UNIVERSIDADE FEDERAL DO PARÁ
CAMPUS UNIVERSITÁRIO DE TUCURUÍ
FACULDADE DE ENGENHARIA DA COMPUTAÇÃO

DÁRLISON SOUZA DE ALENCAR

IMPLEMENTAÇÃO DE UM PROTÓTIPO DE AUTOMAÇÃO RESIDENCIAL VOLTADO PARA SEGURANÇA

TUCURUÍ
2019

DÁRLISON SOUZA DE ALENCAR

IMPLEMENTAÇÃO DE UM PROTÓTIPO DE AUTOMAÇÃO RESIDENCIAL VOLTADO PARA SEGURANÇA

Trabalho de Conclusão de Curso apresentado à Faculdade de Engenharia de Computação, do Campus Universitário de Tucuruí, da Universidade Federal do Pará, como requisito parcial para obtenção do título de Bacharel em Engenharia de Computação.

Orientador: Prof. MSc. Renato Luz Cavalcante

TUCURUÍ
2019

Dedico este trabalho aos meus pais pelo extraordinário esforço que fizeram desde sempre para que eu pudesse realizar com sucesso cada passo nessa jornada, contribuindo com todo o amor e torcida por cada objetivo por mim alcançado.

AGRADECIMENTOS

A Deus, meu fiel orientador e sustentador em todos os momentos de minha vida.

Aos meus pais, que contribuíram sem medir esforços para formação do meu caráter e fizeram de mim quem sou e pela força, perseverança paciência, e à minha família, agradeço todo o amor, carinho, compreensão e respeito.

Ao meu orientador Prof. MSc. Renato Luz Cavalcante, por ter me orientado e participado com atenção, me ajudando a conduzir este trabalho.

Aos meus amigos de curso, pela inestimável amizade que formamos no decorrer destes 5 anos e pela colaboração dedicada e despretensiosa.

Aos meus colegas do campus que contribuíram de forma prestigiosa no desenvolvimento de um hobby pessoal, pelo apoio e pela presteza com que me atenderam, sempre que solicitadas.

Aos meus amigos Núria Carolline, Leonardo Costa, Gustavo Henrique, Artur Cunha e Valdir Santos, agradeço pela contribuição direta na realização deste Trabalho de Conclusão de Curso.

E a todos que contribuíram direta e indiretamente para a realização deste sonho.

RESUMO

A evolução tecnológica promove o desenvolvimento de sistemas de segurança mais eficazes, uma vez que meios tradicionais, como vigilantes ou empresas de segurança, estão sujeitos a falhas. O desenvolvimento deste sistema gera maior segurança e eficácia. O presente trabalho se refere ao desenvolvimento de um sistema de monitoramento e controle de acesso residencial de baixo custo, que fará a vigilância de movimentos e status de portas e janelas de uma casa, sinalizado alterações no sistema. Para o desenvolvimento do sistema foram utilizados sensores de presença, sensores de fim de curso, microcontrolador (nodeMCU ESP32), teclado capacitivo, display LCD, plataforma de desenvolvimento Arduino, plataforma de desenvolvimento Android Studio, um banco de dados e um aplicativo mobile. O sistema proposto faz a detecção de movimentos por meio de uma rede de sensores instalados em uma maquete de uma residência, os dados obtidos pelos sensores são enviados para o microcontrolador que é responsável pelo processamento das informações e posteriormente armazenados em um banco de dados conectado ao aplicativo mobile que é responsável pela comunicação com usuário. O sistema apresenta resultados satisfatórios, com respostas aproximas as de tempo real.

PALAVRAS-CHAVE: Sistema de Segurança, Integridade de Sistema, Comunicação Remota.

ABSTRACT

Technological developments promote the development of more difficult security systems, as traditional methods such as security guards or security companies are subject to failure. The development of this system generates greater safety and effectiveness. The present work refers to the development of a low cost residential access control and monitoring system, which shows signs of movement and status of doors and windows of a home, signaled system changes. For the development of the system, presence sensors, limit switches, microcontroller (nodeMCU ESP32), capacitive keyboard, LCD display, Arduino development platform, Android Studio development platform, a database and a mobile application were used. The adopted system detects movements through a network of sensors installed in a residence module, the data provided by the sensors is sent to the microcontroller that is responsible for processing the information and later used in a connected database. To the mobile app that is responsible for communicating with the user. The system presents satisfactory results, with approximate responses as real time.

KEYWORDS: Security System, System Integrity, Remote Communication.

LISTA DE ILUSTRAÇÕES

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

C.C.	Corrente Contínua
I.D.E	*Integrated Development Environment* (Ambiente desenvolvimento integrado)
P.W.M	*Pulse Width Modulation* (Modulação por largura de pulso)
G.P.I.O	Entrada/Saída de uso geral
I.P	Protocolo de Internet, número dado à algum dispositivo conectado à rede
S.S.I.D	Identificador do conjunto de serviços, é o nome dado à rede WiFi
U.S.B	Barramento Serial Universal
opensource	Código Aberto
firmware	Instruções operacionais programadas em um hardware
A.D.T	*Android Developer Tools*
A.S	Android Studio
JSON	*JavaScript Object Notation*

SUMÁRIO

1 INTRODUÇÃO

A violência nos centros urbanos vem crescendo nas últimas décadas, esse crescimento está diretamente ligado a diversos fatores, como o crescimento desenfreado da zona urbana, os grandes índices de desemprego. Com índices de violência altos crimes como assaltos e invasões de casa também ocorrem com maior frequência. Deste modo a população começa preocupar-se mais com a segurança. Segundo o Instituto de Pesquisa Econômica Aplicada (IPEA, 2019), os assaltos a residências no Brasil tiveram um aumento de cerca de 25% nos últimos 4 anos.

O aumento da violência influência diretamente no crescimento da busca de meios para vigilância residencial. Com avanços nas últimas décadas a automação residencial começa a se torna viável e de baixo custo se comparado com anos anteriores. Com a diminuição dos custos há o aumento de busca por sistemas de segurança. Isto implica diretamente na implementação sistemas automáticos mais completos e com baixos índices de falhas.

Segundo a Associação Brasileira de Automação Residencial (AURESIDE, 2016), cerca de 300 mil residências brasileiras possuem automação. Porém, esse número é considerado muito baixo em relação à quantidade de residências existentes. Um dos principais motivos é o alto custo e o desconhecimento das pessoas pelo assunto. Logo, o aumento do número de residências automatizadas passa a ser um desafio para os desenvolvedores de novas tecnologias. Na tabela 1 é apresentado a evolução da adoção das principais tecnologias nas residências.

Tabela 1-Evolução da adoção de algumas tecnologias

TECNOLOGIA	2003	2004	2005	2006	2015*
Cabeamento Estruturado	42%	61%	49%	53%	80%
Monitoramento de Segurança	18%	28%	29%	32%	81%
Controle de Iluminação	1%	2%	6%	8%	75%
Automação Integrada	0	2%	6%	6%	70%
Gerenciamento de Energia	1%	5%	11%	11%	62%

Fonte: NHAB, 2015

1.1 Descrição do Problema

O Brasil enfrenta grandes problemas de violência e insegurança, os índices de violência se tornam alarmantes, segundo dados oficiais do Atlas da Violência-IPEA em 2017 teve o maior nível histórico o de letalidade violenta intencional no país. A violência está presente principalmente em centros urbanos, com estes números tão elevados nem mesmo as residências podem ser consideradas seguras. Deste modo a maioria das residências estão vulneráveis a invasões e roubos (IPEA, 2019).

Os sistemas de segurança podem ter um custo elevado dependendo do modelo de sistema a ser implantado, além da necessidade de técnicos para instalação e taxas mensais, essas características podem dificultar o acesso da maioria da população a esses dispositivos de segurança.

1.2 Objetivos

Este trabalho tem como objetivo o desenvolvimento, implementação e teste de um sistema de segurança e monitoramento residencial de baixo custo e fácil uso. O mesmo poderá ser controlado remotamente através de um aplicativo móvel. O projeto consiste em uma em uma rede de sensores integrada a um microcontrolador e sistema de controle.

Além disso, tornam-se objetivos secundários a elaboração do um protótipo de uma rede de sensores, dispositivos de entrada e saída conectados a um microcontrolador, desenvolvimento de um software para integração e comunicação entre microcontrolador e componentes eletrônicos. Desenvolvimento de um aplicativo móvel para controle e resposta do sistema. A etapa final é a comunicação entre o aplicativo móvel e o microcontrolador, assim obtendo o sistema completo.

1.3 Organização do Trabalho

Este trabalho possui cinco capítulos, o primeiro capítulo possui informações referentes à introdução, considerações sobre o tema e problemas foram abordados, assim como o objetivo.

O segundo capítulo apresenta o embasamento teórico, que abrange pontos importantes sobre domótica, softwares e componentes usados, além de suas características.

O terceiro capítulo descreve a metodologia do trabalho proposto, o desenvolvimento e o descritivo de como o sistema vai atuar. Detalhes a construção e da implementação do projeto.

O quarto capítulo entra a análise do código e aplicação prática do modelo proposto. Detalhes dos resultados que o sistema apresentou nos testes, além do custo dos componentes que foram utilizados do sistema.

Logo, no capítulo cinco possui as considerações finais deste trabalho. Um apanhado geral dos momentos relatados aqui na implementação deste sistema. Ainda, com sugestões de trabalhos futuros que possam vir a enriquecer o modelo proposto.

2 REFERÊNCIAL TEÓRICO

2.1 Domótica

O termo domótica deriva da junção da palavra em latim *domus*, que significa casa e a palavra robótica, ou seja, domótica torna possível controlar de forma automática uma residência. (FERREIRA, 2008). A Domótica pode receber outras denominações sinônimas, tais como, Automação Predial, Automação Residencial, Automação Doméstica, entre outros.

O conceito de domótica segundo Bolzani:

> É a ciência moderna de engenharia das instalações em sistemas prediais. A Domótica é uma ciência multidisciplinar que estuda a relação entre homem e casa. A imersão de pessoas em ambientes computacionalmente ativos revelou a necessidade do uso de técnicas mais sutis que gerenciassem a complexidade e o dinamismo das interações dos moderadores com o ambiente residencial saturado de diminutos dispositivos eletrônicos interligados a rede. (BOLZANI, 2010).

A automação residencial, assim como a predial, é derivada da automação industrial, porém, com tecnologias adequadas para a realidade de uma residência, onde na maioria das vezes não há espaço suficiente para grandes centrais de controle e pesados sistemas de cabeamento (SENA, 2005).

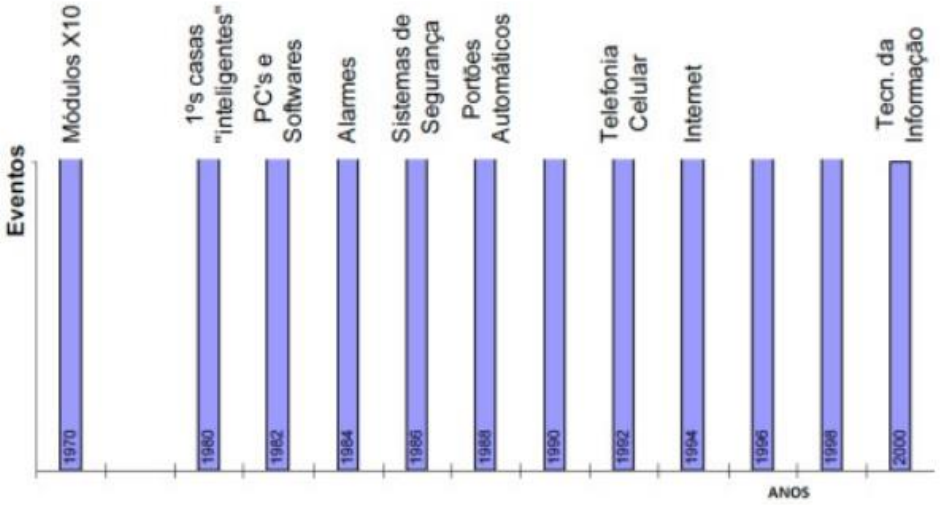
2.1.1 História da Domótica

Segundo Moya e Tejedo (2010), a origem da domótica remete-se aos anos 70, quando surgiram os primeiros dispositivos de automação de edifícios, baseados na tecnologia X-10. O X-10 é um protocolo que permite controlar dispositivos em uma rede elétrica já existente, evitando a necessidade de novos cabeamentos. É um sistema de fácil instalação, mas bastante instável, uma vez os componentes podem vir a falhar, ou até mesmo estragar devido à falta de energia ou até mesmo por descargas eletromagnéticas.

Nos anos 80, com o advento da computação pessoal, surgiram interfaces gráficas e operações muito mais simples que as já existentes, o que levou ao surgimento de novas possibilidades de automação residencial. Mas foi no final da década de 90 que uma vasta gama de novidades surgiu, incorporadas com os telefones celulares e a web.

Atualmente a domótica utiliza esses novos recursos de forma integrada, como por exemplo, em sistemas de controle e monitoramento móvel, através de celulares, tablets, ou via web (OLIVEIRA, 2012). Na figura 2.1 é mostrado os principais marcos da domótica até o ano 2000 segundo AURESIDE (Associação Brasileira de Automação Residencial e Predial).

Figura 1 - Marcos da Domótica



Fonte: AURESIDE,2003.

2.1.2 Características de sistemas de domótica

No mercado já existem diversas opções de sistemas de domótica, alguns mais simples que executam funções básicas como, por exemplo, controle de iluminação e controle de portas e portões, outros já permitem controlar até mesmo persianas, sons ambientes, temperatura da água da piscina e da banheira, entre outras diversas funcionalidades.

Entretanto, segundo a AURESIDE, algumas características são essenciais a qualquer sistema, como por exemplo:

- Capacidade de integrar todos os sistemas: os sistemas devem ser interligados através de uma rede, e permitir controle através de uma única interface;
- Atuação em condições variadas: o sistema deve ser capaz de atuar em condições adversas, como interrupções de energia, climas diversificados entre outros;
- Fácil relação com o usuário: os usuários muitas vezes não compreendem programações complexas, portanto, deve haver um sistema com interface intuitiva;

- Monitoramento: o monitoramento desse tipo de sistema é algo crítico, portanto devem ser realizadas auditorias com determinada frequência, e sempre observar relatórios de controle.

2.1.3 Segurança Patrimonial (Sistemas Domóticos)

Segundo Bolzani (2004), para se realizar um bom projeto de segurança patrimonial em uma residência é imprescindível criar soluções que sejam não somente compatíveis, mas também complementares, além de cumprir fundamentalmente os seguintes pontos, considerados básicos em um sistema com tal propósito:

- Prevenção ou dissuasão – utilização de sistemas que inibam e promovam a desistência da ação de intrusão;
- Detecção e alarmes – utilização de sistemas que permitam a detecção de ações de intrusão e possibilitem o acionamento de diversos tipos de alarmes;
- Reconhecimento ou identificação – a residência inteligente deve ser capaz de tomar decisões e cumprir processos baseados no reconhecimento e identificação do usuário;
- Reação – o sistema deve realizar as funções de reação, disparando ações contra o processo de intrusão.

Na central de controle, o tratamento dos sensores proporciona informações claras e objetivas ao usuário, tanto do estado das instalações como também dos eventos que vão se produzindo. O software deve ser capaz de prever várias situações de ataque e reação a fim de nunca deixar o usuário em posição de risco. No caso da ausência de pessoas no local, o sistema deve estar apto a informar o usuário remotamente por meio de mensagens de alerta via rede telefônica, rede de acesso ou outro sistema de conexão com a polícia ou uma agência de segurança particular.

Neste sentido, o desenvolvimento de tais sistemas permite a representação, na tela de um PC remoto (o PC do usuário em seu local de trabalho ou o PC da agência de segurança particular), de gráficos que indicam a situação, em planta, das partes afetadas, bem como as possíveis imagens das câmeras de vigilância. O sistema ainda pode dar as instruções básicas para situações de emergência. Finalmente, o sistema centralizado de segurança deve atuar em conjunto com os outros que controlam os demais serviços residenciais, tanto para receber informações como para dar ordens (controlando as luzes e outros subsistemas, consegue-se criar cenas de simulação de presença).

Os sistemas que são empregados na proteção de espaços e edificações, controlados por meio de uma central, podem ser divididos basicamente em cinco subsistemas (Bolzani, 2004):

- Subsistema de detecção perimetral: pretende-se detectar a intrusão ou evasão de qualquer indivíduo pela cerca perimetral no menor intervalo de tempo possível, com exatidão do ponto, auxiliando o trabalho de um possível corpo de vigilância. É importante que na fase de projeto exista a preocupação com barreiras para evitar os falsos alarmes. Os sensores mais utilizados na detecção perimetral são o sensor infravermelho ativo, sensor de microondas e o sensor sísmico;
- Subsistema de sensoriamento interno: é o sistema que supervisiona as áreas internas. Atua normalmente associado aos sistemas de controle de acesso e CFTV (Circuito Fechado de Televisão), enviando um sinal de alarme para a central quando da ocorrência de alguma anormalidade. Os dispositivos mais utilizados para o sensoriamento interno são o sensor magnético de abertura, sensor de vibração, sensor acústico, sensor infravermelho passivo, sensor microondas e botões de pânico;
- Subsistema de circuito fechado de televisão: o sistema de CFTV atende às áreas críticas do ponto de vista da segurança, isto é, portão de entrada, corredores, acessos a áreas restritas, garagens e outras áreas, sendo cada local supervisionado por uma ou mais câmeras, com lentes adaptadas para cada situação;
- Subsistema de controle de acesso: permite ou não a entrada de pessoas. Crachás, cartões, senhas e sistemas biométricos são exemplos de sistemas de controle de acesso. Existe um item deste capítulo especialmente dedicado a este assunto;
- Subsistema de controle de rondas: o efetivo controle da movimentação do pessoal da segurança deve ser cuidadosamente planejado para que não ocorram erros ou omissões possibilitando a intrusão de estranhos.

2.1.4 Benefícios proporcionados por sistemas de domótica

Os benefícios que podem ser observados imediatamente pelos moradores de uma casa automatizada são (LAGUÁRDIA, 2015):

- Economia de energia: a energia é utilizada somente quando necessário, pois o controle da intensidade de iluminação, sensores de presença, controle da temperatura ambiente, elimina gastos desnecessários;
- Conforto: ajuste de temperatura de piscinas, filtros de ar, ar condicionado, entre outros equipamentos através de uma única interface;
- Conveniência: a temperatura do ambiente pode ser controlada mesmo antes da chegada dos moradores, lâmpadas e portões podem ser controlados de qualquer local;
- Acessibilidade: sistemas de automação com dispositivos touchpad e com reconhecimento de voz, proporcionam a portadores de necessidades especiais, a possibilidade de controlar o ambiente que estão, seja ascendendo uma lâmpada, controlando luminárias, televisores, portas, entre outros, devolvendo ao indivíduo sua independência;
- Segurança: a possibilidade de controle de luminosidade, ar condicionado, televisores e outros dispositivos, podem fazer que a residência pareça sempre ocupada.

Câmeras de monitoramento podem ser acessadas de qualquer local, o que permite que a casa esteja sempre monitorada, um exemplo citado por LAGUÁRDIA (2015), foi o caso do empresário Ronan Soares. O empresário estava na cidade de Colônia, na Alemanha, ao verificar a câmera de monitoramento de sua residência, percebeu que havia ladrões invadindo o local, rapidamente avisou a esposa que estava no Brasil, que praticamente no mesmo instante acionou a polícia.

2.1.5 Potencial do mercado de Domótica no Brasil

Existe no mercado Brasileiro um potencial atual para o fornecimento de equipamentos para 1,8 milhão de residências e estima-se que cerca de 300 mil residências no Brasil já possuam equipamentos de automação residencial. Portanto, existe ainda um mercado inexplorado de pelo menos 1,5 milhão de residências (AURESIDE).

Ainda segundo AURISIDE, no Brasil, existe um interesse por parte de 78% dos consumidores, no mundo a média é de 66%. Em termos mundiais, o valor de mercado em 2014 foi de US$ 20,38 bilhões, com uma expectativa de atingir US$ 58,68 bilhões até 2020 (AURESIDE). Esses índices mostram o grande interesse da população em sistemas de automação residencial, esses sistemas podem ter múltiplas finalidades que vão desde lazer a segurança.

2.2 Microcontrolador

O microcontrolador é um dispositivo onde estão integrados uma CPU, também chamada de *core* (núcleo), e circuitos auxiliares (periféricos) como memória de programa, memória de dados, circuito de *clock*, interface de comunicação serial, temporizadores/contadores, portas de I/O, etc. estão integrados uma CPU, também chamada de *core* (núcleo), e circuitos auxiliares (periféricos) como memória de programa, memória de dados, circuito de *clock*, interface de comunicação serial, temporizadores/contadores, portas de I/O, entre outros componentes.

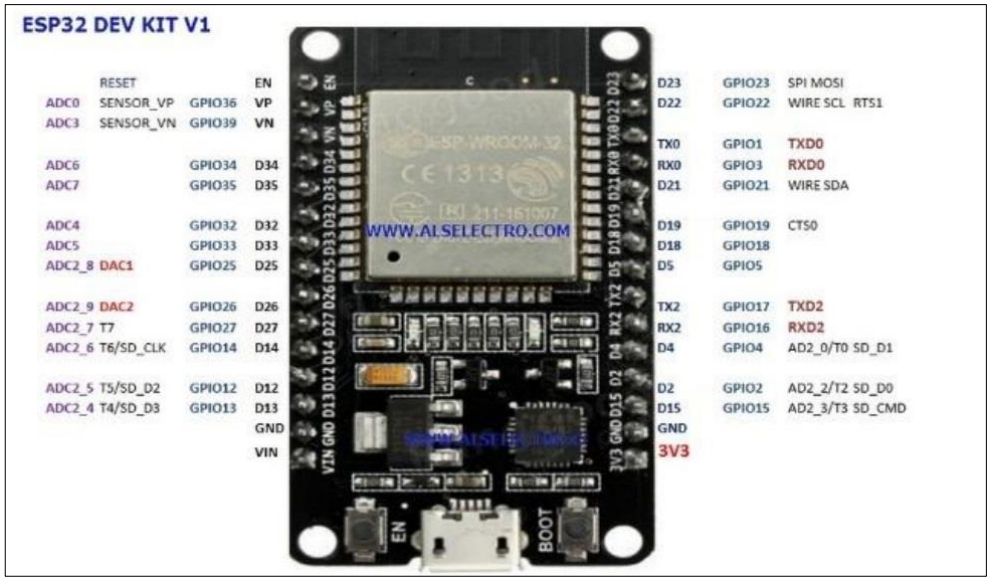
Os microcontroladores são compactos e possuem uma ótima relação custo/benefício, por estes motivos estão presentes em quase tudo que envolve a eletrônica. Eles são dispositivos que possuem internamente todos os componentes necessários para seu funcionamento autônomo.

2.2.1 Microcontrolador ESP32 nodeMCU

O módulo ESP32 NodeMCU é um dispositivo IoT (Internet das Coisas), conforme mostra a figura 2. É constituído por um microprocessador dual core Tensilica Xtensa de 32 bits com suporte embutido à rede Wi-Fi (802.11) e bluetooth versão 4.2, e com a memória flash integrada. Nesta plataforma, desenvolvida a partir de 2017, a programação pode ter mais de uma interface, como IDE (Arduino). A plataforma ESP-WROOM-32 é voltada para prototipagem IoT. Esse projeto, open source ou código aberto, consiste em um modelo de

desenvolvimento que promove um licenciamento livre da esquematização de um produto, e a redistribuição universal desse esquema oferece a possibilidade para que qualquer um consulte, examine ou modifique o produto, e também possa aprimorá-lo. O microcontrolador possui 36 GPIOs (entradas ou saída) e 18 entradas A/D e duas saídas D/A.

Figura 2 - Microcontrolador ESP32



Fonte:Site Saravana Electronics (2017).

2.2.1.1 Hardware

A placa ESP32 a ser utilizada no projeto será a versão ESP-WROOM-32. Esta placa utiliza o microcontrolador ESP32-D0WDQ6 de 12 bits e dois núcleos (dual core), de acordo com a documentação (ESPRESSIF, 2018c) as especificações estão na Tabela 2 a seguir:

Tabela 2 - Especificações do ESP-WROOM-32

ESP-WROOM-32	
Tensão de Operações	2,7 V/3,6 V
Tensão de Operação (Recomendada)	3,3 V
Memória interna ROM	448 kB
Memória interna SRAM	520 kB
Frequência de clock interno	40 MHz
Corrente de operação (média)	80 mA

Fonte: Adaptado de ESPRESSIF (2018c).

2.2.1.2 Portas

A documentação do ESP32-WROOM-32 (ESPRESSIF, 2018c) informa que há no total 32 portas digitais, das quais 16 podem ser utilizadas como saída PWM de 12 bits. Cada porta digital opera em uma tensão de 0 V para nível lógico baixo ou 3,3 V para nível lógico alto, podendo fornecer uma corrente de 80 mA em condições normais de operação.

A placa possui 18 entradas analógicas para conversão digital, as quais fornecem uma resolução de 12 bits na escada de 0 a 3,3 V. Estas entradas serão utilizadas para a aquisição dos dados dos sensores que vem em um formato analógico, convertendo-os para uma escala digital entre 0 e 4095 (12 bits) para posteriormente serem manipulados no software (ESPRESSIF, 2018b).

De acordo com a documentação do ESP-WROOM-32 (ESPRESSIF, 2018c) os pinos de tensão existentes da placa são:

- 5 V: Tensão de saída com 5 V fornecida pela placa.
- 3,3 V: Tensão de saída com 3,3V fornecida pela placa com corrente máxima de 80 mA.
- GND: Pino conhecido como terra, que realiza fechamento do circuito.

A placa ainda possui dez pinos GPIO com suporte a toque capacitivo, dois canais conversores digital-analógico (DAC) de 8 bits, duas portas com interface I²C, muito utilizadas para módulos externos a placa, duas portas com interface UART, e duas portas com interface I²S, utilizadas para conectar componentes de áudio digital (ESPRESSIF, 2018c).

As vantagens do ESP32 NodeMCU são:

- Baixo custo;
- Suporte integrado para rede Wi-Fi 802.11 b/g/n e bluetooth versão 4.2;
- Tamanho reduzido da placa;
- Baixo consumo de energia, alto desempenho;
- Amplificador de baixo ruído, robustez, versatilidade e confiabilidade.

Entre as desvantagens do ESP32NodeMCU, podem-se citar:

- Pinagem reduzida;
- Necessidade de aprender uma nova linguagem por ser um projeto recente;
- Possui pouca documentação, pois essa placa foi lançada em 2017.

2.3 Ambiente de Desenvolvimento (IDE)

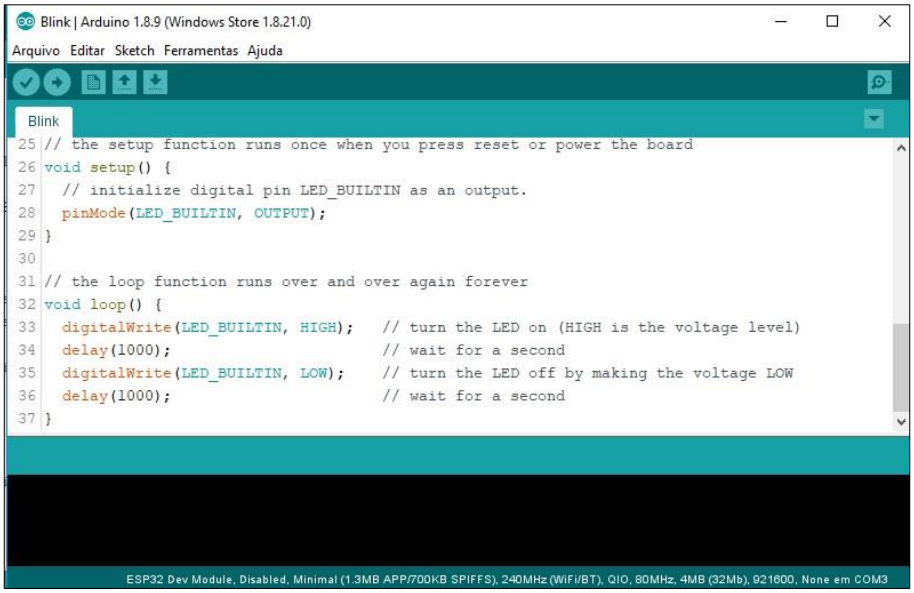
O ambiente de desenvolvimento consiste em um software gratuito, onde será escrito a sequência de instruções que serão interpretadas pelo Arduino. Ele se conecta ao hardware para realizar a comunicação e carregar o código desenvolvido. Os códigos escritos neste ambiente de desenvolvimento são chamados de Sketches, que são salvos com a extensão ".INO". Quando se inicia a Ambiente para Desenvolvimento Integrado (IDE) encontra-se a área para escrever o software, a barra de ferramentas, o console de textos, que exibe uma lista completa de erros no código e o resultado das instruções enviadas ao Arduino, e os seguintes botões descritos na tabela a seguir na tabela 2:

Tabela 2 - Descrição dos botões da IDE do arduino.

Verify:	Tem a função de verificar erros no código.
Upload:	Compila o código e carrega para a placa.
New:	Cria um novo esboço.
Open:	Apresenta um menu com vários códigos prontos.
Save:	Salva o *sketch* atual.

Há também alguns comandos adicionais oferecidos para facilitar o desenvolvimento, como por exemplo: *Copy for* fórum, encontrado dentro do menu editar, que torna possível copiar o código para postar em fóruns, *Copy as HTML*, oferece opção de copiar como HTML para inserir em páginas *Web, Import Library*, que adiciona uma biblioteca ao projeto, *Examples*, onde encontra-se diversos exemplos de código prontos, entre outras opções que tornam essa ferramenta muito simples de trabalhar. A IDE do arduino é mostrada na figura 3, o *Sketche* contido na imagem da IDE é o código de exemplo *Blink*:

Figura 3 - IDE do Arduino



Fonte: Autoria própria

2.3.1 Bibliotecas Arduino

As bibliotecas são um conjunto de códigos disponibilizados pela desenvolvedora do projeto Arduino, que têm por objetivo facilitar a comunicação com os componentes acoplados à placa. Existem diversas bibliotecas disponíveis, algumas são internas, como por exemplo: *LiquidCrystal*, que foi utilizada no desenvolvimento do projeto de automação residencial aqui apresentado, outras são disponibilizadas para download e podem ser instaladas muito facilmente. A seguir é feita uma descrição das bibliotecas isoladas neste trabalho.

2.3.2 Biblioteca LiquidCrystal

Esta biblioteca permite que uma placa Arduino controle as telas *LiquidCrystal (*LCDs) baseadas no chipset Hitachi HD44780 (ou compatível), encontrado na maioria dos LCDs baseados em texto. A biblioteca trabalha no modo de 4 ou 8 bits, isto é, usando 4 ou 8 linhas de dados além das linhas *rs, enable, and e rw control* (ARDUINO).

2.3.3 *Biblioteca TTP229* (Teclado capacitivo)

Esta biblioteca permite que uma placa Arduino ou outra placa disponível na IDE do Arduino receba a detecção dos toques na superfície numérica indicadas no *Teclado Capacitico Touch* (*Toque*), este teclado é baseado no circuito integrado TTP229-BSF (8229BSF), o teclado possui 16 teclas sensíveis ao toque e com alta sensibilidade.

2.3.4 Biblioteca FirebaseESP32

Biblioteca de clientes Arduino do *Firebase Realtime* Database para Espressif ESP32, esta biblioteca faz conexão entre uma placa ESP32 e o banco de dados Firebase. E necessário o download desta biblioteca pois a placa é nativa da plataforma arduino. Essa biblioteca fornece as operações mais confiáveis para ler, armazenar, atualizar, excluir, fazer backup e restaurar os dados do banco de dados do *Firebase Realtime*. A biblioteca pode ser usada em outras placas ESP32, para as primeiras placas nodeMCU que são placas mais antigas, como o nodeMCU8266 é utilizada uma outra biblioteca que apresenta as mesmas funcionalidades, mas com compatibilidade com o ESP8266

2.3.5 Biblioteca WiFi

Essa biblioteca permite que uma placa Arduino se conecte à internet. Ele pode servir como um servidor aceitando conexões de entrada ou um cliente fazendo conexões externas. A biblioteca suporta criptografia WEP e WPA2 Personal, mas não WPA2 Enterprise. Observe também que, se o SSID não for transmitido, a blindagem não poderá se conectar (ARDUINO).

2.3.6 Biblioteca Servo

Esta biblioteca permite que uma placa Arduino controle servo motores. Os servos padrão permitem que o eixo seja posicionado em vários ângulos, geralmente entre 0 e 180 graus. Os servos de rotação contínua permitem que a rotação do eixo seja ajustada para várias velocidades.

A biblioteca Servo suporta até 12 motores na maioria das placas Arduino e 48 no Arduino Mega. Em outras placas que não o Mega, o uso da biblioteca desativa a funcionalidade *analogWrite* () (PWM-*Pulse Width Modulation*) nos pinos 9 e 10, independentemente de haver ou não um Servo nesses pinos. No Mega, até 12 servos podem ser usados sem interferir na funcionalidade do PWM, o uso de 12 a 23 motores desativará o PWM nos pinos 11 e 12.

2.4 Android SDK

A principal linguagem de programação para Android é Java. Porém, para se obter o melhor do hardware, o processo de transformação de um código fonte em uma aplicação executável não é o mesmo de uma aplicação Java tradicional, sendo necessário um compilador e algumas ferramentas especais. Primeiro, o compilador irá prover uma codificação otimizada e em seguida procedimentos especiais irão transformar o código compilado em um formato para o Android (GRIFFITHS; GRIFFITHS, 2015, p. 5). Para atender esta necessidade surgiu o kit de desenvolvimento de software (SDK) específico para o Android.

O SDK, que está incluído na instalação do Android Studio, integra tudo o que é necessário para a elaboração de softwares para Android. Encontramos nesse kit algumas ferramentas de desenvolvimento - como compilador e depurador de código -, um emulador de máquina virtual, documentação e exemplos de código. (MEIER, 2009, p. 12)

A cada lançamento de uma nova versão do sistema operacional Android, um SDK proporcional também é lançado. Assim é necessário realizar uma escolha de qual SDK será usado tendo em vista a verão do Android alvo do projeto. (TECHOPEDIA, s.p.)

2.5 Mecanismo de Comunicação do Sistema

O protótipo do sistema de automação residencial aplicado para segurança apresentado neste projeto, baseia-se na comunicação sem fio. Esta comunicação é feita através da rede Wi-Fi que conecta o sistema a rede de internet e o usuário ao sistema, para fazer a conexão entre o sistema e o usuário foi desenvolvido um aplicativo móvel, este aplicativo é responsável pelo controle do alarme, abertura do portão (ou de portas, dependendo da necessidade do usuário) e monitoramento de status do sistema. O aplicativo foi desenvolvido na Plataforma Android Studio que é conectado ao Firebase para que possa transmitir as informações remotamente, essas informações podem ser transmitidas como *inteiros, floats, booleanos* e *string.*

2.5.1 ANDROID STUDIO

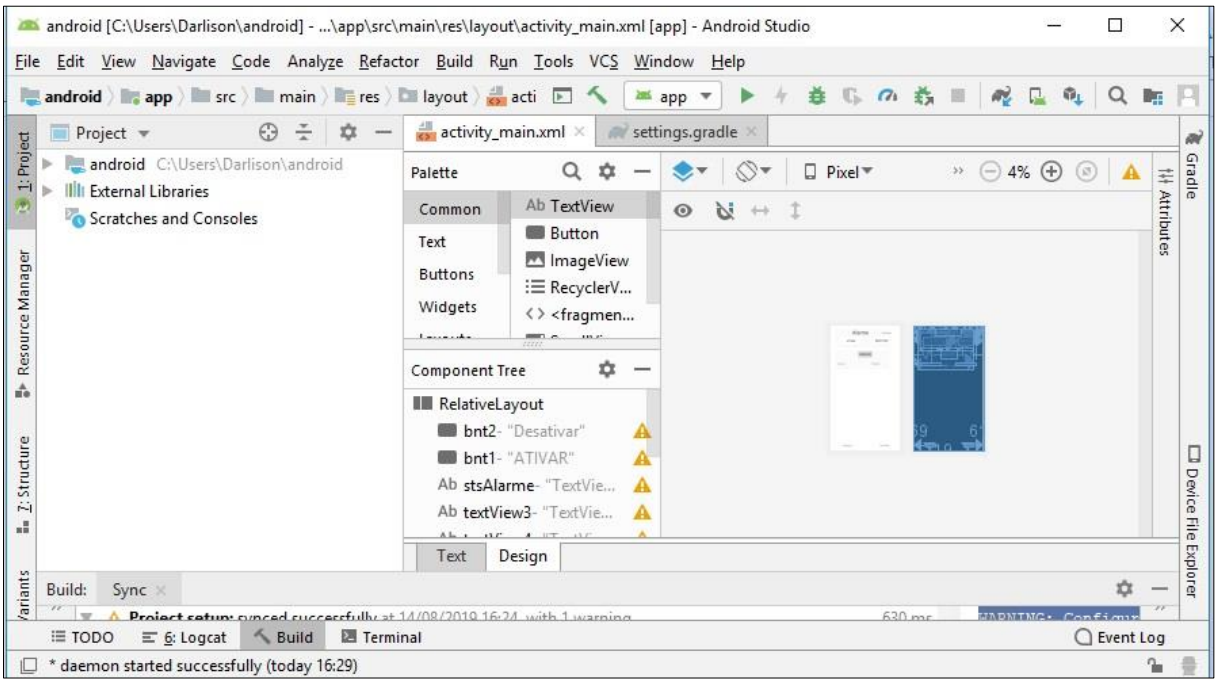
Em 2013 o Android Studio (AS) foi lançado como o ambiente integrado de desenvolvimento (IDE) oficial de aplicações para o sistema operacional Android, em substituição ao Android Developer Tools (ADT), um plugin para o Eclipse IDE. Essencialmente é uma variante da edição Community do IntelliJ IDEA, um IDE para a linguagem Java que combina o SDK do Android com o kit de desenvolvimento Java (JDK)

para disponibilizar o necessário para criar, executar e depurar softwares para Android, além de exportá-los para distribuição – por exemplo, fazer o upload para o Google Play (DEITEL; DEITEL; WALD, 2016).

Um conjunto de particularidades do Android Studio o tornam a principal escolha de IDE, em oposição ao Eclipse. Para CARVALHO, (2013, s.p.), os temas para a seleção da aparência da interface gráfica, a personalização de teclas de atalho, o auto completar do código sem pressionar teclas de atalho, a integração com sistemas de controle de versão e a criação de layouts por meio de arrastar e soltar componentes na tela são as principais vantagens do Android Studio.

A figura 4 apresenta a interface gráfica do Android Studio no modo de design da tela do aplicativo que está sendo desenvolvido. Nesse modo é possível arrastar e soltar componentes para a tela do app, como caixas de texto, botões e imagens.

Figura 4 - Editor de layout do Android Studio



Fonte: Autoria própria

Os projetos são estruturados em módulos onde estão organizados os arquivos de códigos fonte e de recursos, como imagens, sons e vídeos. Esse arranjo, que é ancorado do lado esquerdo na janela, ajuda a acessar de forma rápida os principais arquivos da aplicação (ANDROID DEVELOPER, s.p. 2019).

2.5.2 Firebase

O Firebase é uma plataforma para a construção de aplicativos mobile e web através de ferramentas e infraestruturas que visam ajudar desenvolvedores a construir aplicativos de qualidade (FIREBASE DATABASE 2016) onde são agrupados diversos serviços importantes tais como o sistema de análise (*Firebase Analytics*), sistema de autenticação de usuário (*Firebase Auth*), armazenamento (*Firebase Storage*), banco de dados (*Firebase Realtime Database*), hospedagem (*Firebase Hosting*) entre outros. Porém, para este trabalho foi utilizado apenas o serviço de banco de dados não relacional, o Firebase *Realtime Database*. Esse banco de dados nada mais é do que uma árvore JSON (JavaScript Object Notation) gigante em que todos seus dados estão armazenados nos nodos, o que facilita uma modelagem simples de dados. O maior benefício do Firebase Database Realtime é que ele já possui um sistema de sincronização instantânea implementado, fazendo com que, caso ocorra uma modificação no banco, todos os aplicativos que tenham a referência daquele item, o atualizem automaticamente, ao invés de trabalhar com requisição e resposta normalmente utilizado em outros bancos.

2.6 Componentes Físicos

2.6.1 Display 16x2

O LCD (*Liquid Crystal Display* – Display de Cristal Líquido) é um elemento fundamental para que os equipamentos eletrônicos tornem-se mais compactos, interativos e de fácil operação, pois pode facilitar a forma como o usuário vai interagir com o equipamento e, consequentemente, aumentar o valor agregado a ele. Existe uma variedade de displays LCD no mercado, desde os LCDs capazes de exibir uns poucos caracteres até LCDs gráficos, coloridos, e nos mais variados tamanhos.

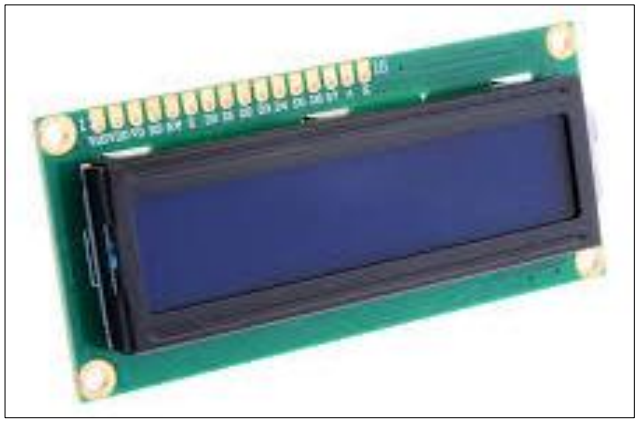
Os LCDs caracteres normalmente são compatíveis com o código ASCII, e podem gerar letras, números e caracteres especiais, além de caracteres europeus e gregos.

Os LCDs caracteres podem possuir ainda uma memória RAM interna que permite criar caracteres especiais ou símbolos que podem ser imprescindíveis numa determinada aplicação.

O LCD caractere 16x2, ou seja, dezesseis caracteres por duas linhas é um dos LCDs mais utilizados em equipamentos eletrônicos.

A Figura 5 mostra a função de cada um dos 16 pinos do LCD caractere 16 x 2. Os dois primeiros pinos (1 e 2) são relativos a alimentação do componente e devem ser ligados a uma tensão de alimentação, sendo 5 V o valor ideal.

Figura 5 - Display LCD 16x2



Fonte: Felipeflop

Os pinos 15 e 16 acionam um conjunto de LEDs (*Light Emitting Diode* – Diodo Emissor de Luz) responsáveis pela iluminação do painel (*backlight*). Sua alimentação é feita aplicando-se 5 VDC nesses pinos (Figura 6).

Figura 6 - Função Pinos

Pino	Símbolo	Função	Pino	Símbolo	Função
1	V_{SS}	Comum	9	DB2	Dado
2	V_{DD}	5V	10	DB3	Dado
3	V_0	Ajuste do contraste	11	DB4	Dado
4	RS	Seleção de registro	12	DB5	Dado
5	R/W	Leitura/Escrita	13	DB6	Dado
6	E	Inicia ciclo R/W	14	DB7	Dado
7	DB0	Dado	15	A	Anodo
8	BD1	Dado	16	K	Catodo

Fonte: (Muniz, 1999)

O pino três (V0) é utilizado para que seja possível fazer ajuste no contraste da imagem exibida no Display. Para isso, liga-se esse pino ao centro de um potenciômetro de 10kΩ, para que se possa obter uma tensão variável entre e VSS e VDD.

O pino RS (*Register Select*) é utilizado para informar ao LCD o tipo de informação que se encontra no barramento de dados (DB7:DB0).

Para dar início a um ciclo de leitura ou de escrita, é necessário aplicar um pulso no pino 6 (E – *Enable*). No pino E, que normalmente é mantido em nível lógico 0, deve ser aplicado

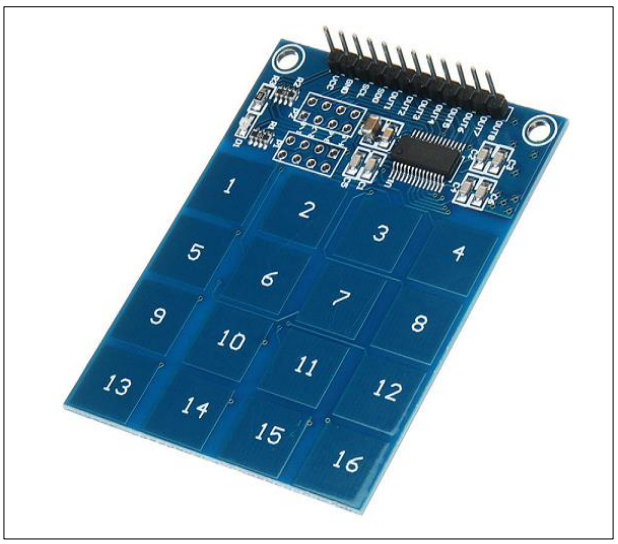
nível lógico 1 e depois aplicado nível lógico 0. Quando for efetuada uma escrita no LCD, é necessário que o dado se mantenha por alguns nano segundos no barramento depois do pulso aplicado no pino E. Isso é necessário porque o LCD efetua a leitura do dado na borda de descida do pulso aplicado ao pino E. No caso de leitura no LCD, alguns nano segundos após ser aplicado nível lógico 1 no pino E, o dado estará disponível no barramento para ser lido pelo microcontrolador. Depois do dado lido, no pino E pode ser aplicado novamente nível lógico 0.

Os pinos de DB0 a DB7 equivalem ao barramento de dados paralelo. Este é um barramento bidirecional, pois ele pode ser efetuado tanto para a escrita quanto para leitura dos dados armazenados na memória RAM do LCD. Apesar de existirem oito vias de dados, esses displays também podem operar com quatro vias (DB4 a DB7), ficando assim as demais vias sem função. Neste caso, as informações são enviadas em dois pacotes de quatro bits cada um (Muniz, 1999).

2.6.2 Teclado Capacitivo Touch (TTP229)

O Teclado Capacitivo Touch TTP229 com 16 Teclas é capaz de detectar toques em cada uma das superfícies numéricas indicadas na placa. O teclado é baseado no circuito integrado TTP229-BSF (8229BSF) e o mesmo possui 16 teclas sensíveis ao toque e com alta sensibilidade. Quando um dedo toca as regiões numéricas indicadas na placa à saída é ativada.

Figura 7 - Teclado Capacitivo Touch



Fonte: MasterWalker, 2019

Especificações e características:

- Controlador: 8229BSF;

- Tensão de operação: 2,4 – 5,5V DC;
- Corrente típica (8 teclas): 2 µA;
- Corrente típica (16 teclas): 2,5 µA;
- Interface de comunicação: I2C;
- Taxa de amostragem: 8 Hz (lenta) / 16 Hz (rápida);
- Quantidade de teclas: 16;
- Configuração de funcionamento: 8 teclas ou 16 teclas;
- LED indicador para presença de tensão.

É geralmente usado em projetos com Arduino ou outras plataformas microcontroladas que necessitem de um teclado numérico capacitivo para entrada de dados, no projeto foi utilizado para entrada de dados como, por exemplo, entrada de senha e controle do sistema.

2.6.3 Sensores

Os sensores são responsáveis pela obtenção de valores de entrada no ESP32, com a aquisição dos valores é possível converte-los para um formato digital em uma resolução de 12 bits através das portas analógicas ou digitais, posteriormente manipular os mesmos dentro do software.

2.6.3.1 Sensor de Movimento (PIR)

O sensor de movimento PIR, é um sensor que usa Piroeletricidade (do grego pir, fogo e electricidade) que é a capacidade de certos materiais gerarem uma tensão temporária quando são aquecidas ou arrefecidas utilizado infravermelhos. Os PIR também são chamados de “detectores de movimento” e “detectores de infravermelhos passivos”. Eles são chamados de “passivo” porque não projetaram qualquer tipo de feixe.

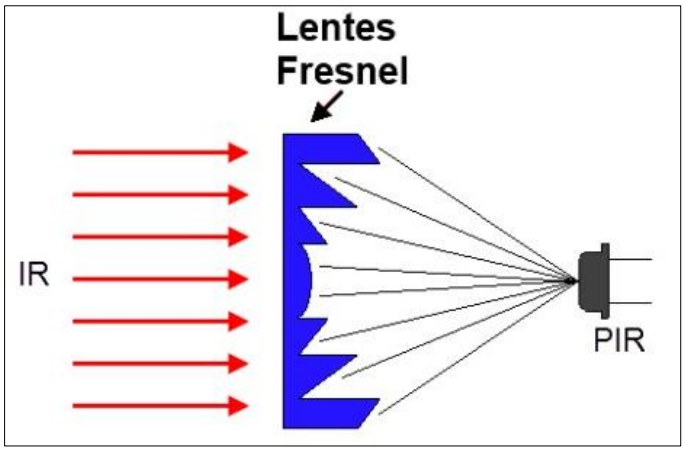
Os PIR são constituídos por três partes básicas. No lado de fora existe uma lente de Fresnel, que é uma janela de plástico branco opaco, como mostra a figura 8. O chip detector fica atrás da lente.

Figura 8 - Sensor PIR

Fonte: ARDUINOECIA, 2019

A lente Fresnel é muito parecida com as lentes usadas em faróis. Esta técnica permite que luzes, relativamente pequenas, possam ser vistas a distâncias muito grandes. Neste caso a lente Fresnel serve para amplificar a assinatura de calor infravermelho que está sendo dissipada por uma pessoa ou animal a passar (animais de sangue quente). Para além de amplificar a energia, a lente também divide o sinal de infravermelhos em vários feixes de luz. A figura a seguir representa o funcionamento de uma lente Fresnel.

Figura 9-Funcionamento da lente Fresnel

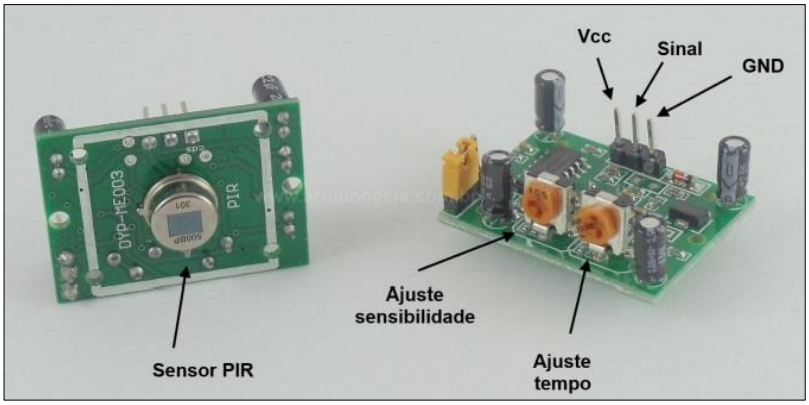


Fonte: ARDUINOECIA, 2014

O chip detector é um pequeno componente electrónico que é sensível à energia infravermelha. O chip vê as ondas infravermelhas e cria um pequeno pico de tensão, o qual é enviado para um circuito e desencadeia um contato, contato esse que pode ser usado para acionar um alarme, acender uma luz, abrir uma porta ou qualquer outra função que se pretenda. Neste referido projeto o sensor PIR foi utilizado no sistema alarme residencial, onde o usuário pode ativar e desativar o alarme pelo iPhone utilizado pelo protótipo.

Este módulo com o sensor PIR é de fácil utilização quando se trabalha na plataforma arduino, além de ser um componente facilmente encontrado nas lojas especializadas e com um preço bastante acessível, contribuindo para a utilização deste no trabalho de automação residencial aqui apresentado. A figura a seguir mostra com mais detalhes este módulo sensor.

Figura 10 - Módulo Sensor PIR Detalhado



Fonte: ARDUINOECIA, 2014

A figura acima apresenta de forma bem ilustrativa todas as partes integrantes deste módulo, este ainda possui ajuste de sensibilidade, onde o usuário pode controlar sensibilidade de detecção do sensor, podendo ajudar a evitar falsos alarmes e também ajuste de tempo, onde pode-se configurar o tempo que o sensor permanece com estado ativado ao detectar algum movimento.

2.6.3.2 Sensor Óptico

Os sensores Ópticos ou fotoelétricos tem como princípio de funcionamento o uso da propagação da luz, este tipo de sensor é utilizado comumente para indexação de objetos ou para medições de distância em que um objeto se encontra em relação ao sensor

A luz emitida pelos sensores óticos pode ser dos seguintes tipos:

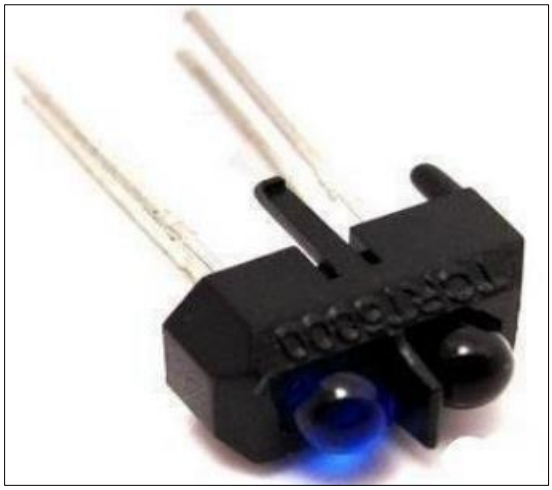
- Vermelha;
- Laser Vermelho;
- Infravermelho.

Cada tipo de luz é indicado para uma determinada aplicação, por exemplo, a luz VERMELHA é indicada para detecção de objetos opacos de médio e grande porte, como caixas de papelão e embalagens não metalizadas. Já a luz LASER é utilizada para detecções mais precisas envolvendo objetos de pequeno porte, devido ao feixe de emissão da luz ser estreito e

focalizado. Por último, a luz INFRAVERMELHA é utilizada quando há a necessidade de se detectar objetos transparentes, como vidro, garrafas plásticas entre outros objetos.

Este Sensor Óptico TCRT5000 de reflexão possui acoplado no mesmo dispositivo um sensor infravermelho (emissor) e um foto transistor (receptor). Foi especialmente projetado para bloquear outras faixas de luz que não sejam a do próprio emissor, evitando que iluminações do ambiente venham causar alguma interferência, no projeto o sensor foi utilizado como sensor de fim de curso, indicando fechamento de uma janela ou porta, na Figura 11 podemos ver o sensor óptico reflexivo TCRT5000.

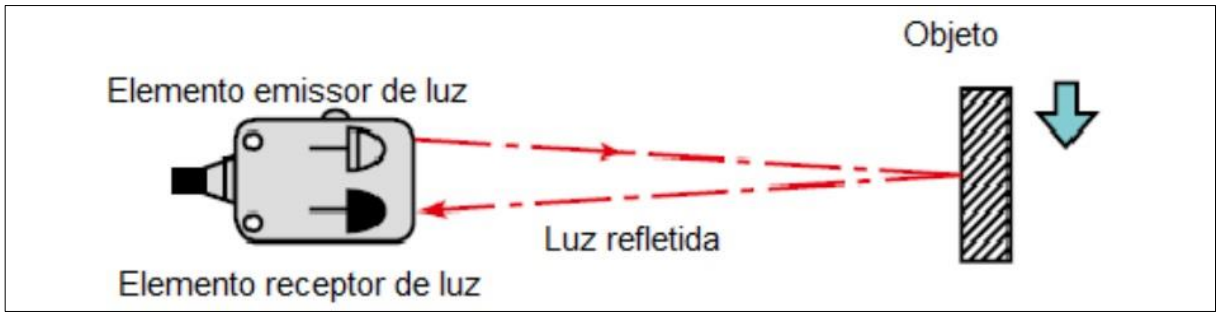
Figura 11 - Sensor Óptico TCRT5000 de reflexão



Fonte: ARDUINOECIA, 2014

A Figura 12 mostra o funcionamento do sensor.

Figura 12 - Princípio de funcionamento do sensor

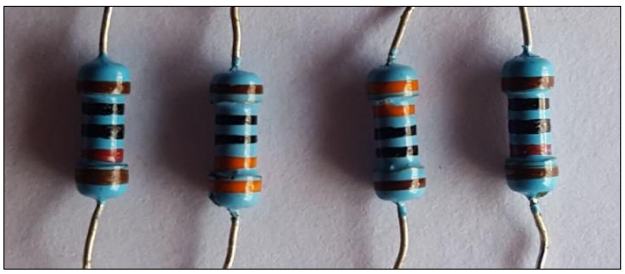


Fonte: SENSOR, 2019

2.6.4 Resistor

O resistor é um dispositivo elétrico muito utilizado na Eletrônica. Transforma a energia elétrica em energia térmica por meio do efeito Joule, limitando a corrente elétrica em um circuito. No caso do projeto proposto, o resistor será usado em conjunto com o sensor óptico evitando que ele receba uma corrente muito alta, a figura 13 mostra alguns resistores.

Figura 13- Resistores



Fonte: Autoria própria

2.6.5 Servo Motor

O servo motor é muito utilizado para aplicações de robótica, e também para o controle de braço mecânico, por causa da precisão do posicionamento desse sistema e do seu torque. Os módulos de servo motor apresentam movimentos proporcionais ao dos comandos indicados, e controla o giro e a posição. Os servos motores em geral têm uma abertura de 180°, podendo fazer adaptações para que se adeque no projeto e faça a abertura do portão (USINAINFO, 2017).

O servo motor utilizado como referência neste estudo foi o micro servo motor SG90, conforme a figura 14, um componente compacto, cujo torque é de 1,6 kg, suficiente para a movimentação do portão representado na maquete, com o seu ângulo de abertura é de 180°.

Figura 14 - ServoMotor SG90

Fonte: Site Usina da Informática, 2019

2.6.6 Buzzer 5V

O buzzer é um eletrônico pequeno, do tamanho de uma moeda, composta de 2 camadas de metal e uma camada interna de cristal piezoelétrico (como um sanduiche). Ao ser alimentado com uma fonte de sinal, vibra da mesma frequência recebida, funcionando como uma sirene ou alto-falante.

O mercado apresenta uma variedade de buzzers, sendo que eles se diferenciam pela alimentação, frequência e intensidade da onda sonora. O buzzer escolhido deve ser alimentado com 5 V, e possui um pino de controle do oscilador interno que pode emitir ondas na faixa de frequência 1500 MHz.

Figura 15 - Buzzer Ativo 5V Bip Contínuo

Fonte: Site Usina da Informática, 2019

O buzzer mostrado na figura 15 possui uma estrutura cilíndrica de diâmetro 12 mm e altura 10mm. O interessante do buzzer é que seja integrado ao sistema, buzzer é importante para avisar uma possível violação do alarme de modo sonoro, deste modo, inibindo o invasor.

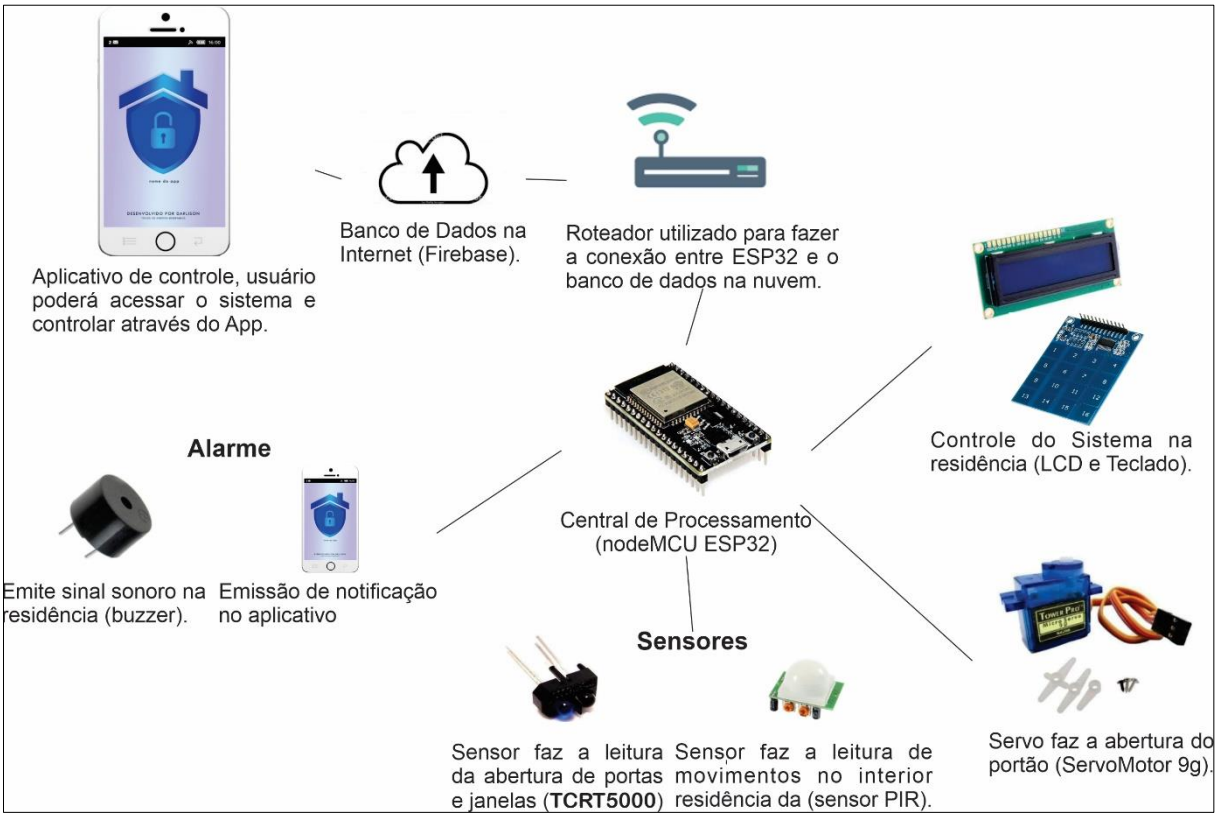
3 Metodologia do Modelo Proposto

3.1 Apresentação Geral do Modelo Proposto

O desenvolvimento deste projeto requer conhecimento em áreas distintas da computação, como conhecimento na área de algoritmo e programação, conceitos de sistemas embarcados, eletrônica, banco de dados e desenvolvimento de aplicativos móveis.

O projeto desenvolvido consiste em algumas funcionalidades de segurança, quais o sistema se divide em partes 5 partes principais, que são sistemas de controle interno, sistema de controle pelo Aplicativo (utilizando o Firebase), sistema de alarme, sistema de sensores e Central de processamento, que serão descritas com mais detalhamentos no decorrer deste capítulo. Em contribuição com a figura 16 e para melhor entendimento do que foi desenvolvida neste projeto. A figura a seguir apresenta o esquema geral do funcionamento deste protótipo.

Figura 16 - Descritivo do Projeto



Fonte: Autoria própria

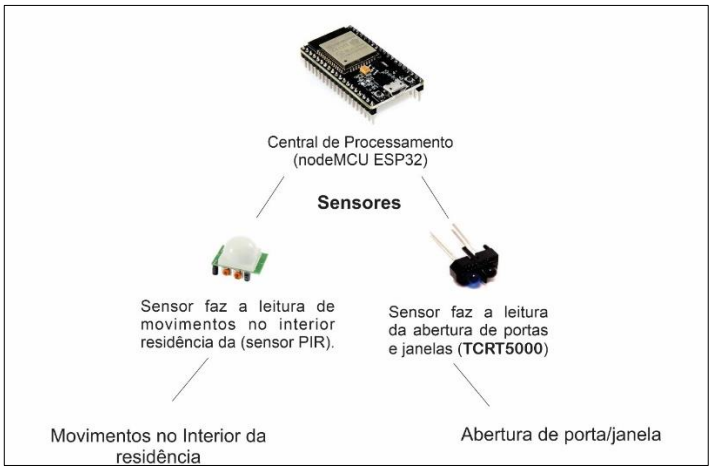
3.2 Descrição do Funcionamento.

Como mencionado anteriormente o sistema de segurança descrito possui basicamente cinco partes, estas são sistema de sensores, sistema de alarme, controle interno e aplicativo de controle, agora será descrito cada umas partes citadas.

3.2.1 Sistema de Sensores

O Sistema de Sensores do protótipo é uma das partes fundamentais no sistema, pois ele é parte do sistema que identifica possíveis tentativas de invasões e arrombamentos e transmite os dados para unidade central de processamento. Este sistema utiliza dois tipos de sensores, sensores ópticos e sensores de movimento (sensores PIR). Os sensores ópticos detectam objetos próximos a eles em uma área de milímetros por este motivo foram implantados em portas e janelas e conseguem detectar se uma porta ou janela está aberta ou fechada, seguindo este princípio o mesmo pode detectar possíveis arrombamentos ou invasões na residência. Os sensores PIR são capazes de captar movimentos de objetos a uma distância de até 7 metros e tem uma abertura de até 120° possibilitando cobertura de uma grande área, por isso são implantados em cômodos do protótipo, caso algum dos sensores citados detectarem alguma perturbação no sistema eles enviam para unidade de processamento, na figura 17 podemos ver um diagrama de funcionamento os dados

Figura 17 - Descritivo Sistema de Alarme

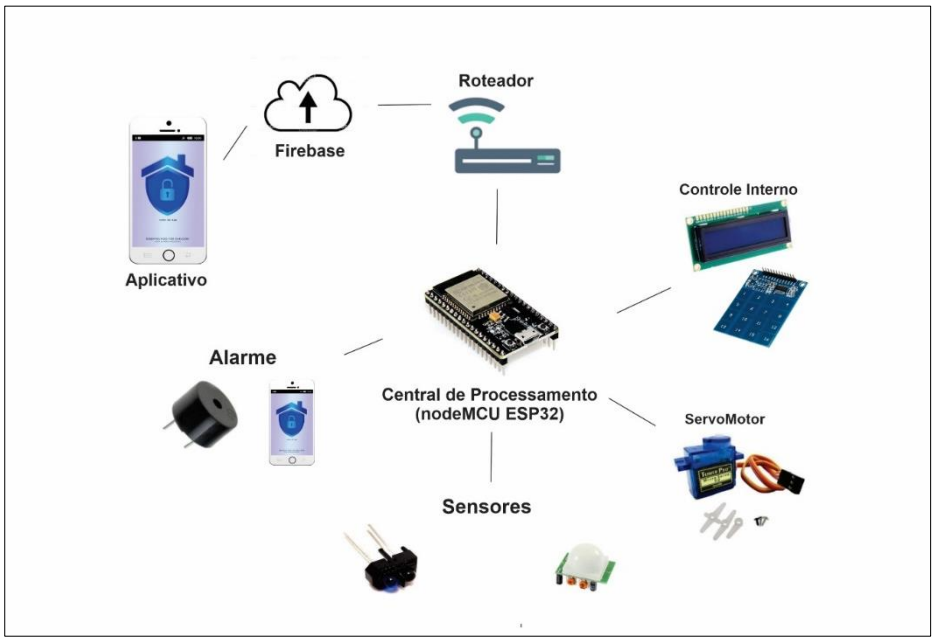


Fonte: Autoria própria

3.2.2 Sistema Central de Processamento

O Sistema Central de Processamento é o coração do projeto, responsável por todo o processamento, aquisição de dados através dos sensores, dispositivos de entrada e saída (Controle interno) e banco de dados (Firebase), além de fazer a comunicação entre todas as outras partes do projeto, para fazer todo o processamento foi utilizado o microcontrolador antes citado o ESP32, uma placa robusta e de bom processamento. A unidade central precisa estar em um ambiente seguro no interior da residência, para que deste modo não seja afetada por problemas externos e não viole a integridade do sistema, podemos ver um diagrama mostrando a conexão da Unidade Central de Processamento com os componentes do projeto (figura 18).

Figura 18 - Central de Processamento



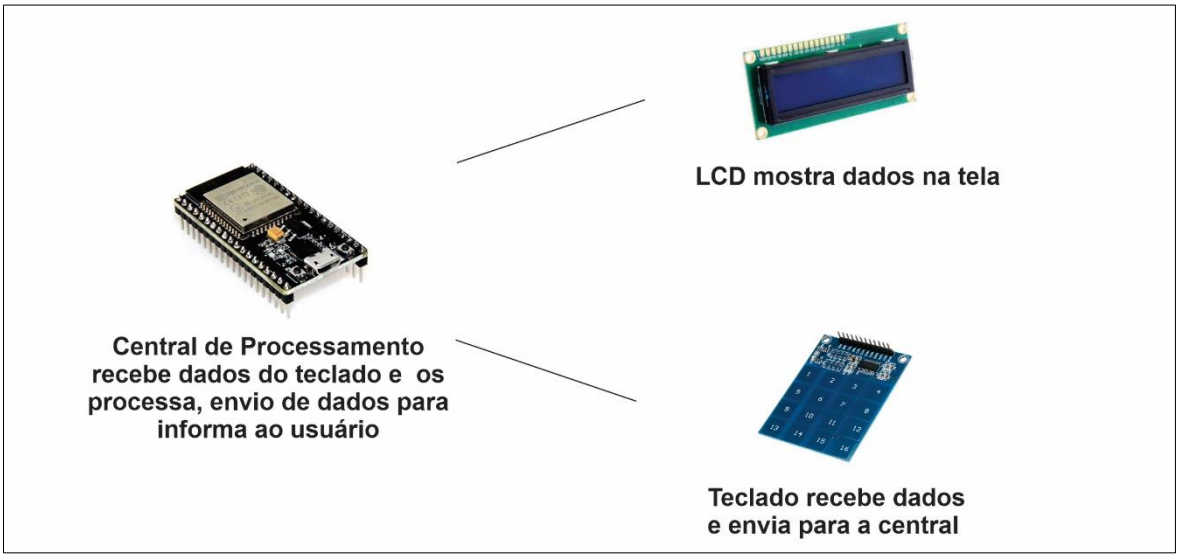
Fonte: Autoria própria

3.2.3 Controle Interno

O Sistema de Controle Interno é a parte que recebe os comandos e faz o envio para a central de Processamento além de receber os sinais da Central e mostrar para o usuário, o controle interno possui dois componentes principais o display LCD e o teclado capacitivo. O teclado capacitivo recebe as entradas de dados como senhas para ativar e desativar o alarme, comandos como abertura do portão ou verificação de portas e janelas, isso sem o uso do aplicativo. O LDC é responsável por fazer a interface com o usuário, mostrando qualquer dado

que a central de processamento deseja mostrar para o usuário, a figura 19 mostra o descritivo do sistema de controle interno.

Figura 19 - Controle Interno

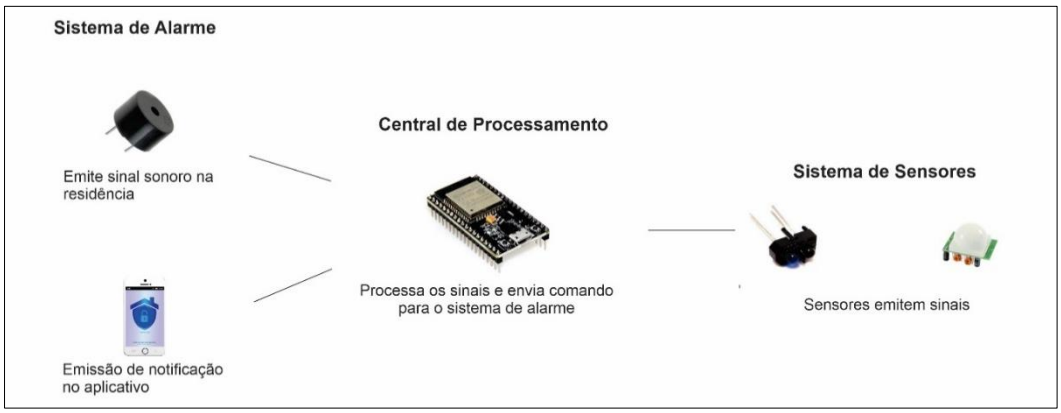


Fonte: Autoria própria

3.2.4 Sistema de Alarme

O Sistema de Alarme é peça fundamental no protótipo, pois o mesmo é responsável emitir sinais sonoros e notificações via aplicativo caso ocorra invasão na residência, este mecanismo é composto por dois itens um buzzer e um sistema de notificação via app. O Buzzer tem o papel de emitir um sinal sonoro, o objetivo destes sinais são os de inibir os possíveis invasores, o sistema de notificações possui objetivo similar. Ele gera notificação para usuário informando a invasão, o sistema de alarme funciona de maneira simples, o sistema de sensores detecte uma invasão esta invasão é processada e posteriormente o sistema de alarme é acionado, descritivo do Sistema de Alarme na figura 20.

Figura 20 - Sistema de Alarme

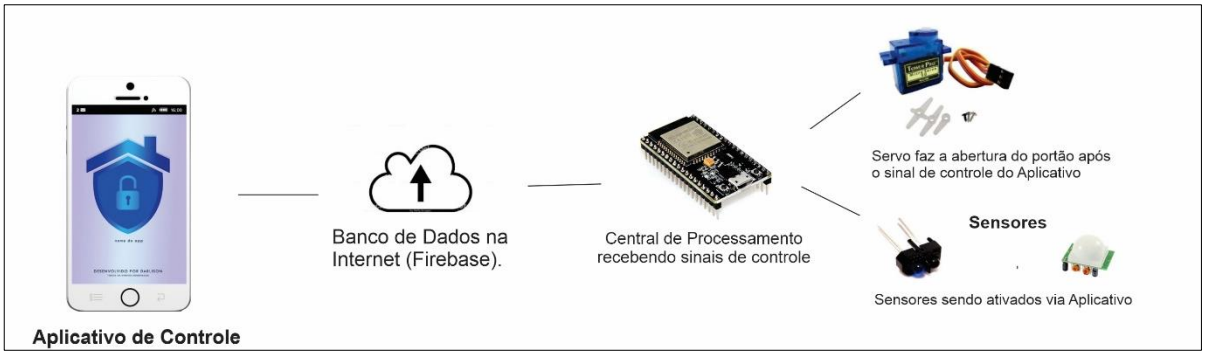


Fonte: Autoria própria

3.2.5 Aplicativo de Controle

O Aplicativo de Controle desempenha múltiplas tarefas, algumas delas são controle do sistema de alarme, controle de abertura do portão, verificação de portas/janelas e notificações de invasão, o aplicativo é um diferencial neste projeto pois através dele o usuário pode acompanhar a segurança de sua residência de qualquer lugar e de forma automática, desde que o mesmo possua uma conexão com a internet. O aplicativo é conectado ao banco de dados Firebase, que permite que o projeto seja controlado remotamente, o mesmo possui uma interface simples e de fácil manuseio e utilização, além de ser protegido por login e senha de acesso, o app pode modificar senhas de entrada na residência. A figura 21 é um diagrama que ilustra a conexão do Aplicativo com os componentes internos da residência.

Figura 21 - Aplicativo de Controle



Fonte: Autoria própria

3.3 Desenvolvimento do Modelo Proposto

Em busca de melhores resultados no desenvolvimento e implementação deste protótipo do sistema de segurança residencial, foi implementado um projeto prático, através de

uma maquete residencial. Para melhor visualização e aplicação dos conceitos já citados anteriormente. Este projeto, cujo sistema de controle de automação é controlado via dispositivos móveis, mais especificamente por um smartphone Android, onde foi passado para este o aplicativo com todas as funções do sistema de segurança residencial.

Portanto, dividiu-se o projeto em algumas etapas principais, estas vão desde os estudos iniciais até a fase de conclusão do projeto. Portanto, para desenvolver o projeto decidiu-se utilizar as seguintes etapas:

- Estudo de Metodologias;
- Construção da Maquete;
- Implementação Física do Protótipo;
- Criação do aplicativo móvel;

Sendo assim, estas etapas sequenciadas foram seguidas de forma a se obter um melhor resultado para análise sobre o projeto aqui desenvolvido.

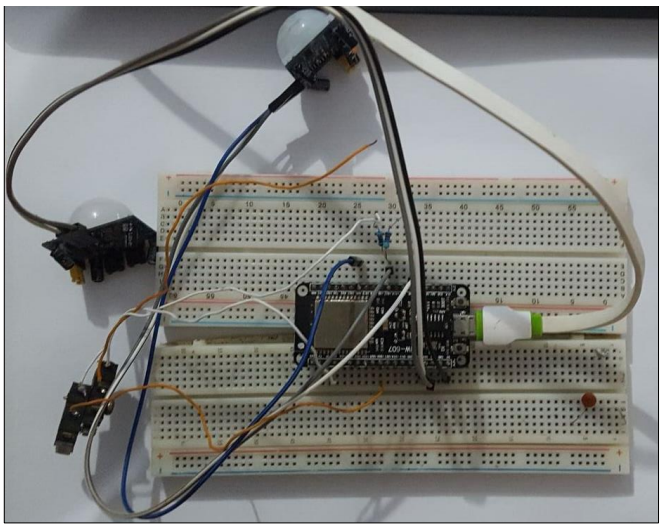
3.3.1 Testes iniciais

Após uma pesquisa inicial, a utilização da IDE do arduino com uma adaptação para a utilização do ESP32, mostrou-se extremamente viável para desenvolvimento e busca dos resultados esperados. Devido a projetos já implementados em áreas de automação. Além de ser de código aberto e possibilitar implementações funcionais e de baixo custo. Outro ponto para escolha foi o grande interesse do autor na área.

Ressalta-se que foram realizados estudos para melhor desenvolvimento na IDE do arduino utilizando o ESP32, bem como suas funcionalidades e linguagens utilizadas. Em

seguida, foram realizados uma série de experimentos iniciais para se obter compreensão dos componentes que são utilizados no projeto. A figura 22 mostra um dos testes feitos.

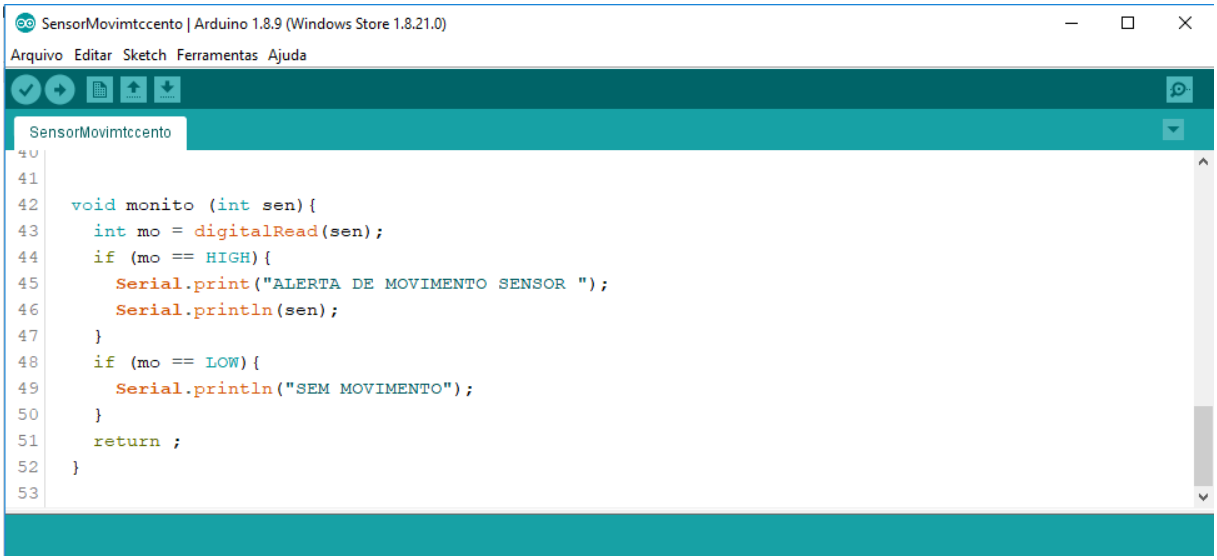
Figura 22 - Testes iniciais de sensores PIR e fim de curso



Fonte: Autoria própria

A figura 23 mostra a utilização de sensores PIR e sensores de fim de curso foram ser usados como sensores para alarme, pode-se notar a compatibilidade do ESP32 com a IDE do arduino.

Figura 23 - Testes iniciais de sensores PIR e fim de curso na IDE



Fonte: Autor

Conforme explicado na seção 2.6.3.1 o sensor PIR possui em sua saída um sinal de digital para informa que um objeto foi captado, por este motivo observa-se no código acima um simples "*if*" para saber qual a saída do sensor. Assim como estes testes que foram apresentados na figura 3.6, foram realizados uma série de testes com outros componentes que seriam também utilizados no projeto, como sensor óptico, servo motor, entre outros.

Logo em seguida aos testes com os primeiros componentes, foi realizado um estudo sobre a utilização do uso da rede wi-fi do ESP32 que seria uma peça chave neste projeto. Conforme visto na seção 2.4.1 este pode conectar-se à internet, sendo assim seria possível enviar comandos através de um celular conectado à rede. Então, assim como os demais componentes a utilização da rede wi-fi foi testada inicialmente com um código simples e posteriormente foi realizada a simulação mais complexa, esta simularia o funcionamento do protótipo que seria construído, onde os sensores estavam conectados ao ESP32, a tela serial mostrava os resultados simulando os envios de dados para o banco, aplicativo e dispositivos de saída.

Os testes iniciais contribuem para que no decorrer projeto sejam contornados possíveis erros. Sejam estes problemas de software ou de hardware, por exemplo incompatibilidade de componentes ou de suporte de software ou até mesmo periféricos de mesma função, mas com funcionamento diferente. Uma delas foi a substituição sensores de fim de curso por sensores óptico, pois mostraram ser mais fáceis de serem implementados no protótipo e não sofrerem desgastes mecânicos.

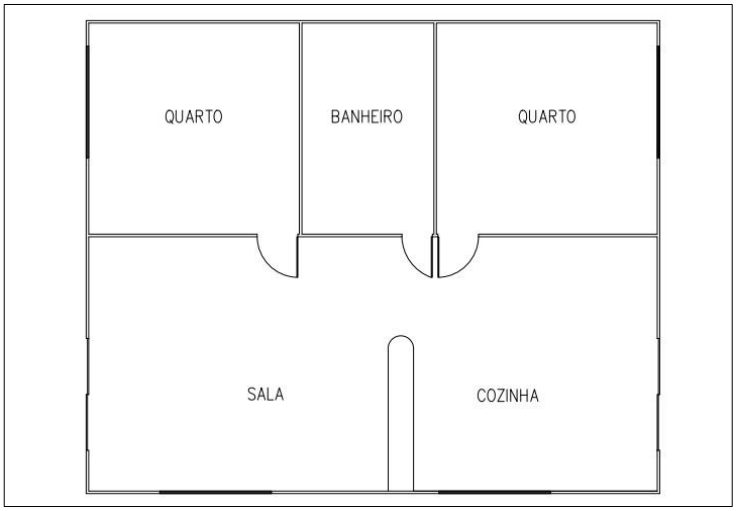
Posteriormente foi estuda o uso do Firebase e implemento programas simples para testar o carregamento e atualização de variáveis pelo banco de dados, o Firebase foi escolhido como o banco de dados do projeto pois ele possui todos os protocolos de segurança automatizados sem necessidade de configuração do mesmo, além de autenticação e atualizações em tempo real do banco, por utiliza a segurança do Google aumenta a integridade do sistema proposto.

Após todo o estudo de metodologias que poderiam ser utilizadas e realização dos testes com os componentes e diferentes aplicações de controle, passou-se para a segunda etapa do desenvolvimento do modelo proposto, construção da maquete que será utilizada.

3.3.2 Construção da Maquete

A segunda etapa no desenvolvimento deste projeto é a construção da maquete residencial para aplicação e simulação prática do sistema de segurança residencial. Como futuramente pretende-se implementar este sistema na residência real, as dimensões da maquete são as mais próximas de uma casa popular, casa tem uma sala, dois quartos, um banheiro e uma cozinha. A maquete da residência foi construída com dimensões de 56 cm de comprimento, 46 cm de largura e 20 cm de altura. Para projetar esta maquete foi utilizado um software de engenharia para este fim, a figura 24 mostra a planta baixa da residência já com as dimensões.

Figura 24 - Planta baixa da maquete



Fonte: Autoria própria

Após o término do projeto do modelo proposto, partiu-se então para construção física. Para isto, utilizou-se uma folha de compensado adquirida em uma loja de materiais de construção com dimensões de 2,20 m x 1,6 m x 5 mm, sendo suficiente para construção do

projeto de sistema de segurança residencial apresentado nesta monografia. A figura 25 apresenta a construção da maquete em sua fase inicial.

Fonte: Autorria própria

Figura 25 - Construção da Maquete

Finalizou a etapa de construção da maquete, pintando-a para dar mais realidade ao projeto, a figura 26 mostra a pintura da maquete.

Figura 26 - Pintura da Maquete

Fonte: Autor

E por fim a figura a seguir mostra como ficou a maquete construída depois de pintada. Com isso, concluiu-se com êxito a segunda etapa do desenvolvimento deste projeto. Agora a etapa seguinte será a implementação física da parte eletrônica, central de controle e demais componentes do projeto.

Figura 27 - Estrutura da Maquete Concluída

Fonte: Autoria própria

3.3.3 Implementação Física do Protótipo

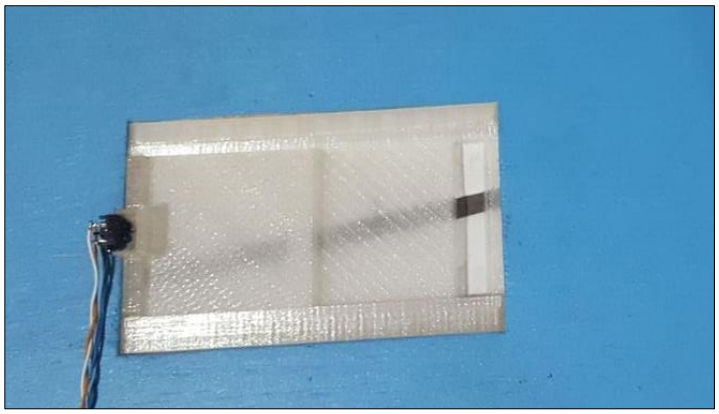
Nesta etapa é mostrado como foi realizada a montagem dos componentes que seriam utilizados em cada uma das funcionalidades do sistema, que como apresentado anteriormente consiste em: Sistemas de controle interno; Sistema de controle pelo Aplicativo (utilizando o Firebase); Sistema de alarme; Sistema de sensores; e Central de processamento.

3.3.3.1 3.3.3.1. Sistema de Sensores

O protótipo apresentado possui como principal funcionalidade a detecção de invasores na residência através do sistema de sensores, o usuário pode ativar ou desativar sistema, o mesmo pode detectar invasores e informar automaticamente ao usuário, além deste processo acontecer instantaneamente. O sistema é composto por dois tipos de sensores, sensores PIR de movimento e sensores ópticos.

Os sensores ópticos foram implantados nas portas e janelas da maquete. A figura 28 mostra uma das janelas com o sensor.

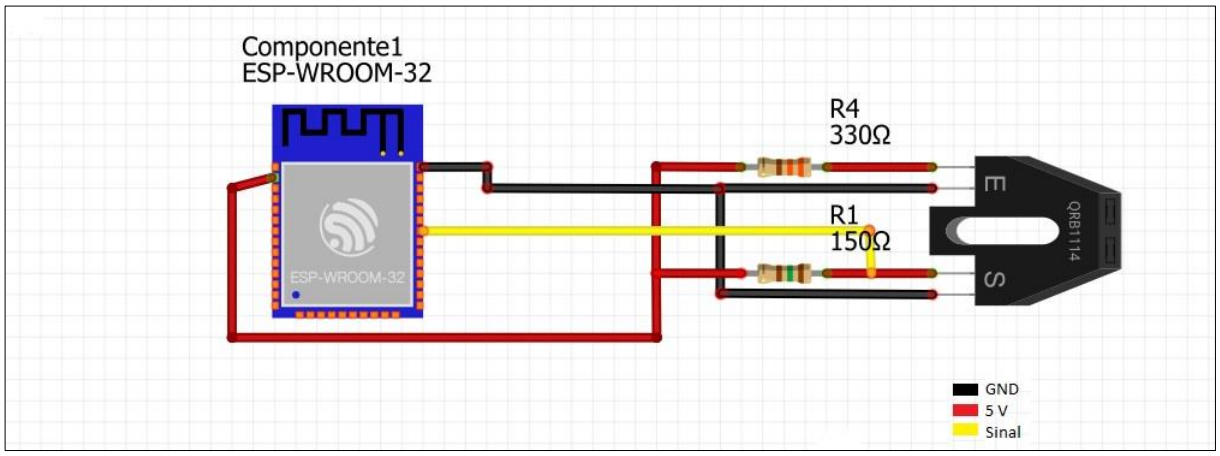
Figura 28 - Implementação do sensor



Fonte: Autor

Para funcionamento correto do sensor é necessário a utilização de resistores na alimentação de 5 V, um resistor de 150 Ω no sensor e 330 Ω no emissor, na figura 3.14 podemos observar como são feitas as ligações entre sensores, resistores e a unidade de processamento.

Figura 29 - Ligação do sensor óptico



Fonte: Autoria própria

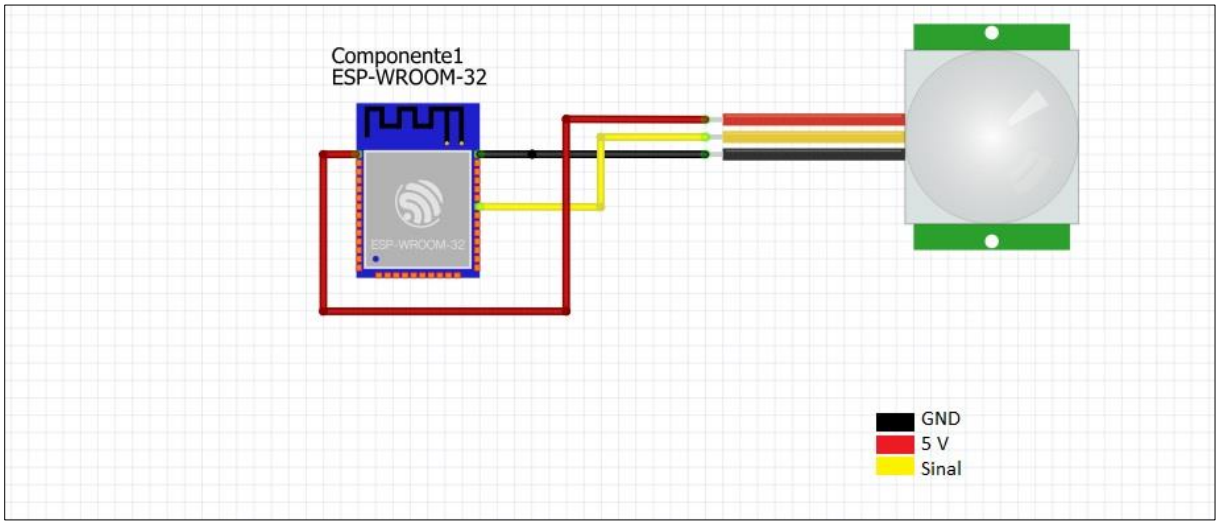
A implementação sensor PIR é simples pois ele possui apenas 3 pinos, alimentanção 5 V, GND e o sinal. O sensor é instalado em um local estratégico para que sua abertura de até 120º seja bem aproveitada, a figura 30 mostra implementação na maquete.

Figura 30 - Implementação do Sensor PIR

Fonte: Autor

A figura 31 mostra o diagrama de conexão do sensor PIR com a unidade de processamento.

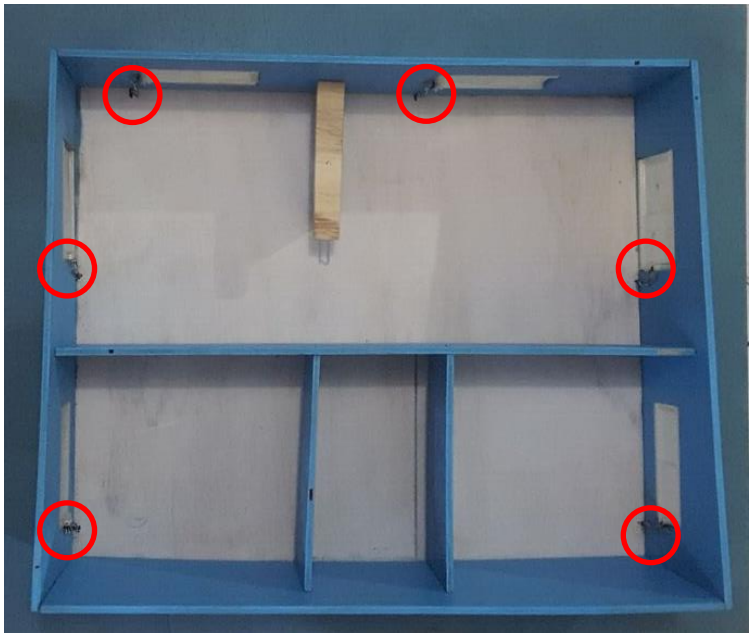
Figura 31 - Diagrama sensor PIR



Fonte: Autoria própria

Para que o sistema de sensores tivesse uma melhor conexão entre a unidade de processamento foi utilizado uma placa perfurada e na mesma foram soldados toda a fiação dos os sensores. A alimentação dos sensores foi unificada e a placa possui conectores de saída para a unidade de processamento. Assim o sistema de sensores foi finalizado, na imagem 32 podemos observa os sensores intalados a placa com a saída para central de processamento.

Figura 32 - Rede de sensores na maquete

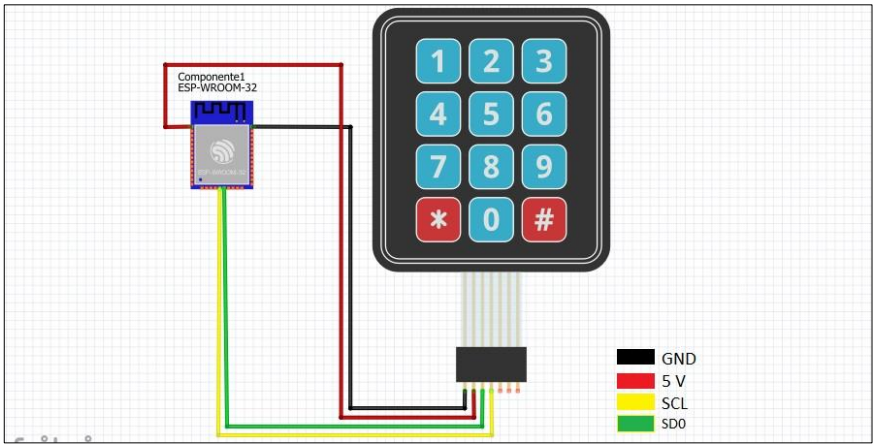


Fonte: Autoria própria

3.3.3.2 Controle Interno

O controle interno teve uma implementação simples, pois utiliza apenas dois componentes um teclado e um display LCD. O teclado para entrada de dados e controle, o LCD para mostrar informações para o usuário. O teclado possuí quatro linhas de conexão, no qual duas são para alimentação e duas para conexão serial. A figura 33 mostra a um diagrama de conexão entre teclado e o microcontrolador, mas o teclado na imagem é apenas ilustrativo, pois é diferente do que foi usado no projeto.

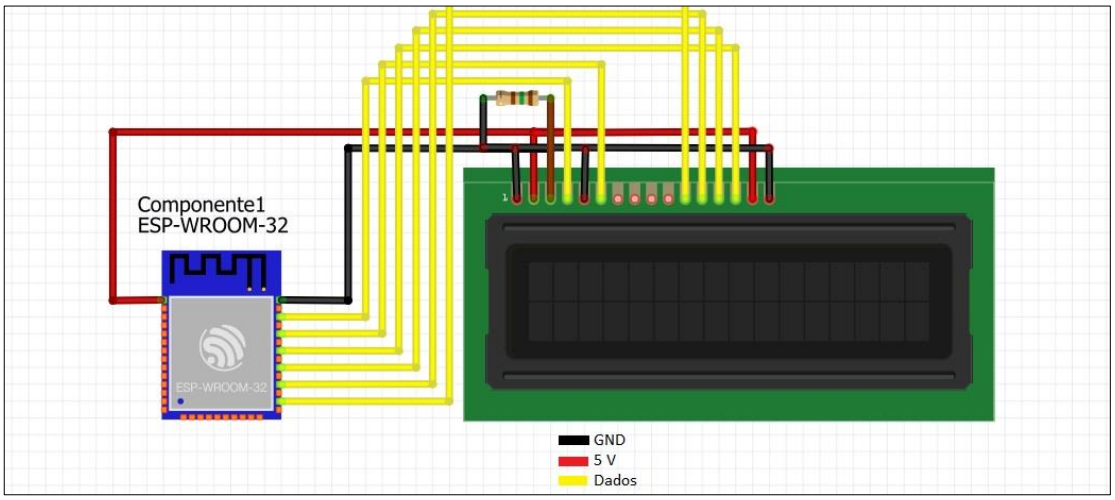
Figura 33 - Conexão teclado e ESP32



Fonte: Autor

O LCD possui uma conexão mais complexa quando se compara ao teclado, pois o mesmo possui linhas de dados e de alimentação, podemos observa na figura 34 as saídas de dados e de alimentação display.

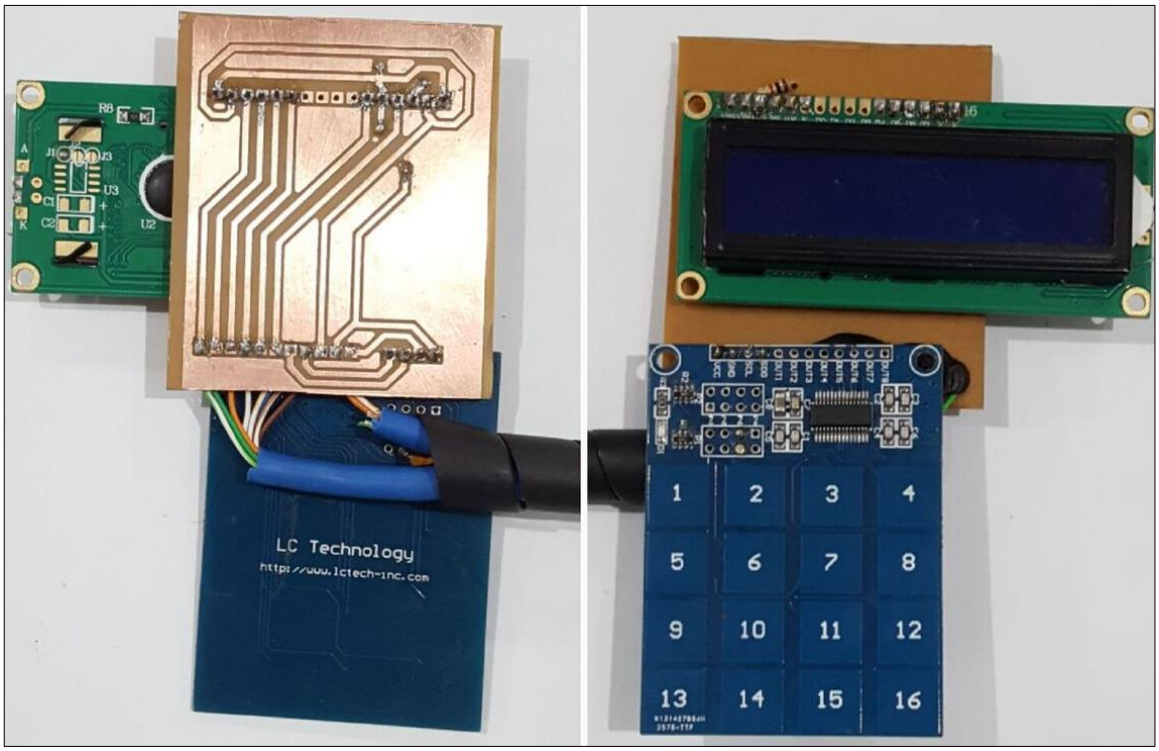
Figura 34 - Conexões LCD e ESP32



Fonte: Autoria própria

Para minimizar o número de linhas no barramento toda a alimentação do sistema de controle foi unificada, na figura podemos observar a placa do sistema completa e pronta para testes.

Figura 35 - Controle interno finalizado

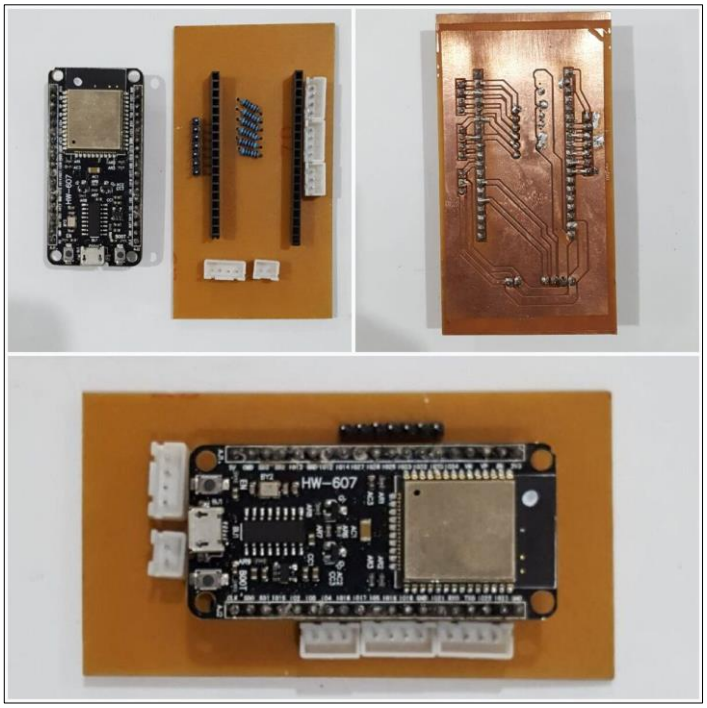


Fonte: Autoria própria

3.3.3.3 Central de Processamento

Para finalizar a montagem dos componentes na maquete, é necessário a implementação da central de processamento (ESP32). Para comunicação entre os periféricos foram utilizados conectores de forma que facilitasse a montagem e introdução da central no projeto. Utilizou-se cabos ligados aos periféricos posteriormente conectados a central. A figura 36 mostra como ficou e organização da unidade central.

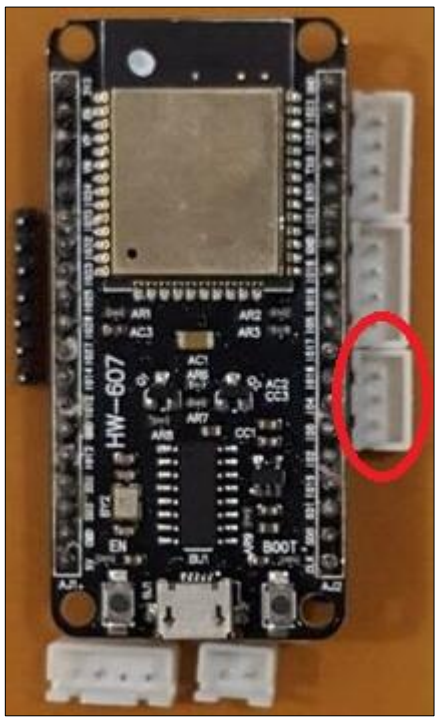
Figura 36 - Unidade Central de Processamento



Fonte: Autoriaa própria

Na figura 3.23 mostra a unidade de processamento e um conector de três pinos, esse conector tem a função de controlar o servo motor, fazendo a abertura do portão da residência.

Figura 37- Unidade Central de Processamento



Fonte: Autoria própria

Assim, finalizou-se a terceira etapa do desenvolvimento do projeto, a figura 38 mostra como ficou a maquete com todo o sistema pronto e montado. Então, neste ponto do projeto bastava desenvolver a aplicação de controle no Android Studio bem como seu código na plataforma arduino.

Figura 38 - Modelo finalizado

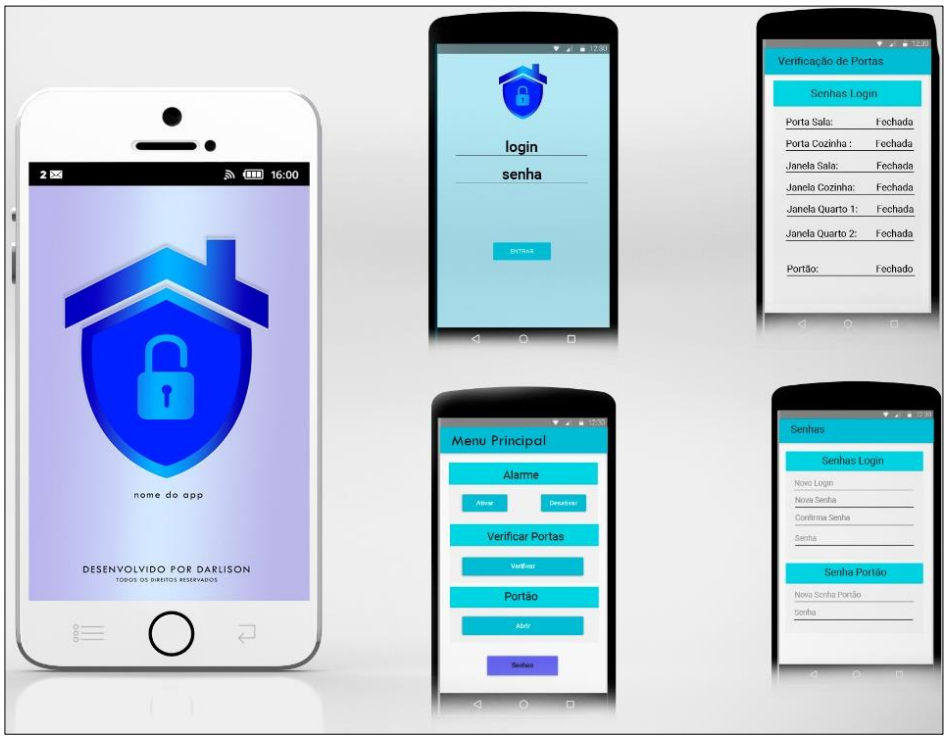
Fonte: Autoria própria

3.3.4 Criação do Aplicativo de Controle

Antes de iniciar a criação do aplicativo no Android Studio houve um levantamento de requisitos, nesse levantamento é discutido quais ferramentas o aplicativo pode ter. Inicialmente as ferramentas escolhidas para implementar são ferramentas de maior utilidade e necessidade em um sistema de segurança. O sistema precisa controlar o alarme, acesso de senhas, notificação para o usuário e como um bônus o app pode verificar o estado de portas e janela.

Seguindo a necessidade do sistema, um protótipo de telas foi criado, o mesmo possuindo as possíveis telas do App. A criação do aplicativo é feita com base no modelo do protótipo. Este possuí 5 telas, a primeira de login, a segunda principal e duas telas secundárias, uma para verificação de portas outra para redefinir senhas. A figura 39 mostra o protótipo de telas, assim complementando o entendimento.

Figura 39 - Protótipo de telas



Fonte: Autoria própria

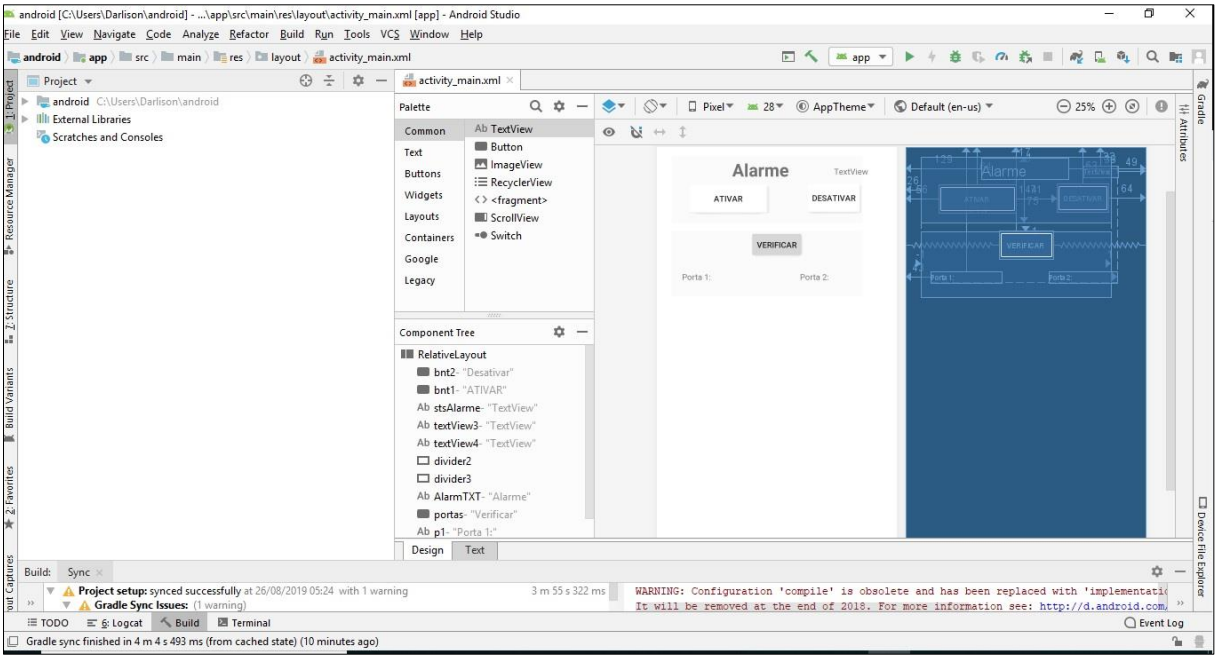
Para estabelecer como essa comunicação utilizado o Firebase Realtime que é um banco de dados não relacional do Google. As principais vantagens de usar esse banco são as seguintes:

- Possui toda uma estrutura de segurança e autenticação implementada, podendo desenvolver uma aplicação segura e confiável;
- Atualização de variáveis sem requisição;
- Versão gratuita sem custos;
- Conectividade com várias plataformas, inclusive a arduino que foi a utilizada.

A desvantagem do uso do Firebase é o a limitação da versão gratuita, que são um número limitado acessos e o limite de memória na nuvem. Porém o sistema desenvolvido não sofre com essas limitações, pois a maioria das variáveis são booleanas que ocupam pouco espaço, o aplicativo utiliza menos de 10% limite de acessos e requisições dessa versão.

Após o processo de prototipação de telas, para se criar o aplicativo que posteriormente é transferido para o celular que controlará a residência foi necessário primeiramente fazer o download gratuito do software Android Studio no site do desenvolvedor, depois de baixar e instalar a versão deste aplicativo compatível com o sistema operacional utilizado deu-se início a criação do app. A figura 40 mostra a tela inicial deste software.

Figura 40 - Criação do App no Android Studio



Fonte: Autoria própria

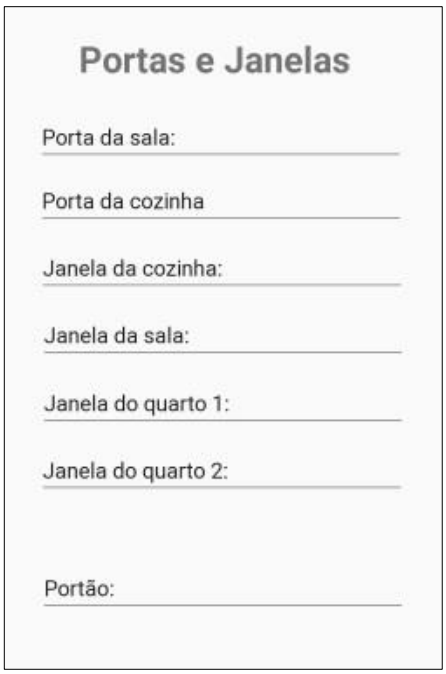
Primeiro passo no desenvolvimento é através da inteface de desenvolvimento foi implementado uma tela de login, sendo esta a primeira tela do aplicativo e posteriormento uma tela pricipal contendo as principais funções do App, essas funções são controle do Alarme, controle do portão, verificação de portas e senhas. O sistema é conectado ao banco de dados através de autenticação e para conexão do o ESP32 é gerado um código para uso do banco de dados, a figura 3.25 mostra a tela de login e a tela principal.

Figura 41 - Tela de principal e tela de loginl

Fonte: Autoria própria

Duas outras telas foram criadas dentro do programa, a primeira é a de verificação de portas/janelas esta é aberta ao clicar no botão verificar na tela principal, ela contém todas as portas e qual o status dela atual, mesmo que o alarme não esteja ativado essa função pode ser utilizada. A figura abaixo mostra a tela.

Figura 42 - Tela de verificação



Fonte: Autoria própria

Outro tela do App é a tela Senhas, que é aberta após clicar no botão “senhas” na tela principal, localizada no final da tela. Essa tela pode redefinir a senhar de login do usuário e a

senha de acesso ao portão da residência, para isso basta inserir nova senha, senha antiga e clicar no botão redefinir, a figura abaixo mostra essa Tela.

Figura 43 - Tela de Senhas

Fonte: Autoria própria

Feito isso bastou-se compilar o App e gerar o executável para um celular com sistema Android. Sendo assim, finalizou-se o desenvolvimento do protótipo de automação residencial proposto neste trabalho. O capítulo a seguir apresenta uma breve análise do código na arduino e a aplicação prática do modelo proposto, bem como o seu funcionamento

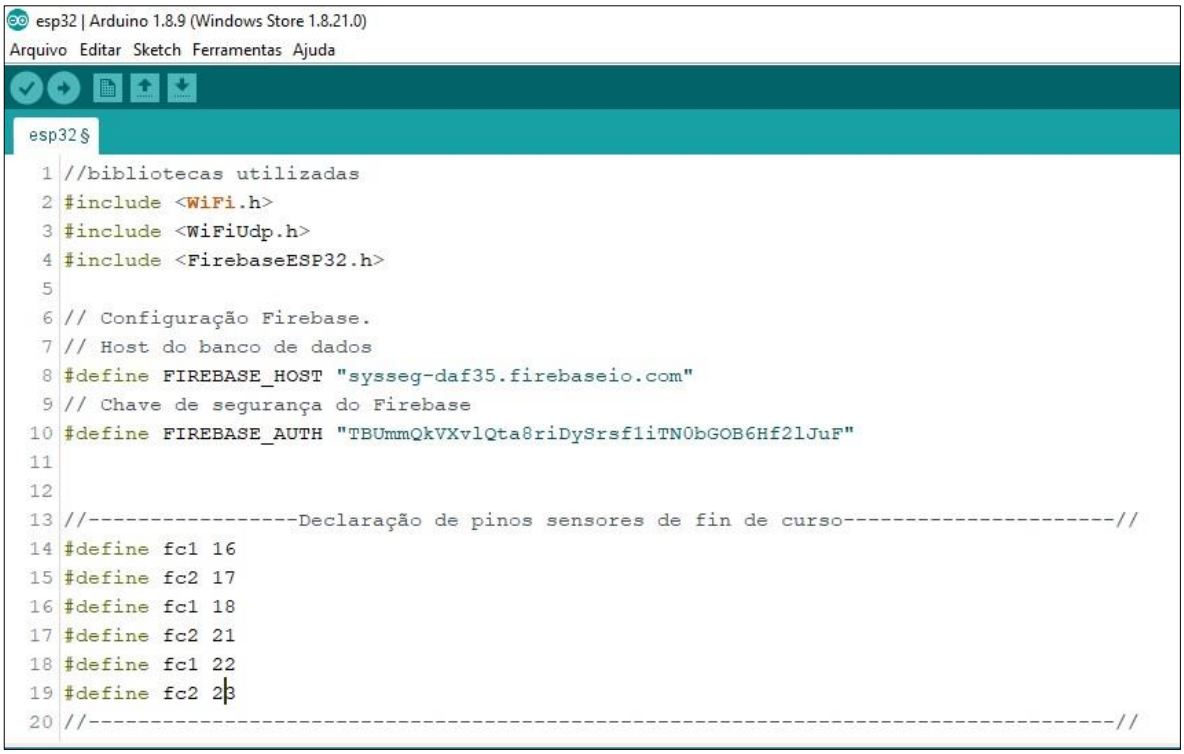
4 ANÁLISE DO CÓDIGO E APLICAÇÃO PRÁTICA DO MODELO PROPOSTO

Este capítulo apresenta uma breve análise de trechos código utilizado para esta aplicação. Vale ressaltar que todo o código que foi utilizado pode ser consultado no apêndice A deste trabalho.

4.1 Análise Geral do Código

Para desenvolver o código deste projeto foi necessário realizar as configurações de rede que o arduino receberá e definir a chave de conexão e configurações conexão com banco de dados. Então as primeiras linhas do *scketch* criado são referentes as bibliotecas que foram utilizadas e a definição das configurações de rede. Como pode ser observado na figura 41.

Figura 44 - Configurações inicias no scketch do arduino



Fonte: Autoria própria

Em seguida, foram definidos todos pinos que seriam utilizados. Em “void setup” foram inicializadas as configurações de rede e definido as funções que seriam executadas para cada comando recebido.

No loop principal do programa foram inseridas as linhas de código responsáveis pelo envio e recebimento dos status das variáveis que estão contidas no banco de dados e que estão sendo monitorado durante a execução do programa, são variáveis como status dos sensores e

sistema de alarme. Por fim, ao término do loop principal foram inseridas todas as funções que foram criadas para realizarem aa funcionalidades, como verificação do sensor, abertura do portão e alarme, que foram referenciadas nos comandos do aplicativo.

4.2 Descrição Aplicada do Modelo

Para o sistema de sensores foi utilizado sensores de movimento e sensores ópticos, ou seja, quando o usuário ativa o alarme seja pelo aplicativo ou pelo controle interno, após isso se um dos sensores detectar algum movimento ou abertura em alguma porta os sensores ativam o alarme será acionado, e consequentemente o buzzer será ativado emitindo sinal sonoro. O buzzer que foi utilizado neste projeto funciona com 5 V, por isso foi uma das saídas digitais, ou seja, o alarme ser acionado a central de controle envia uma saída alta para o buzzer, emitindo assim um sinal sonoro informando que o alarme foi disparado. Para aumentar a eficiência do sistema há a necessidade de um sistema de notificação além do acionamento do buzzer. Para que o mesmo usuário seja alertado que o sistema disparou.

Para resolver este problema foi implementado um sistema de notificação, quando o alarme é acionado e identifica uma invasão, além do buzzer entrar em funcionamento, a unidade central atualiza uma variável no banco de dados, esta variável aciona um sistema de notificação que informa o usuário da invasão.

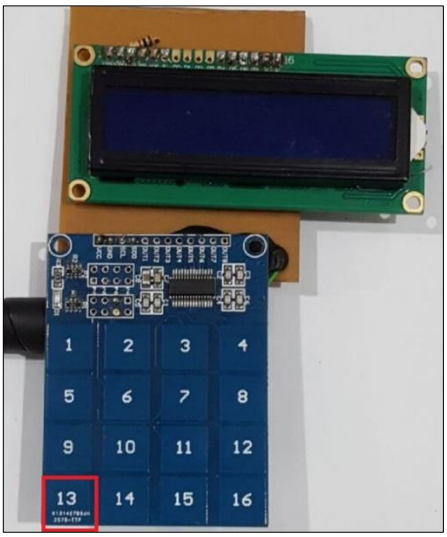
O sistema de controle do portão também se mostrou eficiente, seu funcionamento ocorreu como esperado, a abertura do portão pode ser feita através aplicativo na tela principal, que ocorre da seguinte maneira, após acionar o botão abrir envia um sinal para a unidade de processamento e a mesma controla o servo e e faz a abertura do portão, a imagem abaixo mostr.

Figura 45 - Aplicativo função abrir portão

Fonte: Autoria própria

Outra maneira de abertura do portão é através do controle interno que é bastante simples, se alarme estiver destivado basta tocar na tecla 13 que o portão será aberto, caso o usuário esteja no exterior da casa não possa acessar o aplicativo, basta inserir a senha de abertura do portão e o mesmo abrirá. Após o variados testes o sistema se mostrou eficiente e sem erros, a imagem a mostra o procedimento.

Figura 46 - Abertura do portão através do Controle Interno



Fonte: Autoria própria

Por fim, foi analisado o funcionamento protótipo do sistema de segurança residencial como um todo, o sistema mostrou muito útil e prático, que pode exercer a funções de segurança sem apresentar problemas. Com isso finalizando os testes e aplicação prática do modelo de segurança residencial proposto pelo projeto.

4.3 Custos do Modelo Proposto

Um dos objetivos do projeto foi desenvolver um protótipo de sistema de segurança residencial de baixo custo utilizando a plataforma livre arduino em conjunto com ESP32. Vale ressaltar que hoje a automação está a cada dia se tornando mais acessível, contudo a mesma ainda é restrita a usuários com alto poder aquisitivo. Projetos de automação que podem ser contratados por empresas já especializadas podem variar entre 10 e 50 mil reais. Segundo a AURESIDE, mesmo que não sejam apenas voltados para segurança podemos notar que os sistemas podem ter valores altos. Estes variam de simples automatização das luzes a complexos sistemas de automação residencial. A tabela abaixo apresenta os custos que foram gastos (sem o valor do frete) para desenvolver este protótipo de automação residencial, podendo futuramente lançar um produto bom e mais acessível no mercado, levando em consideração as melhorias que precisam ser feitas.

Tabela 3 - Custo do modelo proposto

Material	**Quantidade**	**Preço Unitário**	**Total**
ESP32	1 Un.	R$ 20,00	**R$ 20,00**
Maquete	1 Un.	R$ 50,00	**R$ 50,00**
Sensor óptico	6 Un.	R$ 1,50	**R$ 9,00**
Sensor PIR	2 Un.	R$ 10,00	**R$ 20,00**
Servo motor	1 Un.	R$ 5,00	**R$ 5,00**
Display 16x2	1 Un.	R$ 15,00	**R$ 15,00**
Teclado Capacitivo	2 Un.	R$ 10,00	**R$ 20,00**
Resistores	13 Un.	R$ 0,30	**R$ 4,00**
Total			**R$ 143,00**

5 CONSIDERAÇÕES FINAIS

O uso de Microcontroladores presentes no contexto de automação residencial, mostra-se uma ferramenta eficiente, além de uma implementação relativamente simples. Os benefícios foram amplos, entre eles destacam-se a boa velocidade de resposta no processamento dos dados. Este implica diretamente no funcionamento e desempenho do sistema. Por ser uma placa com wi-fi integrado, a conexão com a rede internet foi implementa, facilitando a conexão remota com o usuário, para fins de monitoramento na residência.

Embora limitando-se em alguns aspectos físicos, como números de portas disponíveis, restrições poderiam ocorrer no projeto. Se, por exemplo, o ambiente a ser implementado necessite de uma grande quantidade de sensores, isto pode se torna um entrave para projeto, uma vez que há a necessidade de reservar uma quantidade de portas mínimas para os periféricos (Display e teclado por exemplo). Problemas como esse podem ser contornados através da utilização de circuitos extensores, como por exemplo, demultiplexador ou multiplexador.

Outro ponto importante, foi a utilização do Android Studio no desenvolvimento do aplicativo que veio a controlar todo o sistema. Este mostrou-se extremamente viável e eficiente, pois possuí uma vasta gama de ferramentas para desenvolvimento de aplicativos, além de muitos materiais para aprendizagem na utilização do software. Aliado as ferramentas do Android Studio o Firebase foi de grande vantagem na construção do projeto, porque suas funções de segurança e atualização em tempo real já são inclusas no banco de dados Firebase.

Além disso, a aplicação desenvolvida mostra-se intuitiva e funcional. Esta afirmação foi comprovada durante a bateria de testes, e não apresentou falhas ou problemas de comunicação com o banco de dados, e com resultados praticamente em tempo real. Assim aplicação mostrou-se satisfatória para suas funcionalidades para o projeto. A princípio, não houve necessidade de correções ou implementação de novas ferramentas, porém podem ser implementadas novas funcionalidades, como por exemplo, implementação de câmeras processamento de imagem.

Um facilitador para implementação deste projeto é que ele se mostra financeiramente atrativo. Este fato dar-se-á relação custo-benefício do protótipo completo se comparado a um sistema disponível comercialmente. Ademais, a área de automação residencial está em crescente evolução, e a tendência é a utilização de sistemas mais robustos e com maior capacidade de processamento de dados, integrando o maior número possível de aplicações e fazendo com que o binômio custo-benefício tenha cada vez mais relevância.

Por fim, neste trabalho foi apresentado a elaboração do protótipo de um sistema de segurança residencial utilizando a plataforma de prototipagem arduino em conjunto com ESP32. Buscando aliar um baixo custo de investimento ao projeto que foi proposto. Então, a partir dos fatos relatados e resultados encontrados no decorrer do desenvolvimento e testes do modelo proposto, pode-se concluir que o objetivo do trabalho foi alcançado através dos métodos que foram desenvolvidos.

5.1 Trabalhos Futuros

Como sugestões para trabalhos futuros, consideram-se:

- Implementar um sistema de automação para automatizar toda a residência e não só o sistema de segurança. Submeter o sistema a testes de durabilidade. Podendo fazer controle de luzes, monitoramento de temperatura, controle eletrodomésticos, entre outras funções na área da automação.
- Utilização de mais um ESP32, visto que deste modo o sistema terá maior número de portas e maior capacidade de processamento de dados. Podendo através dessa substituição o sistema trabalhar com maior velocidade e possibilidade de implementar novas funcionalidades.
- Implementar um sistema mais complexo, podendo fazer uso de câmeras e processamento de imagens, além de recursos que ampliem a segurança.
- Tornar o protótipo um sistema comercializável.

REFERÊNCIAS

IPEA, **Instituto de Pesquisa Econômica Aplicada**. Disponível em: www.ipea.gov.br/. Acesso em: 09 de Maio de 2019.

AURESIDE, **Associação Brasileira de Automação Residencial.** Disponível em: www.aureside.org.br. Acesso em: 09 de Maio de 2019.

OLIVEIRA, J.P. **Domótica: Perspectiva da Plataforma Arduino.** 2012. Monografia (Conclusão de Curso) – Universidade Estadual de Goiás, Goianésia.

SENA, D.C.S. **Automação Residencial.** 2005. Monografia (Conclusão de Curso) – Universidade Federal do Espírito Santo, Vitória.

BOLZANI, C.A.M. **Análise de Desenvolvimento de uma Plataforma para Residências Inteligentes.** Tese (Doutorado em Engenharia Elétrica) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2010.

BOLZANI, C.A.M. **Análise de Desenvolvimento de uma Plataforma para Residências Inteligentes: Uma Introdução a Sistemas Domóticos.** Tese (Mestrado em Engenharia Elétrica) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2004.

BOLZANI, C.A.M. **Residências Inteligentes.** São Paulo: 1. ed. Livraria da Física, BOLZANI, C.A.M. **Análise de Desenvolvimento de uma Plataforma para Residências Inteligentes.** Tese (Doutorado em Engenharia Elétrica) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2010.

BOLZANI, C.A.M. **Análise de Desenvolvimento de uma Plataforma para Residências Inteligentes.** Tese (Doutorado em Engenharia Elétrica) – Escola Politécnica, Universidade de São Paulo, São Paulo, 2010.

MOYA, J. M. Huidobro; TEJEDOR, R. J. Milán. **Manual de Domótica.** 2010. Disponível em: < http://books.google.com.br/books?id=V6IzqqDcfF8C&printsec=frontcover&hl=pt-BR&source=gbs_ge_summary_r&cad=0#v=onepage&q&f=false>. Acesso em: 23 Julho 2019.

LAGUÁRDIA, Susy. **Automação residencial cresce no Brasil.** Disponível em: < http://www.aureside.org.br/imprensa/default.asp?file=62.asp >. Acesso em: 24 Julho 2019.

AZEVEDO, C. L.; PIZZOLATO, N. D.; **Aplicabilidades e Sistemas de Automação Residencial**. Disponível em: < http://essentiaeditora.iff.edu.br/index.php/vertices/article/viewFile/1809- 2667.20040015/86>. Acesso em: 16 de Junho 2019.

GRIFFITHS, Dawn; GRIFFITHS, **David. Head First**, Android Development. Sebastopol: O'Reilly, 2015.

MEIER, Reto. **Professional AndroidTM Application Development**. Indianapolis: Wiley, 2009.

TECHOPEDIA. **Android SDK**. Disponível em: <http://techopedia.com/definition/4220/android-sdk>.Acesso em: 26 de Julho de 2019.

DEITEL, Paul; DEITEL, Harvey; WALD, Alexander. **Android 6 for programmers**. 3a. ed. Ann Arbor: Prentice Hall, 2016.

CARVALHO, Suelen. **Android Studio: vantagens e desvantagens com relação ao Eclipse**. Disponível em: <https://imasters.com.br/mobile/android>. Acesso em: 06 de outubro de 2015.

ANDROID DEVELOPER. Página Oficial do **Android Studio Developer**. Disponível em: <https://developer.android.com/studio/intro?hl=pt-br>. Acesso em: 20 de Julho de 2019.

MINATEL, Pedro. **Imasters**. Disponível em:<https//imasters.com.br/open-hardware-2/os-primeiros-passos-com-o-esp32/> Acesso em: 04 de Julho de 2019.

MINATEL, Pedro. **Sistemas Embarcados e Internet das coisas**. Disponível em: <https//pedrominatel.com.br/pt/esp32/> Acesso em: 04 de set de 2019.

MORAIS, José. **Portal Vida de Silício**. Disponível em: <https//portal.vidadesilicio.com.br> Acesso em: 04 de Julho de 2019.

FIREBASE DATABASE. Disponível em: <https//Firebase.google.com/docs/datebase/>Acesso em: 15 de Julho de 2019.

ESPRESSIF. **Placas eletrônicas de desenvolvimento 2018a**. Disponível em: <https://www.espressif.com/en/products/hardware/development-boards>. Acesso em 14 de Julho de 2018.

ESPRESSIF. **ESP32 Datasheet 2018b**. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pd>. Acesso em 14 de Julho de 2019.

ESPRESSIF. **ESP-WROOM-32 Datasheet 2018c**. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp-wroom-32_datasheet_en.pdf>. Acesso em 14 de Julho de 2019.

VAGAPOV, Yuriy. **Comparative Analysis and Practical Implementation of the ESP32 Microcontroller Module for the Internet of Things**. 2017. Disponível em: <https://www.researchgate.net/publication/320273388_Comparative_Analysis_and_Practical_Implementation_of_the_ESP32_Microcontroller_Module_for_the_Internet_of_Things>. Acesso em: 14 de Julho de 2019.

KAMAMI. **Modelo da placa ESP-WROOM-32**. 2017. Disponível em:<https://kamami.pl/wycofane-z-oferty/563392-esp32-devkitc-plytka-rozwojowa-iot-z-modulem-wi-fi-esp-wroom-32-firmy-espressif.html> Acesso em: 19 de Julho de 2019.

USINAINFO. **Usina da Informática**. Disponível em: <https//www.usinainfo.com.br> Acesso em: 04 de Julho de 2019.

ANEXO A – Parte do código na IDE arduino

```
#include <WiFi.h>
#include <WiFiUdp.h>
#include <FirebaseESP32.h>

// Configuração Firebase.
// Host do banco de dados
#define FIREBASE_HOST "sysseg-daf35.firebaseio.com"
// Chave de segurança do Firebase
#define FIREBASE_AUTH "TBUmmQkVXvlQta8riDySrsf1iTN0bGOB6Hf2lJuF"



//-----------------Declaração de pinos sensores de fin de curso----------------------//
#define fc1 26
#define fc2 25
//-----------------------------------------------------------------------------------//

boolean tst = false;

FirebaseData firebaseData;

const char* ssid = "Leonardo";
const char* password = "86234528";

void setup() {
  //-----------------Declaração de váriaveis de sensores de movimento------------------//
  pinMode(fc2, INPUT);                                          //
  pinMode(fc1, INPUT);                                          //
  //-----------------------------------------------------------------------------------//
  Serial.begin(115200);
  Serial.println("Booting");
  WiFi.mode(WIFI_STA);
  WiFi.begin(ssid, password);
  while (WiFi.waitForConnectResult() != WL_CONNECTED) {
    Serial.println("Connection Failed! Rebooting...");
    delay(5000);
   // ESP.restart();
  }
  Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
```

```
  Firebase.reconnectWiFi ( true );
  Firebase.setMaxRetry (firebaseData, 3 );
  Serial.println("Em conexão");
}

void loop() {
  Serial.println("atualizando sensores");
  if (Firebase.getBool(firebaseData, "SensorPJ")) {

   // if (firebaseData.dataType() == "boolean")) {
      Serial.println(firebaseData.boolData());
      tst = firebaseData.boolData();
    //}
  } else {
    Serial.println(firebaseData.errorReason());
  }

  if (tst == true){
    verif();
  }
  else {
    Serial.println("SEM pedido de verificação");
  }

  delay(500);
}

void verif (){
    // Definindo variaveis para leitura dos sensores de fim de curso
    int sf1 = digitalRead(fc1);
    delay(80);

    // Sensor 1
    if (sf1 == HIGH){
      Serial.println("Porta 1 Fechada");
      Firebase.setBool(firebaseData,"porta1", true);
    }
    else if(sf1 == LOW){
      Serial.println("Porta 1 Aberta");
      Firebase.setBool(firebaseData,"porta1", false);
```

```
  }

  int sf2 = digitalRead(fc2);
  delay(80);

  // Sensor 1
  if (sf2 == HIGH){
   Serial.println("Porta 2 Fechada");
   Firebase.setBool(firebaseData,"porta2", true);
  }
  else if(sf2 == LOW){
   Serial.println("Porta 2 Aberta");
   Firebase.setBool(firebaseData,"porta2", false);
  }
  Firebase.setBool(firebaseData,"SensorPJ", false);
  return ;
}
```

ANEXO B – Parte do código no Android Studio

```
6   package com.example.darlison.appsisseg;
    /*service cloud.firestore {
      match /databases/{database}/documents {
        match /{document=**} {
          allow read, write;
        }
      }
    }regras iniciais do banco de dados*/

    import android.content.Intent;
    import android.support.v7.app.AppCompatActivity;
    import android.os.Bundle;
    import android.view.View;
    import android.widget.Button;
    import android.widget.TextView;

    import com.google.firebase.database.DataSnapshot;
    import com.google.firebase.database.DatabaseError;
    import com.google.firebase.database.DatabaseReference;
    import com.google.firebase.database.FirebaseDatabase;
    import com.google.firebase.database.ValueEventListener;


    public class MainActivity extends AppCompatActivity {


        private DatabaseReference databaseReference =
    FirebaseDatabase.getInstance().getReference();
        private DatabaseReference alarmReference =
```

```
databaseReference.child("Alarme");
    private DatabaseReference senFcReference1 =
databaseReference.child("SensorPJ");
    private DatabaseReference p1Ref =
databaseReference.child("porta1");
    private DatabaseReference p2Ref =
databaseReference.child("porta2");
    private Boolean a;
    private Boolean p01;
    private Boolean p02;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        Button bt1 = (Button) findViewById(R.id.bnt1);
        Button bt2 = (Button) findViewById(R.id.bnt2);
        Button port = (Button) findViewById(R.id.portas);
        final TextView stAlarm = (TextView)
findViewById(R.id.stsAlarme);
        final TextView port1 = (TextView) findViewById(R.id.p1);
        final TextView port2 = (TextView) findViewById(R.id.p2);

        databaseReference.addValueEventListener(new
ValueEventListener() {
                    public void onDataChange(DataSnapshot dataSnapshot)
{
                        Boolean b = (Boolean)
dataSnapshot.child("Alarme").getValue();
                        a = b;
                        //stAlarm.setText(String.valueOf(a));
                        if(b) {
                            //boo.setText("Alarme True");
                            stAlarm.setText("(Ativado)");
                        }
                        else{
                            //boo.setText("estava False");
                            stAlarm.setText("(Desativado)");
                        }
                        Boolean p02 = (Boolean)
dataSnapshot.child("porta2").getValue();

                        //stAlarm.setText(String.valueOf(a));
                        if(p02) {
                            //boo.setText("Alarme True");
                            port2.setText("Porta 2:Fechada");
                        }
                        else{
                            //boo.setText("estava False");
                            port2.setText("Porta 2:Aberta");
                        }

                        Boolean p01 = (Boolean)
dataSnapshot.child("porta1").getValue();

                        //stAlarm.setText(String.valueOf(a));
                        if(p01) {
                            //boo.setText("Alarme True");
                            port1.setText("Porta 1:Fechada");
                        }
                        else{
```

```
                            //boo.setText("estava False");
                            port1.setText("Porta 1:Aberta");
                        }

                    }
                    public void onCancelled(DatabaseError
databaseError) {}
                });


        bt1.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                alarmReference.setValue(true);
                //senFcReference1.setValue(false);
                stAlarm.setText("(Ativando)");

                /*databaseReference.addValueEventListener(new
ValueEventListener() {
                    public void onDataChange(DataSnapshot dataSnapshot)
{
                        Boolean b = (Boolean)
dataSnapshot.child("LEDStatus").getValue();
                        if(b) {
                            text1.setText("Alarme ativado");
                            boo.setTag(b);
                        }
                        else{
                            boo.setTag(b);
                        }

                    }
                    public void onCancelled(DatabaseError
databaseError) {}
                });*/
                Intent intent = new Intent (MainActivity.this,
ActivityVerif.class);
                startActivity(intent);
            }
        });
        bt2.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                alarmReference.setValue(false);
                //senFcReference1.setValue(true);
                stAlarm.setText("(Desativando)");

            }
        });
        port.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                senFcReference1.setValue(true);



            }
        });
```

```
        }
    }
```