1999, os membros do Grupo Apache formaram a *Apache Software Foundation* para fornecer suporte organizacional, legal e financeiro para o Apache HTTP Server. A fundação tem colocado o *software* em uma base sólida para o desenvolvimento futuro, e expandiu significativamente o número de projetos de *software Open Source* (APACHE, 2016).

No mês de março deste ano, o número de *sites* passou de 1 bilhão, contabilizados desde setembro de 2014. Porém o número de *sites* ativos é de 171 milhões, ou seja, 1 em cada 6 *sites* está ativo. Foi registrado uma perda de quase 1 milhão de *sites* ativos da Apache. Mas apesar dessas perdas em *site* ativos, o servidor Apache continua muito à frente de outros servidores, como mostra o Tabela 1 que corresponde a porcentagem de *sites* ativos (NETCRAFT, 2016).

Tabela 1 – Desenvolvedores de servidores *web*: Mercado de *sites* ativos

Desenvolvedor	Fevereiro 2016	Porcentagem	Março 2016	Porcentagem	Mudança
Apache	84,790,816	49,50%	83,825,658	49.16%	-0.34
nginx	27,327,583	15,95%	28,026,677	16.44%	0.48
Microsoft	17,257,986	10,08%	17,228,197	10.10%	0.03
Google	13,596,295	7,94%	13,545,864	7.94%	0.01

Fonte: NETCRAFT (2016).

2.2.2. Linguagem HTML

A linguagem de marcação HTML (*HyperText Markup Language*) surgiu no final dos anos 80, sendo desenvolvida pelo físico Tim Berners-Lee, que trabalhava no CERN (*Conseil Européen pour la Recherche Nucléaire*). Baseava-se em armazenar um conteúdo em um servidor remoto e acessá-lo através de estações de trabalho por meio de um navegador. Isso permitiu a disponibilização de páginas com conteúdos mais enriquecidos, como a exibição de imagens associadas (MOZILLA, 2014).

A linguagem HTML possui a capacidade de distinguir e separar o conteúdo de sua forma de apresentação, onde utiliza um conjunto pré-definido de elementos para identificar os diversos tipos de conteúdo e realizar a otimização de apresentação para diferentes formatos. Esses elementos contêm uma ou mais *tags* que incluem ou expressam um determinado conteúdo (MOZILLA, 2014). As *tags* são caracteres especiais que informam como o conteúdo deverá ser exibido na tela do usuário (RAMALHO, 1996).

A última versão lançada foi o HTML5, que corrige vários problemas da versão anterior HTML4, como facilitar a manipulação do elemento possibilitando ao desenvolvedor modificar características dos objetos de maneira transparente para o usuário final. Esta atualização da linguagem também criou novas *tags* e modificou a função de outras já utilizadas. Essa nova versão também modificou

a forma de escrever os códigos e organizar as informações na página, onde ficou mais interativa sem a necessidade da instalação de *plugins*, evitando a perda de desempenho. Embora seja novo e possua funcionalidades adicionais, o HTML5 está sendo desenvolvido para ser compatível com qualquer navegador, e nenhum *site* precisará ser refeito para se adequar à nova versão (W3C, 2010).

2.2.3. Linguagem PHP

O PHP foi criado em 1994 por Rasmus Lerdorf, e era utilizado para acompanhar quantas visitas o seu currículo online recebia. Esse conjunto de *scripts* recebeu o nome de *Personal Home Page Tools*, que depois passou a ser conhecido como *PHP Tools*. Depois do sucesso de utilização, um novo modelo foi reescrito, adicionando interação com banco de dados e uma estrutura em que páginas *web* simples e dinâmicas pudessem ser desenvolvidas. A linguagem foi desenvolvida para ser parecida com C, Perl e outras linguagens similares, sendo exclusiva de sistemas Unix (PHP, 2016a).

Ela consiste em uma linguagem *open source*, desse modo, qualquer usuário tem acesso ao seu código-fonte. O PHP contém HTML em código mesclado e as instruções são delimitados pelas *tags* de início `<?php` e fim `?>`. O código do PHP é executado no servidor, com isso, apenas a página HTML é enviada para o navegador do cliente (PHP, 2016a).

A linguagem PHP pode-se realizar diversas tarefas, como por exemplo, coletar dados de formulários, gerar páginas com conteúdo dinâmico e enviar ou receber *cookies*. Ela também pode ser utilizada em qualquer sistema operacional, sendo suportado pela maioria dos servidores *web*. Uma das características mais relevantes do PHP é o suporte a uma variedade de banco de dados (PHP, 2016b). Em 2015 foi lançada a versão 7, sendo a maior atualização desde a versão 5, lançada em 2004.

2.3. BANCO DE DADOS

Segundo Silberschatz et al. (2006), um sistema de banco de dados consiste em uma coleção de dados inter-relacionados e conjuntos de programas, que permitem aos usuários acessar e modificar esses dados. Uma importante

finalidade é fornecer uma visão abstrata dos dados, ou seja, o sistema não mostra detalhes de como os dados são armazenados e mantidos.

Os sistemas de banco de dados são projetados para gerenciar grandes blocos de dados. Envolve definir estruturas para armazenamento e fornecer mecanismos para manipular as informações. Além disso, precisa garantir a segurança das informações armazenadas, mesmo com falhas de sistema ou tentativas de acesso negado. Ao compartilhar os dados entre vários usuários, o sistema tem que evitar resultados anômalos (SILBERSCHATZ et al., 2006).

2.3.1. MySql

O MySql surgiu da necessidade dos desenvolvedores em utilizar um sistema para conectar suas tabelas. A intenção era utilizar o mSQL, mas depois de alguns testes, verificaram que essa ferramenta não era rápida o suficiente para as necessidades deles. A solução foi criar uma nova interface SQL, projetada para aceitar códigos de terceiros, que se chamaria MySql (MYSQL, 2016).

Segundo Milani (2006), o MySql é um sistema de gerenciamento de banco de dados (SGBD) de licença dupla. A primeira licença é gratuita e baseada nas cláusulas da GNU-GPL (*General Public Licence*), que estabelece o que pode ou não fazer com a ferramenta. Seu código-fonte também é disponibilizado para que qualquer usuário qualificado possa alterar conforme as necessidades de cada aplicação. A segunda licença é comercial, utilizada para obter suporte diferenciado e pacotes com mais ferramentas.

No começo foi projetado para trabalhar com pequenas e médias aplicações. Hoje ele possui todas as características de um banco de dados de grande porte, sendo reconhecido como um dos maiores sistemas *open source*, com capacidade de concorrer igualmente com sistemas de códigos fechados comerciais (MILANI, 2006).

O mesmo autor relata que o MySql por ser escrito em C e C++, e funciona em quase todos os sistemas operacionais existentes, como Linux (Fedora Core, Debian, Suse, RedHat), Unix (Solaris, HP-UX, AIX, ACO), FreeBSD, Mac OS X Server e Windows. Também fornece uma interface API (*Application Programming Interface*) para várias linguagens, como Java, Python, PHP, Perl,

C e C++. Ele utiliza *threads* diretamente no *kernel*, e isso faz com que aumente significativamente a velocidade de processamento.

A última versão lançada foi a 5.7, em 19 de outubro de 2015, oferecendo maior desempenho, escalabilidade e capacidade de gerenciamento em relação as versões anteriores (ORACLE, 2015).

2.4.SEGURANÇA NA WEB

No começo da expansão da internet para o grande público, ela servia somente para distribuir páginas estáticas. Posteriormente, algumas empresas começaram a usar a internet para realizar transações financeiras, como compra de mercadorias por cartões de crédito e transações bancárias *on-line*. Com isso, surgiu uma necessidade por obter uma maior segurança dos dados para realizar essas transações (TANENBAUM, 2003).

2.4.1. Secure Socket Layer (SSL)

O protocolo SSL foi desenvolvido pela Netscape em 1994, sendo que a versão 3 foi projetada e publicada na internet. Em seguida, a IETF (*Internet Engineering Task Force*), responsável por estabelecer normas para a internet, chegou a um acordo para que o protocolo fosse padronizado. Esse acordo resultou no surgimento do TLS (STALLINGS, 2008).

O SSL é uma camada que se localiza entre a camada de aplicação e a camada de transporte, como pode-se observar na Figura 2. O protocolo aceita solicitações do navegador e as envia para a camada TCP, para transmiti-la ao servidor. Quando a conexão é estabelecida, o SSL manipula a compactação e faz a criptografia. Quando o HTTP (*HyperText Transfer Protocol*) é utilizado sobre o SSL, passa a se chamar HTTPS (*Secure HTTP*) (TANENBAUM, 2003).

Figura 2 – Camadas (e protocolos) para usuários domésticos com SSL

Aplicação (HTTP)
Segurança (SSL)
Transporte (TCP)
Rede (IP)
Enlace de dados (PPP)
Física (modem, ADSL, TV a cabo)

Fonte: TANENBAUM (2003).

Dois conceitos são importantes no SSL, um é a sessão, que realiza a associação entre um cliente e um servidor, e também define conjuntos de parâmetros de segurança criptográficas que podem ser compartilhadas entre as sessões, para evitar a negociação de novos parâmetros de segurança para cada sessão. Outro é a conexão, que são enlaces ponto-a-ponto, onde cada ligação está conectada a uma sessão (STALLINGS, 2008).

2.4.2. Transport Layer Security (TLS)

O protocolo TLS baseia-se no SSL, porem são diferentes e não podem comunicar-se entre si. Neste modelo, o modo com a chave de sessão é derivado da chave pré-mestre, e mudou para tornar a chave mais forte, ou seja, ficou mais difícil de ser violada (TANENBAUM, 2003).

No protocolo em questão duas tecnologias de criptografia são utilizadas:

- A chave pública, que utiliza um par de chaves, sendo uma pública e outra privada. As informações criptografadas com a chave pública só poderão ser descriptografadas com a chave privada. A chave pública é de livre acesso e a chave privada fica em segredo (SERPRO, 2009).
- A chave simétrica, que aproveita a mesma chave para criptografar e descriptografar as trocas de mensagens. A chave é gerada aleatoriamente pelo cliente e utilizada para criptografar os dados

da sessão. Depois, é criptografada com a chave pública do servidor e enviada para o servidor (SERPRO, 2009).

A exclusividade da mensagem, integridade da mensagem e autenticação mútua são três serviços básicos do TLS. A primeira é uma combinação da criptografia de chave pública e chave simétrica, em que todo o tráfego entre o cliente e servidor é criptografado utilizando uma chamada e um algoritmo de criptografia durante a sessão. O outro serviço, de integridade da mensagem, garante que o tráfego da sessão não altere a rota até seu destino. Finalmente para a autenticação mútua, as identidades do cliente e do servidor são codificadas em certificados de chave pública e depois são trocados entre os pontos (SERPRO, 2009).

2.5. ARDUINO

O Arduino® surgiu no *Ivrea Interaction Design Institute* como uma ferramenta destinada a estudantes sem experiência em eletrônica e programação. É uma ferramenta de prototipagem *open source*, podendo ser adaptada às necessidades de quem utilizar. O *software* utilizado para programar o Arduino® também é *open source*, muito semelhante aos comandos utilizados na linguagem de programação C (ARDUINO, 2016).

Com o Arduino® é possível programar processos de entradas e saídas nos dispositivos que estão conectados nele. Ele é um sistema embarcado, ou seja, permite a interação de *software* e *hardware* ao ambiente com objetivo pré-definido. Pode ser utilizado para desenvolver objetos interativos, assim como pode ser conectado no computador, na rede, internet e agir sobre eles. Além disso, pode ser conectado a qualquer dispositivo que envie, receba dados e que possa ser controlado (MCROBERTS, 2011).

A linguagem de programação utilizada para fazer o Arduino® interagir com os componentes é baseada na linguagem C. Utiliza-se uma interface IDE, que é um *software* livre, para escrever o código ou *sketches* e fazer o *upload* da rotina pronta para o módulo Arduino®. Depois de realizado o *upload*, o dispositivo Arduino® executa este código e envia instruções aos dispositivos que estão conectados. A interface IDE é compatível com quase todos os sistemas operacionais existentes (MCROBERTS, 2011). A simplicidade da interface e sua

modularidade para diferentes aplicações, permitem o desenvolvimento de praticamente qualquer interface de automação de processos com baixo custo.

2.5.1. Arduino Uno R3

O Arduino® Uno que está demonstrado na Figura 3, é um módulo baseado no microcontrolador ATmega328P, que possui um *bootloader* para permitir o envio de novos códigos de modo simples. Ele possui 14 pinos digitais de entrada e saída, 6 entradas analógicas, uma porta USB, tomada de energia e tomada de reset. Para usá-lo é só conectar a um computador ou fonte de energia, sendo possível a troca do *chip* em caso de defeito (ARDUINO, 2016).

Figura 3 – Modelos dos módulos Arduino® Uno R3



Fonte: <https://www.arduino.cc/en/Main/ArduinoBoardUno>.

A alimentação do Arduino® pode ser realizada através da conexão USB ou pela fonte de alimentação externa. Ele pode operar entre 6 e 20 Volts, porém se forem fornecidos menos de 7 Volts, a placa pode se tornar instável, assim como se usar mais de 12 Volts, pode danificar a placa (ARDUINO, 2016). A Tabela 2 apresenta as especificações do Arduino® Uno utilizado.

Tabela 2 – Especificações do Arduino® Uno

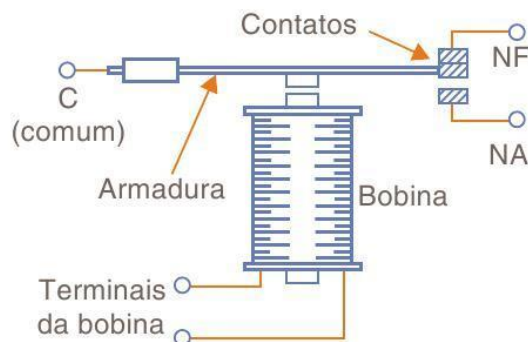
Microcontrolador:	ATmega328P
Tensão de Operação:	5V
Tensão de Entrada (recomendada):	7-12V
Tensão de Entrada (limite):	6-20V
Pinos I/O Digitais:	14 (com 6 saídas PWM)
Pinos I/O PWM Digitais:	6
Pinos Analógicos de Entrada:	6
Corrente DC por Pino I/O:	20 mA
Corrente DC para o Pino de 3,3V:	50 mA
Memória <i>Flash</i> :	32 kB (ATmega328P)
SRAM:	2 kB (ATmega328P)
EEPROM:	1 kB (ATmega328P)
Velocidade do Relógio:	16 MHz
Comprimento:	68,6 mm
Largura:	53,4 mm
Peso:	25 g

Fonte: <https://www.arduino.cc/en/Main/ArduinoBoardUno>

2.5.2. Relé de Dois Estados

O relé é um dispositivo eletromecânico que possui um magneto móvel. Esse magneto é utilizado para conectar os terminais metálicos quando uma corrente circula por sua bobina. Ao passar essa corrente pela bobina, um campo magnético é gerado na extremidade, abrindo ou fechando os contatos de acordo com sua característica construtiva. Os contatos podem ser NA (normalmente aberto), NF (normalmente fechado) ou C (comum). O NA está aberto quando a bobina não está energizada, diferentemente do NF, em que os contatos abrem quando está energizada. O terminal comum serve para estabelecer a ligação da energia através da alternância entre os contatos NA e NF do relé (SANTOS, 2016). A Figura 4 demonstra a estrutura básica de um relé de dois estados.

Figura 4 – Estrutura básica do relé de dois estados



Fonte: <http://www.sabereletronica.com.br/artigos/1702-testando-rele>

Pode-se energizar um relé com correntes e tensões pequenas em relação aos parâmetros que o circuito exige para funcionar, ou seja, pode-se controlar circuitos de altas correntes, que a partir de transistores e circuitos integrados são considerados fracos (INSTITUTO NEWTON C. BRAGA, 2014).

Uma característica importante é a segurança de isolamento que o relé oferece ao circuito que está sendo controlado. Não existe conexão entre a bobina e os contatos do relé, e desse modo, não há a possibilidade de alguma corrente que ativa o relé ir para o circuito que está sendo controlado e vice-versa (INSTITUTO NEWTON C. BRAGA, 2014).

3. MATERIAIS E MÉTODOS

Neste capítulo estão sendo demonstrados os materiais e os métodos utilizados para o desenvolvimento deste projeto de automação residencial através dos recursos disponíveis no ambiente, integrados com um módulo Arduino® Uno e uma página *web* para acesso pelo dispositivo *mobile*.

Inicialmente estão sendo descritos os procedimentos e ferramentas utilizados para a identificação automática dos usuários autorizados a acessarem o sistema, a criação do banco de dados, os mecanismos de proteção de acesso por invasores não credenciados, a criação e disponibilização da página *web* para acesso às funcionalidades de níveis hierárquicos mais elevados, e o sistema de gerenciamento e integração das ferramentas e dispositivos utilizados.

Na sequência estão sendo apresentados os dispositivos de *hardware* utilizados para a integração do sistema com a residência. O sistema de controle desenvolvido opera como tomador de decisões a partir das respostas obtidas pelos mecanismos de identificação e certificação dos usuários, e a partir deste momento, ele necessita acionar o *hardware* desenvolvido para atuar como um dispositivo de automação ou de sinalização de anormalidades.

3.1. SERVIDOR WEB

Para hospedar o *site*, foi utilizado o servidor *web* Apache na versão 2.4.7, instalado no sistema operacional Linux Mint 17.2. O servidor *web* não possui acesso à internet por segurança, oferecendo apenas acesso pela rede local. O *notebook* utilizado para instalar o Apache possui a seguinte configuração:

- Memória: 8 GB.
- Processador: Core i5, 1,7 GHz.
- Hard Disk: 1 TB.

3.1.1. Configuração do Servidor Apache

A configuração do servidor Apache foi realizada em 3 etapas. Na primeira etapa foi realizada a instalação, a criação dos diretórios do *website* e as permissões de acesso aos diretórios, como demonstrado na Figura 5.

Figura 5 – Definição dos diretórios do *site*

```
lucas@lucas-270E5J-2570EJ ~ $ ls /var/www/sistema.com/ -l
total 52
-rw-r--r-- 1 lucas lucas 2100 Mai 10 16:57 cadastrar.php
-rw-r--r-- 1 lucas lucas 4377 Mai 10 17:05 cadastro.php
-rw-r--r-- 1 lucas lucas 5146 Mai 10 16:57 configuracoes.php
drwxr-xr-x 3 lucas lucas 4096 Mar 8 20:44 css
drwxr-xr-x 2 lucas lucas 4096 Mar 1 20:53 funcoes
-rw-r--r-- 1 lucas lucas 1209 Abr 17 11:54 index.php
-rw-r--r-- 1 root root 6 Feb 1 15:21 lista.txt
-rw-r--r-- 1 www-data www-data 61 Jan 25 18:59 maclist.txt
drwxr-xr-x 3 lucas lucas 4096 Jan 3 15:33 nbproject
-rw-r--r-- 1 lucas lucas 5466 Mai 10 16:57 painel_controle.php
lucas@lucas-270E5J-2570EJ ~ $
```

Fonte: Autor

Na segunda etapa da configuração do servidor Apache foi adicionado o nome do *site* no arquivo `/etc/hosts`, para que o usuário não precise digitar o IP do servidor, conforme está sendo demonstrado na Figura 6.

Figura 6 – Adicionando nome do *site* no arquivo *hosts*

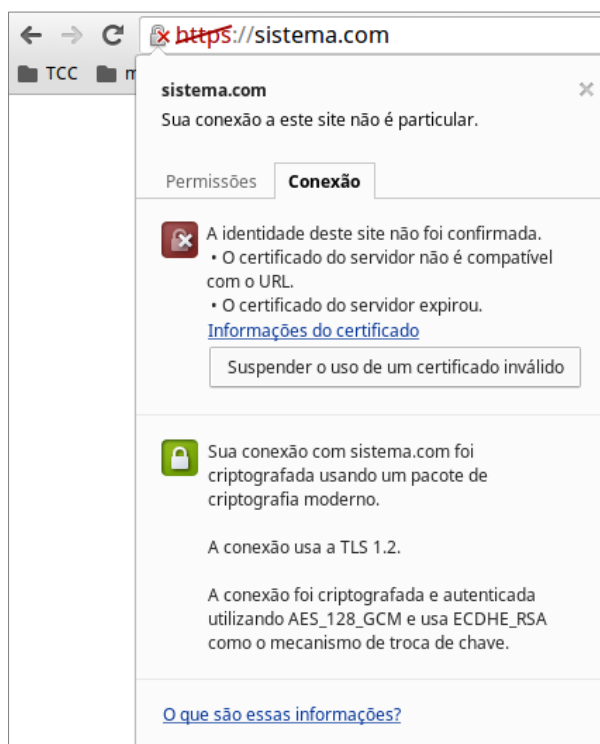
```
lucas@lucas-270E5J-2570EJ ~ $ cat /etc/hosts
127.0.0.1 localhost
127.0.0.1 sistema.com
127.0.0.1 www.sistema.com
192.168.150.1 sistema.com
192.168.150.1 www.sistema.com

# The following lines are desirable for IPv6 capable hosts
::1 ip6-localhost ip6-loopback
fe00::0 ip6-localnet
ff00::0 ip6-mcastprefix
ff02::1 ip6-allnodes
ff02::2 ip6-allrouters
```

Fonte: Autor.

A terceira e última etapa deste processo de implementação foi a configuração do certificado HTTPS. A conexão com o servidor utiliza o pacote de criptografia TLS 1.2, e o certificado foi criado usando o SHA-512 com chave pública de 2048 bits. A Figura 7 apresenta a configuração que foi utilizada para segurança do HTTPS.

Figura 7 – Configuração do HTTPS



Fonte: Autor.

Da mesma forma, a Figura 8 demonstra a configuração do HTTPS no arquivo do servidor Apache.

Figura 8 – Configurando o HTTPS no Apache

```
<VirtualHost *:443>

    ServerName sistema.com
    ServerAlias www.sistema.com
    ServerAdmin webmaster@localhost
    DocumentRoot /var/www/sistema.com/

    SSLEngine on
    SSLCertificateFile /etc/apache2/ssl/certificado.crt
    SSLCertificateKeyFile /etc/apache2/ssl/server.key

    ErrorLog ${APACHE_LOG_DIR}/error.log
    CustomLog ${APACHE_LOG_DIR}/access.log combined

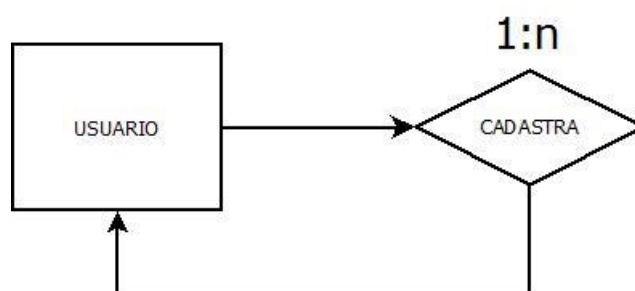
</VirtualHost>
```

Fonte: Autor.

3.2. CRIAÇÃO DO BANCO DE DADOS

O banco de dados escolhido para este projeto foi o MySQL. Optou-se por utilizar este banco de dados por já possuir conhecimento do mesmo em uma disciplina do curso. O desenvolvimento do banco de dados foi dividido em duas etapas. A primeira foi o desenvolvimento do Modelo Entidade Relacionamento (MER) e do Diagrama Entidade Relacionamento (DER). No MER, verificou-se que seria necessário utilizar somente uma entidade contendo todos os atributos, conforme está demonstrado na Figura 9.

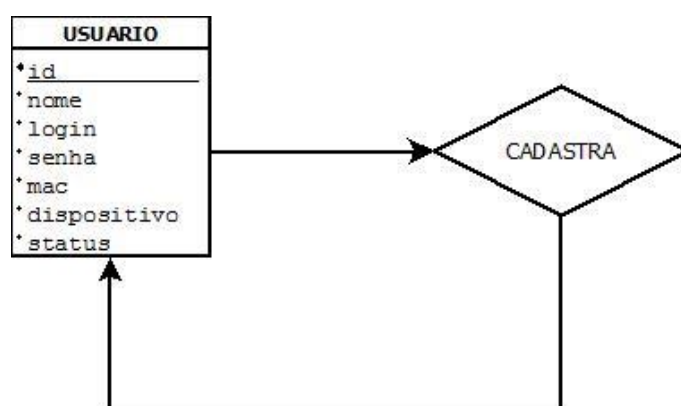
Figura 9 – Modelo Entidade Relacionamento



Fonte: Autor.

O DER está demonstrado na Figura 10, onde pode-se observar a representação gráfica do banco de dados desenvolvido.

Figura 10 – Diagrama Entidade Relacionamento



Fonte: Autor.

Na segunda etapa do desenvolvimento realizou-se a criação do banco de dados, dos seus atributos e da tabela utilizada. Na Figura 11 pode-se observar a tabela do banco de dados pronta.

Figura 11 – Tabela do banco de dados

```
mysql> desc usuario;
```

Field	Type	Null	Key	Default	Extra
id	int(11)	NO	PRI	NULL	auto_increment
nome	varchar(50)	NO		NULL	
login	varchar(50)	NO		NULL	
senha	varchar(600)	NO		NULL	
mac	varchar(50)	NO		NULL	
dispositivo	varchar(50)	NO		NULL	
status	varchar(20)	NO		NULL	

7 rows in set (0.00 sec)

Fonte: Autor.

3.3.DESENVOLVIMENTO DO WEBSITE

O *site* é um dos itens mais importantes deste trabalho, pois é responsável por garantir a segurança e a funcionalidade do projeto. Todo o tratamento das informações é de responsabilidade do mesmo, desde o *login* e cadastro até o acionamento das funcionalidades de *hardware*. Para ter acesso ao *site* é necessário estar conectado na rede *Wi-Fi* local através de um *hotspot*, pois ele não possui conexão com a internet. Essa medida foi adotada por garantir a segurança contra invasões externas pela rede, uma vez que o usuário precisa entrar na área de cobertura do *hotspot* para tentar se conectar ao sistema. Desta forma o *website* se previne de ações de *hackers* e *crackers* de todo o mundo.

Depois de realizar uma avaliação dos requisitos do sistema e as ferramentas de desenvolvimento disponíveis, a linguagem PHP foi selecionada para desenvolver a interface do *site*, para fazer a comunicação com o banco de dados, a interação com o *hardware* do Arduino®, assim como para fazer a verificação e validação dos dados de usuários. Para descrever de uma forma

mais organizada e clara o funcionamento do *site*, o desenvolvimento e operação do mesmo está sendo demonstrado em etapas.

3.3.1. Identificação Automática de Usuário

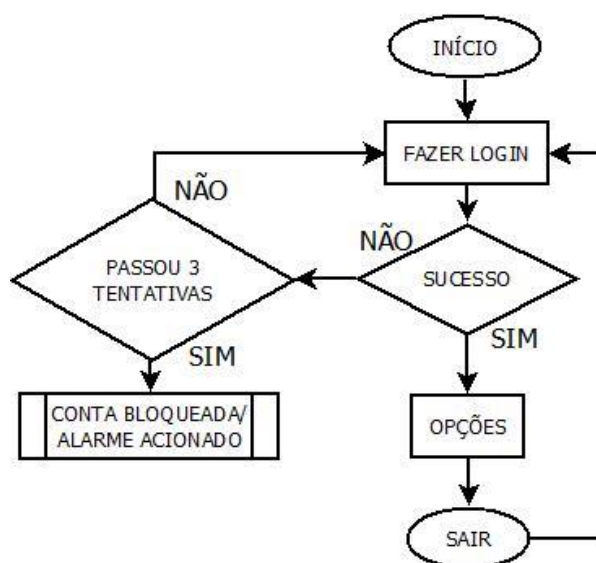
Como a proposta deste projeto é fornecer um sistema de automação residencial que utilize os próprios recursos do ambiente, mas que ao mesmo tempo seja prático de utilizar e forneça segurança contra invasões, foi definida a utilização de uma identificação preliminar de aproximação do usuário.

Para este desenvolvimento, foram consideradas duas premissas básicas, sendo uma delas a praticidade de utilização, minimizando o tempo de interação, e a segunda a disponibilização de uma ação não invasiva. Para esta funcionalidade, inicialmente definida como apenas para o acendimento da iluminação externa para observação segura do perímetro, foi configurada a identificação do usuário pelo MAC de rede do seu dispositivo *mobile*, previamente cadastrado no sistema. É de conhecimento que esse parâmetro pode ser clonado para outros dispositivos, tornando esta credencial relativamente fraca, e desta forma não segura.

Neste nível de segurança de credencial hierarquicamente baixo, a interação com o sistema não permite que o usuário interaja com o ambiente de forma invasiva, mas que seja devidamente identificado de forma automática ao entrar na área de cobertura do sinal *Wi-Fi*. Desta forma, o usuário não necessita realizar nenhuma ação ao chegar na residência, o que em geral causa desatenção ao entorno e um tempo relativamente grande de exposição na área exterior para a entrada de uma credencial mais segura.

3.3.2. Login de Usuário

Para efetuar o *login* no *site* desenvolvido para este projeto, o usuário obrigatoriamente necessita estar dentro da área de cobertura da rede *Wi-Fi* local. Se identificado de forma positiva, então são permitidas três tentativas de *login* deste dispositivo, e se tentar três vezes erradas, a conta é bloqueada e o alarme acionado, como mostra o fluxograma da Figura 12.

Figura 12 – Fluxograma do *login*

Fonte: Autor.

A Figura 13 apresenta o código de bloqueio de usuário do *login*.

Figura 13 – Código de bloqueio de usuário e acionamento do alarme

```

// FAZ A CONTADE DE TENTATIVA DE LOGIN
if ($_SESSION[$tentativa] > 3){
    $sql = mysqli_query($mysqli, "UPDATE usuario SET status='inativo' where login='$_SESSION[$login]'");
    echo "<meta http-equiv='refresh' content='0; URL=../index.php'>";
    function alerta($mensagem){
        echo "<script>alert('".$mensagem."')</script>";
    }
    $site = "Seu login foi bloqueado!";
    alerta($site);
    /* LIGANDO ALARME */
    $port = fopen("/dev/ttyACM0", "w");
    fwrite($port, "6");
    fclose($port);
    /*-----*/
    $_SESSION[$tentativa] = 0;
}
else {
    echo "<meta http-equiv='refresh' content='0; URL=../index.php'>";
    function alerta($mensagem){
        echo "<script>alert('".$mensagem."')</script>";
    }
    $site = "Usuario ou senha incorretos!";
    alerta($site);

    $_SESSION[$login] = $login;
    $_SESSION[$tentativa] ++;
}
}

```

Fonte: Autor.

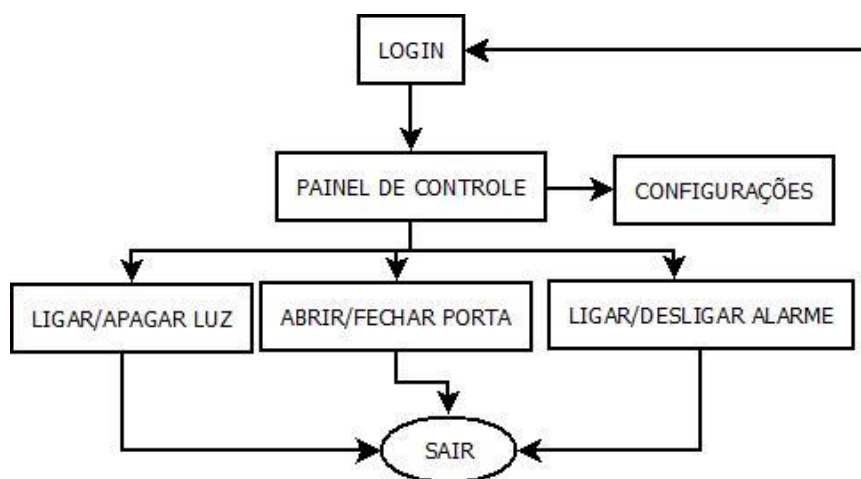
3.3.3. Painel de controle

Conforme descrito anteriormente, inicialmente o usuário é automaticamente identificado ao entrar na área de cobertura do *hotspot* da residência através do MAC da placa de rede de seu dispositivo *mobile*. Caso a identificação seja positiva, o sistema acende a iluminação externa para observação do perímetro da residência, autorizando o usuário a acessar o sistema de controle de funcionalidades.

Para esta etapa de credenciamento, o usuário já necessita de uma certificação de nível hierárquico maior, uma vez que poderá acessar funcionalidades mais invasivas do sistema. Assim que ele realiza o *login* de acesso ao sistema através da interface *web* desenvolvida, que foi disponibilizada apenas na rede local e para usuários previamente identificados, já pode visualizar um painel de controle gráfico com as funcionalidades disponíveis.

O painel de controle é mostrado somente depois que o usuário efetua o *login* com sucesso e não está previamente bloqueado. Nesta página são apresentadas as opções de controle que ele poderá utilizar na sua residência, como mostrado na Figura 14 através do fluxograma de processos. Esses comandos e funções são totalmente configuráveis de acordo com a necessidade de cada usuário, uma vez que acessam portas específicas do módulo Arduino®, e estão conectados a relés de alta corrente e proteções elétricas.

Figura 14 – Fluxograma do painel de controle



Fonte: Autor.

Na Figura 15 pode-se observar fragmento do código que interpreta as entradas de ações solicitadas pelo usuário através do painel de controle e envia os comandos para o Arduino® atuar na automação.

Figura 15 – Código de envio de comandos para o Arduino®

```
<body>
  <?php
    $estado = filter_input(INPUT_GET, "aux", FILTER_DEFAULT);
    $port = fopen("/dev/ttyACM0", "w");

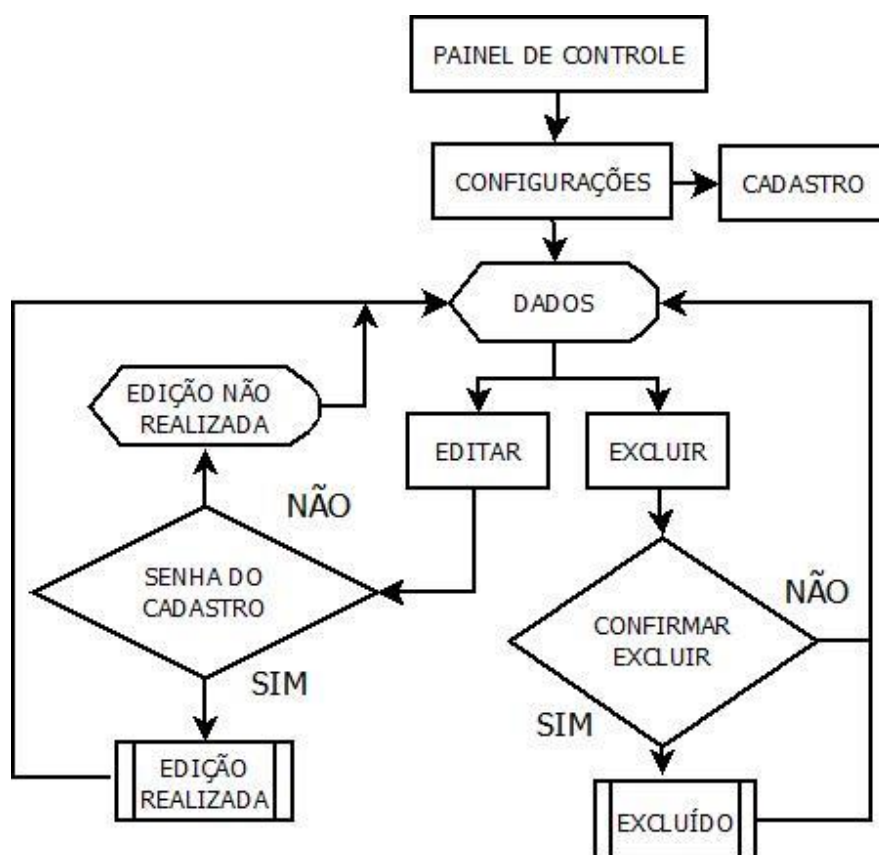
    if ($estado == "ligLuz"){
        fwrite($port, "2");
        header("Location: ../painel_controle.php?aux=ligLuz");
    }
    if ($estado == "apagLuz"){
        fwrite($port, "3");
        header("Location: ../painel_controle.php?aux=apagLuz");
    }
    if ($estado == "abriPorta"){
        fwrite($port, "4");
        header("Location: ../painel_controle.php?aux=abriPorta");
    }
    if ($estado == "fechaPorta"){
        fwrite($port, "5");
        header("Location: ../painel_controle.php?aux=fechaPorta");
    }
    if ($estado == "ligAlarme"){
        fwrite($port, "6");
        header("Location: ../painel_controle.php?aux=ligAlarme");
    }
    if ($estado == "deslAlarme"){
        fwrite($port, "7");
        header("Location: ../painel_controle.php?aux=deslAlarme");
    }
    fclose($port);
  ?>
</body>
```

Fonte: Autor

3.3.4. Configurações do Cadastro

Nas configurações do cadastro o usuário pode verificar os dados que o sistema contém. Ele pode editar, excluir e desbloquear um cadastro se necessário. Quando o usuário for editar algum cadastro é necessário digitar uma senha para fazer a edição com sucesso. A Figuras 16 apresenta o fluxograma simplificado das configurações de cadastro disponíveis.

Figura 16 – Fluxograma das configurações de cadastro



Fonte: Autor.

A Figura 17 apresenta um exemplo do código para excluir um cadastro.

Figura 17 – Código da função excluir

```
// FUNÇÃO EXCLUIR
if (filter_input(INPUT_GET, "funcao", FILTER_DEFAULT) == "excluir") {

    echo "<meta http-equiv='refresh' content='0; URL=../configuracoes.php'>";
    function alerta($mensagem){
        echo '<script>alert("'" . $mensagem . "')</script>';
    }
    $site = "Excluído com sucesso!";
    alerta($site);
    $sql = mysqli_query($mysqli, "DELETE FROM usuario WHERE id='$id'");

}
```

Fonte: Autor.

3.3.5. Cadastro de Usuários

A próxima etapa descrita é o cadastro de usuários no sistema. Nele o usuário terá que colocar o nome, *login*, senha, confirmar a senha, endereço MAC do dispositivo e o nome do dispositivo. Como início do cadastro foi definido que senhas com menos de seis dígitos são consideradas fracas, e desta forma não serão aceitas pelo sistema. Na Figura 18 está sendo apresentado uma parte do código que identifica essa característica e bloqueia as senhas fracas.

Figura 18 – Código de senha fracas

```
// VERIFICA SENHA FRACA
if (!isset($senha[5])) {
    echo "<meta http-equiv='refresh' content='0; URL=cadastrar.php'>";
    function alerta($mensagem){
        echo '<script>alert("'" . $mensagem . "')</script>';
    }
    $site = "Senha muito curta, escolha outra!";
    alerta($site);
    return;
}
elseif (($senha === '012345') or ($senha === '0123456') or ($senha === '01234567') or
        ($senha === '012345678') or ($senha === '0123456789') or ($senha === '012345678910')){
    echo "<meta http-equiv='refresh' content='0; URL=cadastrar.php'>";
    function alerta($mensagem){
        echo '<script>alert("'" . $mensagem . "')</script>';
    }
    $site = "Sua senha é muito fraca, insira outra!";
    alerta($site);
    return;
}
elseif (($senha === 'qwerty') or ($senha === 'abc123') or ($senha === '123abc') or
        ($senha === 'qwerty123') or ($senha === 'abcdef')){
    echo "<meta http-equiv='refresh' content='0; URL=./cadastrar.php'>";
    function alerta($mensagem){
        echo '<script>alert("'" . $mensagem . "')</script>';
    }
    $site = "Sua senha é muito fraca, insira outra!";
    alerta($site);
    return;
}
```

Fonte: Autor.

De forma semelhante ao requisito anterior, também foi definido que a partir de um controle de credenciais de acesso pré-cadastradas, nenhum usuário poderá atribuir duas identificações de *login* iguais através do código que está sendo demonstrado na Figura 19.

Figura 19 – Código que verifica *login* iguais

```
// VERIFICA LOGIN EXISTE
$sql = mysqli_query($mysqli, "SELECT * FROM usuario WHERE login = '$login'");
if (mysqli_num_rows($sql)>=1) {
    echo "<meta http-equiv='refresh' content='0; URL=cadastrar.php'>";
    function alerta($mensagem){
        echo '<script>alert("$. $mensagem.")</script>';
    }
    $site = "Este login já existe, por favor, escolha outro!";
    alerta($site);
    return;
}
```

Fonte: Autor.

Também importante, no momento do cadastro todos os campos terão que ser devidamente preenchidos conforme demonstrado no código da Figura 20. O cadastro só poderá ser realizado por um usuário que já esteja cadastrado.

Figura 20 – Código de campos preenchidos

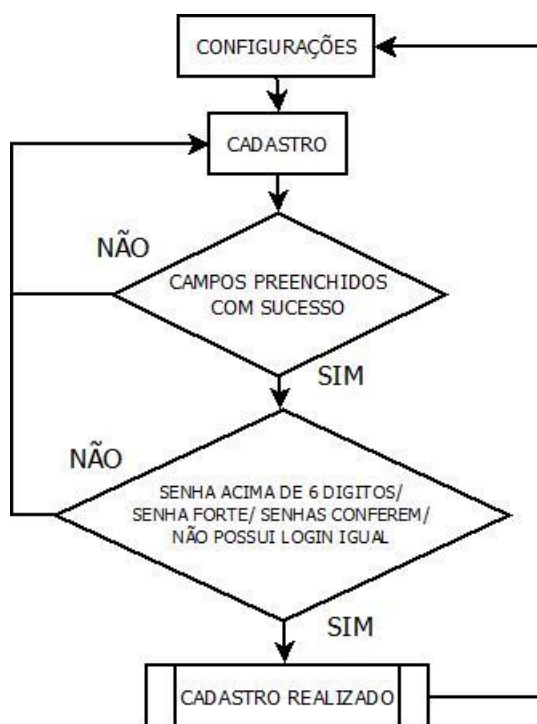
```
// VERIFICA CAMPOS VAZIOS
if ((empty($nome)) or (empty($login)) or (empty($senha)) or (empty($mac)) or (empty($dispositivo))) {
    echo "<meta http-equiv='refresh' content='0; URL=cadastrar.php'>";
    function alerta($mensagem){
        echo '<script>alert("$. $mensagem.")</script>';
    }
    $site = "Por favor, retorne e preencha todos os campos!";
    alerta($site);
    return;
}
```

Fonte: Autor.

Para o cadastro de um usuário no sistema, deve-se observar o atendimento de todos os requisitos solicitados pelo código desenvolvido, para que ele seja executado com sucesso. Um usuário previamente cadastrado deve acessar o servidor e através das configurações solicitar um novo cadastro. Ao preencher os campos fornecidos, executa-se uma verificação automática para conferência se todos os campos foram completados. Em caso positivo, inicia-se a verificação da robustez da senha escolhida, que deve ter no mínimo seis

dígitos. Após esse processo, o cadastro é realizado com sucesso, conforme está demonstrado no fluxograma da Figura 21.

Figura 21 – Fluxograma do cadastro



Fonte: Autor.

3.3.6. Sair

O usuário ao clicar no botão “sair” não conseguirá acessar mais o *site* sem um novo *login*. Foi utilizado o método SESSION do PHP, que funciona quando o usuário faz *login* no *site*, então o mesmo é armazenado na SESSION para possibilitar que o ele tenha acesso as páginas do *site*. Quando o usuário sair do *site*, a SESSION é destruída, impossibilitando que consiga voltar para a página que estava. A Figura 22 demonstra o código que salva o *login* na SESSION para o acesso das outras funcionalidades do sistema pelo usuário.

A vantagem de usar o SESSION, é que não permite que os navegadores (Google Chrome, Mozilla Firefox...) salvem as senhas digitadas. Isso é muito relevante num sistema onde a segurança é essencial, pois, caso o usuário perder o dispositivo *mobile*, a senha não ficará salva.

Figura 22 – Código que salva o *login* na SESSION

```
if (($login == $ln['login']) && ($criptoSenha == $ln['senha']) && ($ln['status'] == "ativo")) {
    $_SESSION['login'] = $ln['login'];
    $_SESSION['senha'] = $ln['senha'];
    $_SESSION[$tentativa] = 0;
    header("Location: ../painel_controle.php");
}
}
```

Fonte: Autor.

A Figura 23 apresenta o fragmento do código do SESSION onde é forçada a destruição da credencial de acesso ao efetuar a saída do sistema.

Figura 23 – Código para destruir a SESSION

```
<?php
if (filter_input(INPUT_GET, "op", FILTER_DEFAULT) == "sair") {
    session_destroy();
    header("Location: index.php");
} |
?>
```

Fonte: Autor.

3.3.7. Criptografia das senhas

Todas as senhas cadastradas no *site* do projeto são criptografadas. A criptografia é necessária para a segurança do sistema e é realizada de um modo que não seja visível no sistema e no banco de dados.

A criptografia é realizada através do método SHA-512, que transforma as senhas digitadas em 128 caracteres alfanuméricos. É uma criptografia de mão única, ou seja, não é possível descriptografar os dados. A Figura 24 mostra o código utilizado para fazer esta criptografia no sistema desenvolvido.

Figura 24 – Criptografia da senha

```
functioncriptoSenha($criptoSenha){
$criptografia = hash('sha512', $criptoSenha);
return ($criptografia);
```

Fonte: Autor.

3.4.HOTSPOT

O *hotspot* é quem vai disponibilizar a rede sem fio local, através do *notebook* ou de um repetidor *Wi-Fi*. Essa rede criada é a que o usuário irá conectar-se para acessar o *site* e ser identificado, como mostra a Figura 25.

Figura 25 – Funcionamento do *hotspot*

Fonte: Autor

O programa utilizado para criar o *hotspot*, foi o *Ap-Hotspot*. É um programa *open source*, ou seja, o código é aberto e pode-se realizar modificações de acordo com as necessidades. Esse *software* cria toda a infraestrutura que é necessária, onde pode-se configurar (Figura 26) a senha de acesso, nome da rede, o canal utilizado, tipo de criptografia, entre outros. No *script* de ativação do *Ap-Hotspot* é colocado o IP que o *hotspot* usará. Este também será o IP do servidor utilizado para acessar o *site* caso o usuário não queira digitar o nome do mesmo.

Figura 26 – Arquivo de configuração do *Ap-Hotspot*

```
# WiFi Hotspot
# Define interface
interface=wlan0
# Select driver
driver=nl80211
# Set access point name
ssid=myhotspot
# Set access point hardware mode to 802.11g
hw_mode=g
# Set WIFI channel (can be easily changed)
channel=6
# Enable WPA2 only (1 for WPA, 2 for WPA2, 3 for WPA + WPA2)
wpa=2
wpa_passphrase=mypassword
#sets wpa key management
wpa_key_mgmt=WPA-PSK
#sets encryption used by WPA
wpa_pairwise=TKIP
#sets encryption used by WPA2
rsn_pairwise=CCMP
```

Fonte: Autor

A Figura 27 demonstra o *script* que ativa o *hotspot*.

Figura 27 – Script para ativar o *Ap-Hotspot*

```
#!/bin/bash
#Start
sudo nmcli nm wifi off

sudo rfkill unblock wlan

#Configure IP address for wlan
sudo ifconfig wlan0 192.168.150.1/24

sudo service hostapd restart

#Start DHCP/DNS server
sudo service dnsmasq restart

#Enable routing
sudo sysctl net.ipv4.ip_forward=1

#Enable NAT
sudo iptables -t nat -A POSTROUTING -o wlan0 -j MASQUERADE

#Run access point daemon
sudo hostapd /etc/hostapd-hotspot.conf
```

Fonte: Autor

3.5. MÓDULO ARDUINO

Para o controle e interação com as funcionalidades de *hardware* da residência, como acender ou apagar luzes, destravar portas e portões e acionar a sirene do alarme, foi utilizado um módulo Arduino® microprocessado como interface de baixo custo. Este Arduino® utilizado no projeto foi o modelo Uno R3 que está sendo demonstrado na Figura 28. Este dispositivo foi selecionado para a integração no sistema por possuir diversas portas de entrada e saída de dados (I/O), que são responsáveis por enviar as respostas dos comandos recebidos pelo *site* aos dispositivos conectados através de relés de dois estados. Estes relés permitem conectar diferentes *hardwares* como atuadores de automação de forma simples e direta, pois constituem uma interface universal para acionamento de dispositivos elétricos. O código desenvolvido para a operação do Arduino® integrado neste sistema de automação foi transferido, através de um *bootloader*, para sua memória interna, permanecendo salva e ativa, podendo até ser desligado da energia, que ao ligar novamente o código estará funcionando normalmente. O código utilizado na página *web* desenvolvida para o acionamento do *hardware* já foi discutido e apresentado anteriormente na Figura 15 deste trabalho, onde pode-se observar que ele lê a opção escolhida pelo usuário e depois envia para o Arduino®.

Figura 28 – Arduino® Uno R3



Fonte: Autor

Na Figura 29 pode-se observar o fragmento do código que foi armazenado no módulo Arduino®, que é responsável por receber e executar todos os comandos referentes à interação com o *hardware* da residência. Suas funcionalidades incluem ligar e deligar as luzes conectadas, abrir ou fechar portas e portões, bem como acionar ou desligar o sistema de alarme. De forma semelhante, este código também é responsável por desligar a luz externa quando passar 30 segundos depois que o usuário desconectou da rede.

Figura 29 – Código armazenado no módulo Arduino®

```
void loop() {  
    char character;  
    character = Serial.read();  
    //ligar LUZ frente  
    if(character == '1'){  
        tempo = millis();  
        digitalWrite(ledPin, HIGH);  
    }  
    //apagar LUZ frente  
    if (millis() > tempo+30000){  
        digitalWrite(ledPin, LOW);  
    }  
    //ligar LUZ 2  
    if (character == '2'){  
        digitalWrite(ledPin2, HIGH);  
    }  
    //apagar LUZ 2  
    if (character == '3'){  
        digitalWrite(ledPin2, LOW);  
    }  
}
```

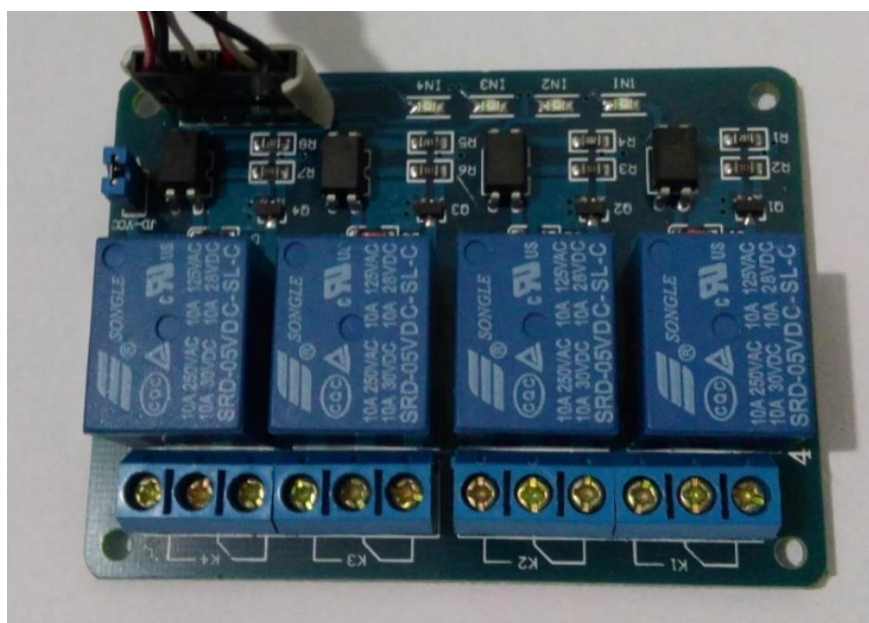
Fonte: Autor

3.5.1. Relé

Por se tratar de um sistema de automação que pode se adaptar as diferentes necessidades de acionamentos de dispositivos elétricos residenciais, foi utilizada nesse projeto uma interface de relés de dois estados, para fazer a interface de corrente mais elevada do Arduino® com os equipamentos, permitindo controla-los ligados à rede de 220V. Foi utilizado um módulo de relés

de 4 canais neste projeto devido à simplicidade de sua operação e segurança contra transientes elétricas, onde cada canal é responsável pelo acionamento de um dispositivo integrado. A Figura 30 apresenta uma imagem do módulo de relés utilizado, onde pode-se observar que cada canal é acionado através de um relé independente.

Figura 30 – Módulo de relés utilizado no projeto



Fonte: Autor

3.6. VERIFICAÇÃO DE USUÁRIOS NA REDE

A verificação de um usuário cadastrado na rede é necessária para acionar automaticamente a iluminação externa da residência quando o mesmo se conecta na rede. Para fazer essa verificação, criou-se um *script* conforme demonstrado na Figura 31, que utiliza a linguagem de programação Shell Script. Esse *script* fica verificando se o endereço *MAC*, do dispositivo que o usuário cadastrou, está conectado à rede. Se estiver conectado, o *script* envia um comando para o Arduino® ligar as luzes externas para que o mesmo possa avaliar a segurança do ambiente e possa prosseguir com as outras autenticações de nível hierárquico mais elevados.

No código que foi armazenado no Arduino® existe uma condição em que se não receber nenhum comando durante 30 segundos, as luzes são desligadas.

O *script* funciona de forma automática, então não existe interação perceptível com o usuário para a identificação positiva e o acionamento da iluminação. Somente é necessário que o usuário deixe habilitado a *Wi-Fi* do seu dispositivo *mobile* para que o *script* o encontre na rede. Quando o *script* o localiza na rede, verifica no banco de dados se o mesmo está cadastrado, e se estiver, o comando para ligar as luzes é enviado para o Arduino®.

Figura 31 – *Script* para verificação automática de um usuário

```
#!/bin/bash

while :
do
    resultado=0
    arp -a | awk '{print$4}' > mac.txt

    while read mac
    do
        resultado=`mysql -u lucas -p*qwe123* --skip-column-names -e
        "select nome from usuario where mac='$mac'" cadastro`
        if [ -n "$resultado" ];
        then
            echo $resultado
            echo -n "1"> /dev/ttyACM0
        fi
        if [ -z "$resultado" ];
        then
            echo "Não encontrado"
        fi
    done < mac.txt
    ip -s neigh flush all
    sleep 10
done
exit
```

Fonte: Autor

4. RESULTADOS E DISCUSSÃO

Neste capítulo estão sendo demonstrados os resultados obtidos através dos ensaios realizados com o sistema em um ambiente de testes. Adicionalmente também estão sendo analisados estes dados obtidos para fornecer subsídios de discussão para eventuais acertos e correções que se tornem necessárias para a plena operação dentro dos requisitos do projeto.

De uma forma ampla, foram testadas as mais diferentes configurações de acesso, autorizados ou não, ao sistema desenvolvido, assim como as diferentes interações com os *hardwares* específicos da residência que se deseja automatizar, como as lâmpadas entre outros. Os resultados satisfatórios obtidos foram utilizados para certificar o sistema, assim como os problemas detectados foram utilizados para realizar as correções necessárias. Neste capítulo são listados os problemas encontrados nos ensaios de operação e as respectivas soluções encontradas para contornar essas falhas e atender ao proposto no projeto original.

4.1. ENSAIOS COM O WEBSITE

Seguindo o procedimento descrito anteriormente, o *website* foi desenvolvido de acordo com os requisitos de operação e funcionalidades definidos nos objetivos deste projeto. Uma das principais premissas para o desenvolvimento deste *website* era a implementação de uma interface simples de utilização, proporcionando uma ferramenta de certificação de usuários cadastrados intuitiva e de rápida operação.

Após o sistema reconhecer o MAC da placa de rede do dispositivo *mobile* do usuário através da entrada na área de cobertura do *hotspot* do sistema, o mesmo autoriza que seja acessada a página *web* de níveis de segurança mais elevados. A primeira tela que aparece para o usuário é a de *login* que está sendo demonstrada na Figura 32. Nesta tela o usuário poderá informar os dados corretos de seu *login* e senha, previamente cadastrados no banco de dados do sistema. Nesta mesma imagem pode-se também observar que já está em funcionamento o modo de acesso HTTPS, onde aparece em vermelho porque o certificado não é reconhecido pelo navegador do Google Chrome.

Neste momento observa-se que o usuário já identificado automaticamente pelo sistema, se autorizado a acessar o painel de controle, pode efetuar apenas três tentativas de certificação erradas antes que o sistema o bloqueie definitivamente para o acesso pela rede *Wi-Fi*. Caso isto ocorra durante esta fase, o sistema irá acionar a sirene do alarme para alertar uma tentativa de interação não certificada, permitindo apenas que essa condição seja revertida através do acesso ao servidor de dados com uma senha específica.

Figura 32 – Tela de *login* da interface web

A imagem mostra a interface de login de um sistema web acessada por um navegador em um dispositivo móvel. No topo, a barra de status do navegador indica o endereço "https://sistema.com", o número "1" em um ícone de aba e o horário "19:09". O título principal da página é "Sistema de Autenticação". Abaixo, há um formulário de login com campos para "Login:" e "Senha:", ambos com caixas de entrada brancas. Um botão azul com o texto "Entrar" em branco está posicionado abaixo dos campos. O fundo da interface é branco.

Fonte: Autor

Após realizar o *login* de forma correta, o usuário tem o acesso a página do painel de controle que está sendo demonstrado na Figura 33, onde poderá escolher entre outras funções, ligar ou desligar as luzes, abrir ou fechar a porta ou portão, ligar ou desligar o alarme, e ainda acessar a página de configurações. Quando uma opção é selecionada pelo usuário, o *status* automaticamente deve informar o estado atual que esta opção selecionada se encontra.

Figura 33 – Tela de apresentação do painel de controle

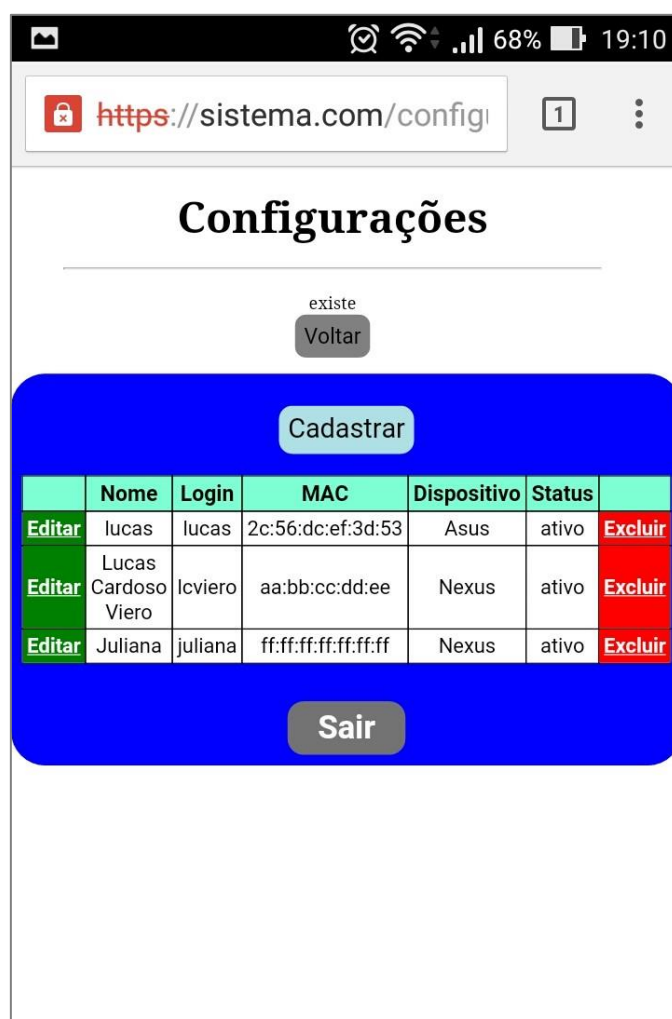


Fonte: Autor

Pode-se observar na imagem que para o botão implementado como “ligar luz” em verde, aparece o *status* de “ligado”, o que indica que esta função já está acionada, e sua solicitação não implicará em uma nova ação no sistema. Não foi implementada nenhuma rotina de impedimento de acionamento dos botões de forma repetitiva, mesmo que o *status* já sinalize que a operação está selecionada, pois não oferece nenhum risco ao sistema, uma vez que simplesmente vai reenviar um comando de ação já executado. Na prática, esse comando repetido apenas enviará um comando para que os níveis de energia em um determinado relé utilizado sejam modificados para o solicitado, o que não irá acontecer uma vez que já se encontram previamente nesta condição.

Pressionando o botão de configuração é possível acessar, utilizando a prévia autorização concedida pelo *login*, a página de configurações do sistema, conforme está sendo demonstrada na Figura 34. Nesta tela serão apresentados os usuários cadastrados juntamente com as suas informações de cadastro, como o nome de usuário, o *login*, o endereço MAC da placa de rede do dispositivo *mobile*, a identificação do nome do dispositivo e o *status* para a utilização do sistema. A única informação que não será apresentada nesta interface é a senha de cada usuário, uma vez que a mesma foi devidamente criptografada para a segurança através de uma via de mão única. Nesta página *web* o usuário poderá editar, excluir ou realizar um novo cadastro de usuário.

Figura 34 – Configurações de usuário na página *web*



Fonte: Autor

Pode-se observar nesta página de configuração dos dados de usuários que as opções de edição, consulta e exclusão são simples e intuitivas, facilitando o uso por qualquer usuário. Seguindo os ensaios para as funcionalidades da página *web* de configurações, encontra-se a interface de cadastro de usuários conforme está sendo demonstrado na Figura 35. Nesta tela um usuário previamente cadastrado no banco de dados poderá informar os dados para realizar um novo cadastro de usuário. É importante lembrar que o sistema não irá aceitar a colocação um nome de usuário existente com novo *login*, assim como uma senha com menos de 6 dígitos ou deixar algum campo do formulário vazio.

Figura 35 – Cadastro de um novo usuário

A imagem mostra a interface de um navegador em um dispositivo móvel. No topo, a barra de status indica o tempo 19:10 e a bateria em 68%. O endereço da página é <https://sistema.com/cadas1>. O título da página é "Sistema de Cadastro". Abaixo do título, há um botão "Voltar". O formulário principal é um retângulo azul com bordas arredondadas, contendo os seguintes campos de texto: "Nome:", "Login:", "Senha:", "Confirme a senha:", "MAC:" e "Dispositivo:". Cada campo tem uma barra de entrada branca. No canto inferior direito do formulário, há um botão "Enviar".

Fonte: Autor

Outras funcionalidades são encontradas na página de editar os dados conforme demonstrado na Figura 36, onde as informações são resgatadas automaticamente do banco de dados. O usuário pode escolher qual informação deseja alterar e informar a senha para concluir a edição. Nesta página que se houver um cadastrado bloqueado, poderá ser realizado o desbloqueio.

Figura 36 – Editar dados de um usuário



Editar

Voltar

Nome: lucas

Login: lucas

Senha:

Confirme a senha:

MAC: 2c:56:dc:ef:3d:53

Dispositivo: Asus

Status: ativo

Enviar

Fonte: Autor

Quando desejar excluir um usuário, um aviso de confirmação será exibido na tela conforme demonstrado na Figura 37, possibilitando ao mesmo cancelar a operação caso desista de realizar a exclusão ou acesse por engano.

Figura 37 – Mensagem de alerta ao excluir um usuário cadastrado

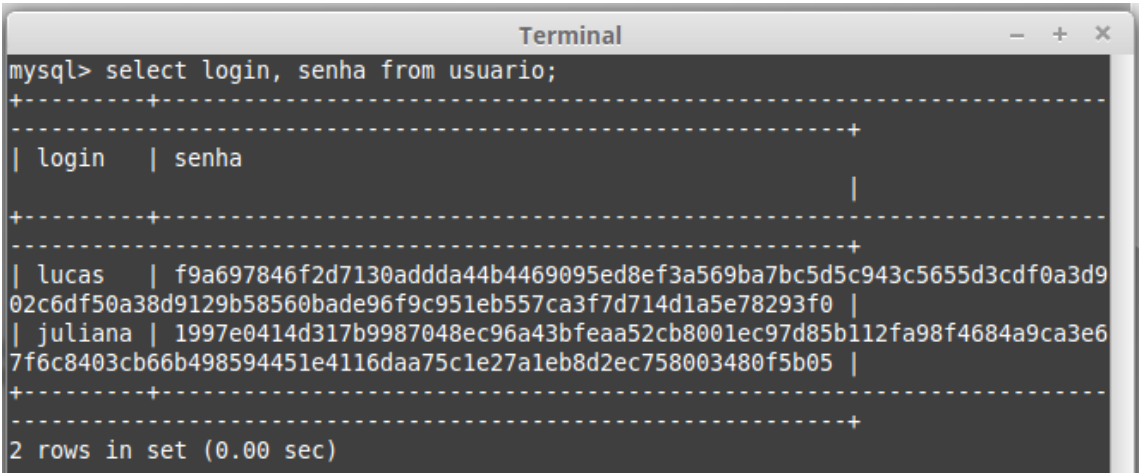


Fonte: Autor

4.2. CRIPTOGRAFIA DE SENHA

Como a segurança de acesso aos níveis de maior hierarquia é realizado através de inserção de senhas, este deve ser um procedimento seguro. A Figura 38 demonstra o banco de dados armazenando uma senha de usuário, onde observa-se que foi utilizado um comando para mostrar somente os campos nome e senha para a verificação da criptografia da mesma, sendo que não é inteligível, e somente o usuário saberá. A senha cadastrada possui 6 dígitos e a criptografia a transformou em 128 caracteres alfanuméricos.

Figura 38 – Senhas criptografadas no sistema



```

mysql> select login, senha from usuario;
+-----+-----+
| login | senha |
+-----+-----+
| lucas | f9a697846f2d7130adda44b4469095ed8ef3a569ba7bc5d5c943c5655d3cdf0a3d902c6df50a38d9129b58560bade96f9c951eb557ca3f7d714d1a5e78293f0 |
| juliana | 1997e0414d317b9987048ec96a43bfeaa52cb8001ec97d85b112fa98f4684a9ca3e67f6c8403cb66b498594451e4116daa75c1e27a1eb8d2ec758003480f5b05 |
+-----+-----+
2 rows in set (0.00 sec)
  
```

Fonte: Autor

4.3. SCRIPT

Para disponibilizar o acesso ao sistema de automação residencial desenvolvido neste projeto, foi determinado que seria montado um *hotspot* de rede sem fio para acesso local do usuário através de um *script* próprio. A Figura 39 demonstra o funcionamento deste *script* do *hotspot* e a rede criada pelo programa *Ap-Hotspot*. No local que está marcado em vermelho na imagem está sendo demonstrado o comando *AP-ENABLE*, e isso quer dizer que a rede está ativada neste momento. Onde está marcado em amarelo, demonstra o momento em que o dispositivo se conecta na rede. O nome desta rede é CASA 32.

Figura 39 – Rede do *hotspot*

```

lucas-270E5J-2570EJ lucas # ./hotspot.sh
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
SIOCSIFFLAGS: Operação não pode ser realizada devido ao RF-kill
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
* Restarting DNS forwarder and DHCP server dnsmasq
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
net.ipv4.ip forward = 1
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
sudo: não foi possível resolver máquina lucas-270E5J-2570EJ
Configuration file: /etc/hostapd-hotspot.conf
Using interface wlan0 with hwaddr 98:83:89:01:8e:91 and ssid "CASA32"
wlan0: interface state UNINITIALIZED->ENABLED
wlan0: AP-ENABLED
wlan0: STA 50:55:27:b2:97:a7 IEEE 802.11: authenticated
wlan0: STA 50:55:27:b2:97:a7 IEEE 802.11: associated (aid 1)
wlan0: AP-STA-CONNECTED 50:55:27:b2:97:a7
wlan0: STA 50:55:27:b2:97:a7 RADIUS: starting accounting session 576192E2-00000000
wlan0: STA 50:55:27:b2:97:a7 WPA: pairwise key handshake completed (RSN)

```

Fonte: Autor

A Figura 40 demonstra o momento em que o *script* responsável por procurar o endereço MAC dos dispositivos *mobile* confirma que o MAC está na rede e no banco de dados. Observa-se que no início o *script* não é encontrado na rede, e que após conectar o dispositivo é realizada a confirmação, e no mesmo momento o comando para ligar a luz é enviado para o Arduino®. O nome do usuário que está relacionado ao endereço MAC também é mostrado.

Figura 40 – *Script* para procurar endereço MAC

```

lucas-270E5J-2570EJ lucas # ./mac.sh
Não encontrado
Nothing to flush.
Não encontrado
Nothing to flush.
Lucas

*** Round 1, deleting 1 entries ***
*** Flush is complete after 1 round ***
Lucas

*** Round 1, deleting 1 entries ***
*** Flush is complete after 1 round ***

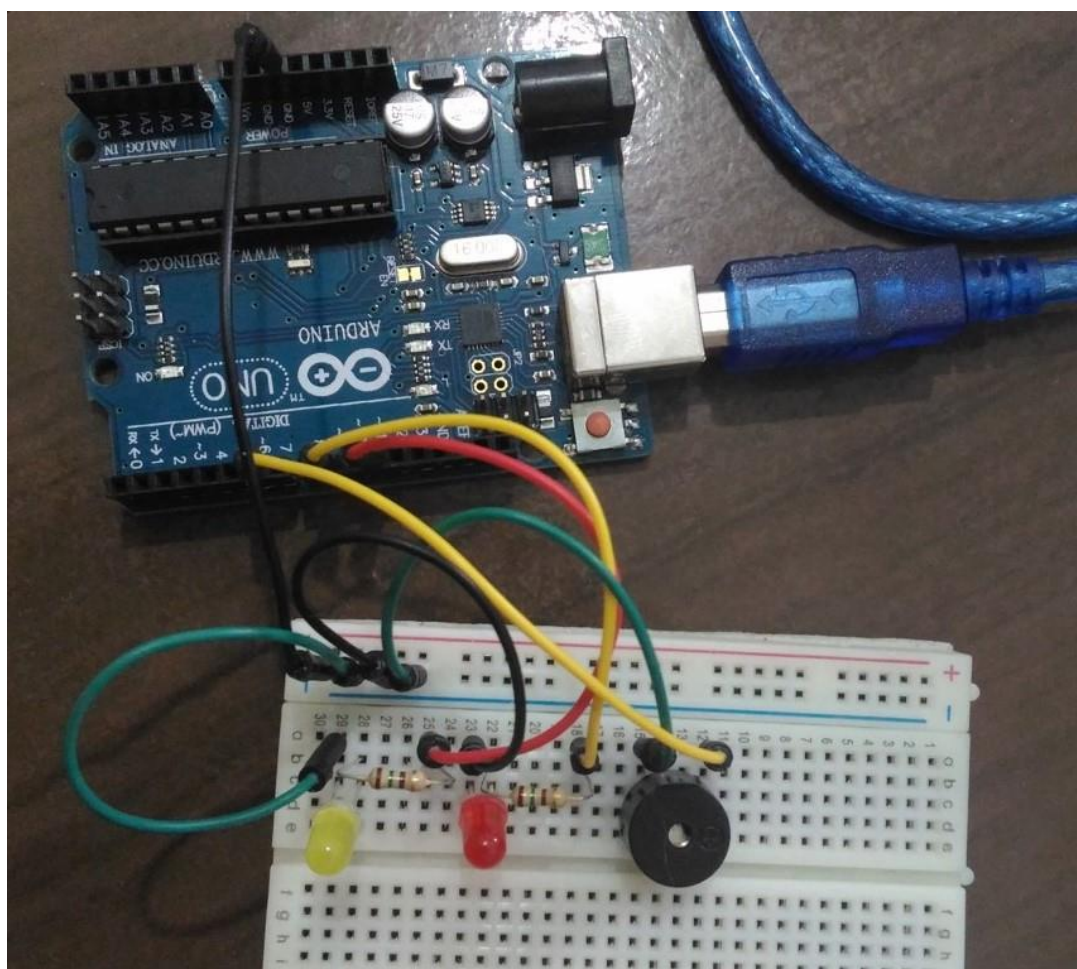
```

Fonte: Autor

4.4. REALIZAÇÃO DOS ENSAIOS

Em um primeiro momento foram realizados diversos ensaio em laboratório para teste de conceito e funcionalidades. Estes ensaios preliminares tiveram como objetivo garantir o controle do sistema e a segurança dos equipamentos, onde foram utilizados o módulo Arduino® sem os dispositivos de alta corrente. Adicionalmente foi utilizada uma *protoboard* para simular em pequena escala os dispositivos, conforme demonstrado na Figura 41. Foram utilizados dois *leds*, um *buzzer* e dois resistores de 150 ohms. O *led* vermelho foi utilizado para ligar quando o MAC do dispositivo fosse reconhecido na rede, o *led* amarelo simulou a porta de entrada da residência e o *buzzer* o acionamento do alarme quando o *login* fosse bloqueado. Os testes preliminares ocorreram com sucesso.

Figura 41 – Ensaio em laboratório com *protoboard*

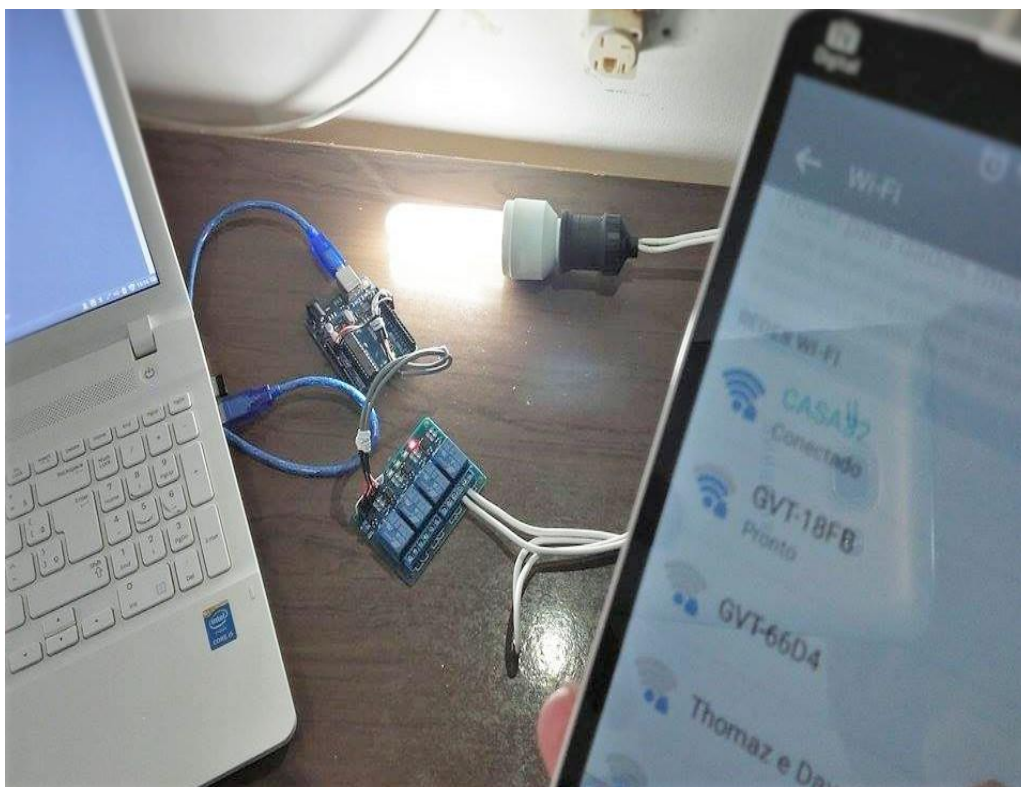


Fonte: Autor

4.4.1. Ensaio Final de Operação

Uma vez que os ensaios em laboratório aprovaram o sistema para operação, foram realizados os demais testes finais com dispositivos elétricos de maior corrente. Nestes ensaios foram utilizadas lâmpadas ligadas na energia elétrica e os próprios relés de acionamento. A Figura 42 demonstra o funcionamento do sistema do momento exato em que o dispositivo reconhece automaticamente o MAC da placa de rede de um dispositivo *mobile* de um usuário cadastrado e a lâmpada externa da residência é ligada.

Figura 42 – Ensaio finais com dispositivos elétricos de maior corrente



Fonte: Autor

Como no primeiro ensaio realizado no laboratório o *led* e o *buzzer* simularam a porta e o alarme sem problemas, foram realizados os testes finais com o acionamento de lâmpadas e dos relés de placa de controle. Quando a porta e o alarme foram acionados, os relés foram ativados e desativados para a confirmação do funcionamento dentro do esperado.

4.5. PROBLEMAS ENCONTRADOS

Durante a realização dos ensaios foram encontrados alguns problemas e falhas que estão sendo neste Capítulo, assim como as possíveis soluções e correções. Em geral, esses pequenos problemas e desvios da operação ideal não chegaram a comprometer de forma definitiva o funcionamento do projeto.

4.5.1. Diferentes Resoluções de Telas

O primeiro problema encontrado nos ensaios foi na aparência do *site* nos dispositivos *mobile*, que conforme o tamanho da tela e sua resolução nativa, influenciava na apresentação gráfica do painel de controle desenvolvido, alterando suas dimensões originais e deixando o *site* deformado.

A criação do *site* foi realizada utilizando como teste um *smartphone* com a tela de 5,5 polegadas. A Figura 43 mostra o *site* em um *smartphone* com a tela de 5 polegadas, onde percebe-se a deformação na página de painel de controle.

Figura 43 – Deformação da apresentação gráfica em telas diferentes



Fonte: Autor

Adicionalmente se verificou neste caso que dependendo do navegador utilizado no ambiente *mobile* também poderia gerar problemas de apresentação, percebendo-se algumas deformações acentuadas na configuração original. Tentou-se verificar uma solução definitiva para o problema em fóruns de desenvolvedores, mas devido à falta de tempo, experiência e conhecimento do autor com o desenvolvimento de *website* para visualização em *smartphones*, não foi possível encontrar uma solução para o problema em tempo.

4.5.2. Número de Tentativas de Acesso para Bloqueio

Da mesma forma que o anterior, durante a fase de testes com o sistema foi encontrado um segundo problema de operação. Após realizar o teste para bloquear o usuário depois da terceira tentativa de *login* errada, verificou-se que o bloqueio inexplicavelmente passou a ser realizado após a quinta tentativa errada, o que não foi proposto originalmente por este projeto. Após realizar várias alterações no código fonte do *site* e configurações das interfaces, não se conseguiu encontrar a solução do problema através da bibliografia disponível. Embora este erro não impeça o pleno funcionamento do sistema, esta configuração não está em acordo com o projeto original e continua sendo avaliada para futuras correções.

4.5.3. Captura do Endereço MAC dos Dispositivos Mobile

Em outro ensaio realizado, um terceiro problema foi detectado. O problema foi identificado como sendo no comando, marcado em vermelho na Figura 44. Ele é utilizado para capturar o endereço MAC dos dispositivos que estão na rede do *hotspot*. Quando este comando é executado logo após a inicialização do sistema, a ação demora em torno de 10 segundos para ser efetivada. Isso significa na prática que se o usuário estiver chegando na residência, o seu dispositivo irá ser reconhecido e a luz externa de segurança acionada após 10 segundos depois de ter sido reconhecido na rede. A partir da segunda identificação, o sistema opera de forma normal.

Pensou-se inicialmente que o problema poderia estar no *script* desenvolvido, que estaria causando essa demora. Porém realizou-se um teste

somente com o comando e os 10 segundos para a primeira ação ser executada se mantiveram. Diversas bibliografias apontam para o *hardware* do próprio Arduino® utilizado no projeto, que devido a flutuações de suas condições de operação na inicialização podem causar demora ou incerteza nas respostas iniciais. Não foi encontrado nenhuma solução para o problema até o momento.

Figura 44 – Comando executado no reconhecimento do usuário

```
#!/bin/bash

while :
do
  resultado=0
  arp -a | awk '{print$4}' > mac.txt

  while read mac
  do
    resultado=`mysql -u lucas -p*qwe123* --skip-column-names -e
    "select nome from usuario where mac='$mac'" cadastro`
    if [ -n "$resultado" ];
    then
      echo $resultado
      echo -n "1"> /dev/ttyACM0
    fi
    if [ -z "$resultado" ];
    then
      echo "Não encontrado"
    fi
  done < mac.txt
  ip -s neigh flush all
  sleep 10
done
exit
```

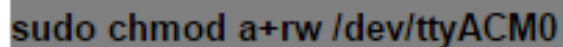
Fonte: Autor

4.5.4. Envio de Dados para o Arduino

Um quarto problema foi verificado foi no momento de enviar os comandos do *site* para o Arduino®. Neste caso específico, o módulo Arduino® não estava recebendo os comandos para executá-los. Procurou-se a solução deste problema na bibliografia e fóruns *online*, onde se verificou que seria necessário alterar as permissões de leitura e gravação da porta serial que o Arduino® utiliza

no Linux. A solução foi encontrada no *site* Laboratório de Garagem. O comando que foi utilizado está sendo demonstrado na Figura 45.,

Figura 45 – Comando para habilitar porta serial no Linux



```
sudo chmod a+rw /dev/ttyACM0
```

Fonte: <http://labdegaragem.com/forum/topics/apos-atualiza-o-do-ubuntu-para-o-12-04-a-ide-do-arduino-n-o>

Após executar o comando no Linux, o Arduino® respondeu aos comandos que eram enviados ao mesmo. O comando tem que ser executado após conectá-lo no computador.

5. CONCLUSÃO

A automação de processos da rotina do dia a dia é uma forma moderna de oferecer praticidade, conforto e segurança para os usuários. Em geral, produtos comerciais de automação residencial necessitam de um investimento elevado para serem implementados, da mesma forma que nem todos são simples e intuitivos para a utilização.

Neste projeto foi proposto o desenvolvimento de um sistema de automação residencial de baixo custo, que opere através de recursos já existentes no próprio ambiente, e que ao mesmo tempo ofereça a praticidade e a segurança desejada para esta aplicação. Para a maior segurança contra invasões de *hackers* e *crackers* ao sistema, o mesmo foi disponibilizado a partir somente do acesso local, via um *hotspot* de rede sem fio especialmente desenvolvido para este projeto. Adicionalmente também foi implementada uma entrada de credenciais de acesso ao sistema que possui níveis de interatividade e segurança diferentes, dependentes das ações que podem ser realizadas.

Como nível hierárquico mais baixo, foi utilizado o reconhecimento automático do usuário quando o mesmo entra na área de cobertura do *hotspot* da residência. Essa identificação é realizada através do endereço MAC da placa de rede do dispositivo *mobile* que está acessando o sistema em modo local. Foram testadas diversas possibilidades de identificação nos experimentos e o sistema respondeu conforme esperado, proporcionando o acesso aos próximos níveis de interação com a residência através de novas credenciais mais fortes, assim como acendeu a iluminação externa para avaliação do perímetro.

Nos níveis mais altos de credenciais de usuários autorizados, o sistema foi amplamente testado e também apresentou todos os resultados satisfatórios. Quando o usuário estava devidamente cadastrado e autorizado a realizar as operações de automação com a residência, não ocorreram problemas que afetassem o funcionamento do mesmo, permitindo as alterações de cadastros de usuários, solicitações para realizar ações a serem executadas nos *hardwares* da casa e bloqueio de usuários não autorizados, com a devida sinalização de alerta de possível tentativa de invasão.

Durante o desenvolvimento notou-se que a principal dificuldade foi relativa a criação do *website* utilizado para o controle do sistema, pois o autor não

possuía conhecimentos suficientes na linguagem PHP. Estas limitações, porém, não interferiram na obtenção dos resultados propostos no projeto, e a implementação do *site* foi concluída conforme o esperado, assim como todas as funcionalidades estão de acordo com o que foi proposto originalmente. Os problemas encontrados e que foram relatados anteriormente não influenciam de forma impactante nos resultados finais do projeto.

De forma resumida, conclui-se que através dos resultados obtidos nos ensaios realizados, tanto com as simulações de operação com o *protoboard* assim como nos ensaios finais, ocorreram dentro do esperado e com sucesso, e desta forma o projeto teve seu objetivo atendido.

5.1. TRABALHOS FUTUROS

Como sugestões para trabalhos futuros está sendo recomendado o desenvolvimento de uma nova versão de *website*, sendo que esta deva ser compatível com qualquer navegador comercial e dispositivo móvel. Adicionalmente, também pode-se sugerir o desenvolvimento de uma ferramenta de *feedback* em tempo real para a confirmação de execução dos comandos depois que o Arduino® receber os mesmos e conseguir realizar a ação de forma satisfatória. Da mesma forma, pode-se desenvolver uma interface, no modelo do painel de controle apresentado, onde o próprio usuário possa implementar novas funcionalidades ao projeto, de acordo com suas necessidades na residência.

REFERÊNCIAS BIBLIOGRÁFICAS

APACHE. **About apache.** 2016. Disponível em: <https://httpd.apache.org/ABOUT_APACHE.html/>. Acesso em: 26 mar. 2016.

ARDUINO. **What is Arduino?** 2016. Disponível em: <<https://www.arduino.cc/en/Guide/Introduction/>>. Acesso em: 07 abr. 2016.

CASTRUCCI, P.; MORAES C. C. de. **Engenharia de Automação Industrial**. 2 ed. Editora LTC, 2007.

HOAG, M. **Servidores web usando o apache**. São Paulo: Editora Berkeley Brasil, 2002. 496 p.

INSTITUTO NEWTON C. BRAGA. **Tudo sobre relés**. 2014. Disponível em: <<http://www.newtoncbraga.com.br/index.php/como-funciona/597-comofuncionam-os-reles>>. Acesso em: 10 abr. 2016.

KUROSE, J. F.; ROSS, K. W. **Redes de computadores e a internet: uma abordagem top-down**. 3 ed. São Paulo: Pearson Addison Wesley, 2006. 625 p.

MCROBERTS, M. **Arduino básico**. São Paulo: Editora Novatec, 2011. 453p.

MILANI, A. **MySQL - Guia do programador**. São Paulo: Editora Novatec, 2006. 397p.

MOZZILA DEVELOPER NETWORK. **Introdução ao HTML**. 2014. Disponível em: <<https://developer.mozilla.org/pt-BR/docs/HTML/Introduction/>>. Acesso em: 26 mar. 2016.

MOZZILA DEVELOPER NETWORK. **What is a web server?**. 2016. Disponível em: <https://developer.mozilla.org/en-US/Learn/Common_questions/What_is_a_web_server/>. Acesso em: 21 mar. 2016.

MYSQL. **História do MySQL**. Disponível em: <<https://dev.mysql.com/doc/refman/5.7/en/history.html/>>. Acesso em: 01 abr. 2016.

NETCRAFT. **March 2016 Web Server Survey**. 2016 Disponível em: <<http://news.netcraft.com/archives/2016/03/18/march-2016-web-server-survey.html/>>. Acesso em: 27 mar. 2016.

ORACLE. **Oracle Announces General Availability of MySQL 5.7**. Disponível em: <<https://www.oracle.com/corporate/pressrelease/mysql-5-7-ga-101915.html/>>. Acesso em: 03 abr. 2016.

PHP. **História do PHP**. 2016a.. Disponível em: <http://php.net/manual/pt_BR/history.php.php/>. Acesso em: 30 mar. 2016.

PHP. **O que o PHP pode fazer?**. 2016b. Disponível em: <https://secure.php.net/manual/pt_BR/intro-whatcando.php/>. Acesso em: 30 mar. 2016.

RAMALHO, J. A. **Iniciando em HTML**. São Paulo: Editora Makron Books do Brasil, 1996. 220p.

SANTOS, D. M. dos. **Relê**. 2016. Disponível em: <<http://www.infoescola.com/electronica/rele/>>. Acesso em: 10 abr. 2016.

SERPRO. **Planejamento, Instalação e Configuração do Host On-Demand**. 2009. Disponível em: <<http://acesso.serpro.gov.br/hod/pt/doc/install/install.html#hodpreface/>>. Acesso em: 05 abr. 2016.

SILBERSCHATZ, A.; KORTH, H. F.; SUDARSHAN, S.; **Sistema de banco de dados**. Rio de Janeiro: Editora Elsevier, 2006. 781p.

STALLINGS, W. **Criptografia e segurança de redes**. 4 ed. São Paulo: Editora Pearson Prentice Hall, 2008. 476p.

TANENBAUM, A. S. **Redes de Computadores**. 4 ed. Editora Campus Elsevier, 2003. 968 p.

VERT. **O futuro da rede Wi-fi**. 2016. Disponível em: <<http://www.vert.com.br/blog-vert/o-futuro-da-rede-wi-fi/>>. Acesso em: 17 mar. 2016.

W3C. **Visão geral do HTML5**. 2010. Disponível em: <<http://www.w3c.br/cursos/html5/conteudo/capitulo1.html/>>. Acesso em: 29 mar. 2016.