UNIVERSIDADE FEDERAL RURAL DO SEMI-ÁRIDO
PRÓ-REITORIA DE GRADUAÇÃO
CAMPUS CARAÚBAS
INTERDISCIPLINAR EM CIÊNCIA E TECNOLOGIA

ÉRIKA ISABEL DA COSTA

DESENVOLVIMENTO DE UM PROTÓPIPO PARA AUTOMAÇÃO RESIDENCIAL USANDO DISPOSITIVOS CONTROLADOS VIA SMARTPHONE.

CARAÚBAS – RN

2019

ÉRIKA ISABEL DA COSTA

DESENVOLVIMENTO DE UM PROTÓPIPO PARA AUTOMAÇÃO RESIDENCIAL USANDO DISPOSITIVOS CONTROLADOS VIA SMARTPHONE.

Trabalho de Conclusão de Curso apresentado ao curso de Interdisciplinar em Ciência e Tecnologia da Universidade Federal Rural do Semi-Árido (UFERSA), como parte dos requisitos para obtenção do Título de Bacharel em Ciência e Tecnologia.

Orientador: Prof. Dr. Zenner Silva Pereira.

CARAÚBAS – RN

2019

Cd Costa, Érika Isabel da.

Desenvolvimento de um protótipo para automação residencial usando dispositivos controlados via smartphone / Érika Isabel da Costa. – 2019.

59 f. : il.

Orientador: Zenner Silva Pereira.

Monografia (graduação) - Universidade Federal Rural do Semi-árido, Curso de Interdisciplinar em ciência e tecnologia, 2019.

1. Arduino. 2. App inventor. 3. Automação. 4. Smartphone. 5.Aplicativo. I. Pereira, Zenner Silva, orient. II. Título.

ÉRIKA ISABEL DA COSTA

DESENVOLVIMENTO DE UM PROTÓPIPO PARA AUTOMAÇÃO RESIDENCIAL USANDO DISPOSITIVOS CONTROLADOS VIA SMARTPHONE.

Trabalho de Conclusão de Curso apresentado ao curso de Interdisciplinar em Ciência e Tecnologia da Universidade Federal Rural do Semi-Árido (UFERSA), como parte dos requisitos para obtenção do Título de Bacharel em Ciência e Tecnologia.

APROVADO EM: 08/08/2019

[redacted]

Prof. Dr. Zenner Silva Pereira - UFERSA
Presidente

[redacted]

Prof. Dr. José Junior Alves da Silva- UFERSA

Membro

[redacted]

Profª. Ma. Ana Tereza de Abreu Lima- UFERSA

Membro

Dedico a minha família, por sempre acreditarem em mim.

AGRADECIMENTOS

Inicialmente agradeço a Deus por sempre está comigo, e por me dar forças para continuar.

Agradeço a minha família, em especial aos meus pais Edineuza e Joaozinho, e ao meu irmão Érico por todo apoio e incentivo.

Agradeço a minha segunda família: Poliana, Laynara, Hortência, Joice, Mikaelly, Mateus, Sandin, Roberto e Paulo Walter por todo apoio, e por estarem sempre presentes em minha vida.

Agradeço a minhas amigas que sempre acreditaram em mim: Eliandra, Stefani e Sheila.

Agradeço em especial a Abraão por me ajudar e acreditar em mim, me incentivando e estando sempre presente.

Por fim, agradeço ao meu orientador Zenner, por todo o aprendizado durante a realização do presente trabalho.

Obrigada!

“Consagre ao Senhor tudo o que você faz, e os seus planos serão bem-sucedidos. Provérbios 16.3.”

Bíblia Sagrada.

RESUMO

A automação residencial vem se destacando cada vez mais ao longo dos anos, por apresentar aos usuários economia, segurança e conforto. O presente trabalho tem como objetivo criar um protótipo de automação residencial que apresente baixo custo e eficiência. Para a criação do protótipo é utilizado a plataforma Arduino juntamente com o sistema Android para viabilizar a ligação entre o *smartphone* e o controle. O Arduino é uma plataforma que tem funcionamento semelhante a um computador, com o qual é possível criar sistemas automáticos com uso de linguagem de programação. O *App inventor* é a ferramenta que foi utilizada para criação de um aplicativo para smartphones com sistema operacional Android. Com o auxílio de alguns componentes eletrônicos, de um Arduino e da plataforma *App inventor,* o presente trabalho tem como proposta criar um aplicativo que seja capaz de controlar lâmpadas, que além de ligar via smartphone, também poderá ser controlada de forma convencional por um interruptor, com o intuito de proporcionar um sistema de automação de baixo custo que apresente resultados satisfatório, gerando segurança e conforto para os usuários.

PALAVRAS-CHAVE: Arduino*, App inventor*, automação, *smartphone* e aplicativo.

ABSTRACT

Home automation has become increasingly prominent over the years, as it presents users with economy, safety and comfort. The present work aims to create a prototype of home automation that presents a low cost and efficiency. To create the prototype, the Arduino platform is used together with the Android system to enable the connection between the smartphone and the control. Arduino is a platform that works similar to a small computer, with which it is possible to create automatic systems using programming language. The inventor App is the tool that will be used to create an application for smartphones with Android operating system. With the help of some electronic components, an Arduino and the inventor App platform, the present work aims to create an application that can control lamps, which besides connecting via smartphone, can also be controlled conventionally by a light switch. Inorder to provide a low cost automation system that presents satisfactory results, generating safety and comfort for users.

KEYWORDS: Arduino, App inventor, automation, smartphone and application.

LISTA DE FIGURAS

LISTA DE ABREVIATURAS E SIGLAS

App - Aplicativo

MIT - Massachusetts institute of technology

QR code - Quick response

Add – Adicionar

IDE - Ide Integrated Development Environment

PWM - Pulse Width Modulation (Modulação por Largura de Pulso)

V - Volts

LED - Light Emitting Diode

USB - Universal Serial Bus

Rx – Recepção/Upload

Tx - Transmissão/Download

mA – miliampere

GND - Graduated neutral density filter

VCC - Voltagem em corrente contínua

Sumário

1 INTRODUÇÃO

A tecnologia vem avançando cada vez mais ao longo dos anos, com o intuito de proporcionar aos indivíduos uma melhor qualidade de vida. A automação é uma tecnologia que tem como objetivo automatizar, isto é, tornar automático tarefas que comandam ou controlam empresas, casas, entre outros.

A automação residencial torna automático as tarefas comuns do cotidiano de uma residência. É também conhecida por domótica, que resulta da palavra latina "*Domus*" que significa casa, junto com a palavra "Robótica" que significar automatizar ou controlar algo (BEGHINI,2013).

A automação residencial apresenta diversas vantagens, com ela é possível reduzir alguns problemas comuns como o alto consumo de energia. Pois o morador, mesmo ausente, pode ter controle da iluminação, e de aparelhos elétricos e eletrônicos da residência. Apresenta vantagens também na segurança com o acionamento de alarmes, câmeras de vigilâncias, entre outras tarefas.

Acionar uma lâmpada, um irrigador, abrir um portão, ligar um ar condicionado, entre outras tarefas comuns, podem ser facilmente realizadas e controladas com o uso dessa tecnologia. Existem diversas maneiras de controlar as tarefas desejadas, podendo ser por um computador, controle remoto ou *smartphone*.

No presente trabalhando, a ideia é criar um protótipo com objetivo de controlar lâmpadas via *smartphone* com sistema operacional Android e um interruptor simultaneamente. Para fazer a comunicação com o hardware via *smartphone*, foi criando um aplicativo na plataforma desenvolvida pela Google*: App Invetor.*

O *App inventor* é uma interface capaz de programar aplicativos de forma simples, sem a necessidade de conhecimentos avançados em programação. A interface é dividida em duas partes, a *App Inventor Designer* é onde o aplicativo será produzido, inserindo imagens, botões, e entre outros, criando o formato no qual ficará no smartphone. Já a parte *App Inventor Blocks Editor*, é onde o código da programação é criado, associando ações aos itens selecionados no *designer,* criando uma programação com blocos que se encaixam como peças de quebra-cabeça.

Além do *App inventor*, foi utilizando a plataforma Arduino, que é um controlador *open source*, que recebe informações, processa e retorna a saída (VIEIRA, 2011). Nele há um

microcontrolador que é usado para controlar as tarefas desejadas. A programação é feita por interface de desenvolvimento integrado, utilizando a linguagem própria do Arduino que é semelhante a linguagem C++. A comunicação entre o *smartphone* e um módulo *bluetooth* conectado ao Arduino é o que possibilita essa interação. Dessa forma, é possível controlar o sistema via *smartphone*. Para montar o presente protótipo alguns componentes eletrônicos são importantes para auxiliar na realização das tarefas, como relés, lâmpadas, *bluetooth*, resistores, interruptores, entres outros que serão apresentados.

2 OBJETIVO

2.1 Objetivos gerais

O objetivo do presente trabalho é criar um protótipo de automação residencial e entender o seu funcionamento. A ideia é proporcionar um sistema de controle lâmpadas via interruptor e via smartphone simultaneamente.

2.2 Objetivos específicos

- Criar um aplicativo utilizando a plataforma *App Inventor* para controlar lâmpadas via smartphone.
- Com o auxílio da plataforma Arduino e alguns componentes eletrônicos, produzir um sistema de automação residencial.
- Construir o protótipo para demonstração e para buscar comprovar que o mesmo além de apresentar um sistema de baixo custo, apresenta também resultados satisfatórios.

3 REFERÊNCIAL TEÓRICO

Neste capítulo serão abordados conceitos necessários para a aplicação do protótipo de automação residencial controlados via *smartphone.*

3.1 Automação

A automação vem desde a pré-história, quando o homem tentava automatizar suas tarefas. Em 1950, a automação se tornou mais popular, e caracterizava a movimentação automática de materiais. Logo então, os computadores e controladores passaram a fazer parte da tecnologia da automação a partir do século XX, sendo até hoje os computadores e controladores as principais bases da automação *(*Murrelektronik, 2018).

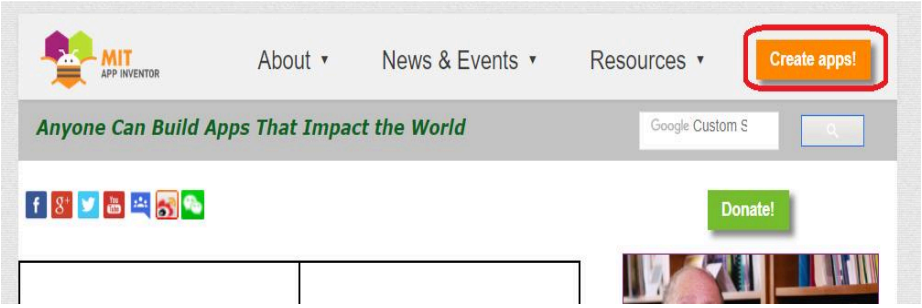
A automação residencial pode ser definida como a ideia de tornar tarefas comuns de uma residência automáticas (Foneplan, 2015). A automação propõe controlar serviços e equipamentos comuns do cotidiano, com o intuito de proporcionar conforto, praticidade e segurança para os que a utilizam.

3.2 App Inventor

O *App inventor* é uma ferramenta que possibilita a criação de aplicativos de forma simples. Nele há uma interface de programação gráfica que pode ser facilmente programada. Originalmente foi disponibilizada pela Google, mas em dezembro de 2011, o MIT (*Massachusetts Institute of Technology*) passou a ser responsável por manter a plataforma (BEGHINI,2013). Os aplicativos criados na plataforma são apenas para dispositivos com sistema operacional Android.

Para a criação do aplicativo é necessário obter uma conta no gmail, nessa conta ficam salvo todos os aplicativos criados. Para dar início a criação do aplicativo basta acessar o site oficial do *App Inventor* e clicar no botão “create apps!” apresentado na figura 1 a seguir:

Figura 1 - Botão create apps!



Fonte: site oficial do *App inventor*

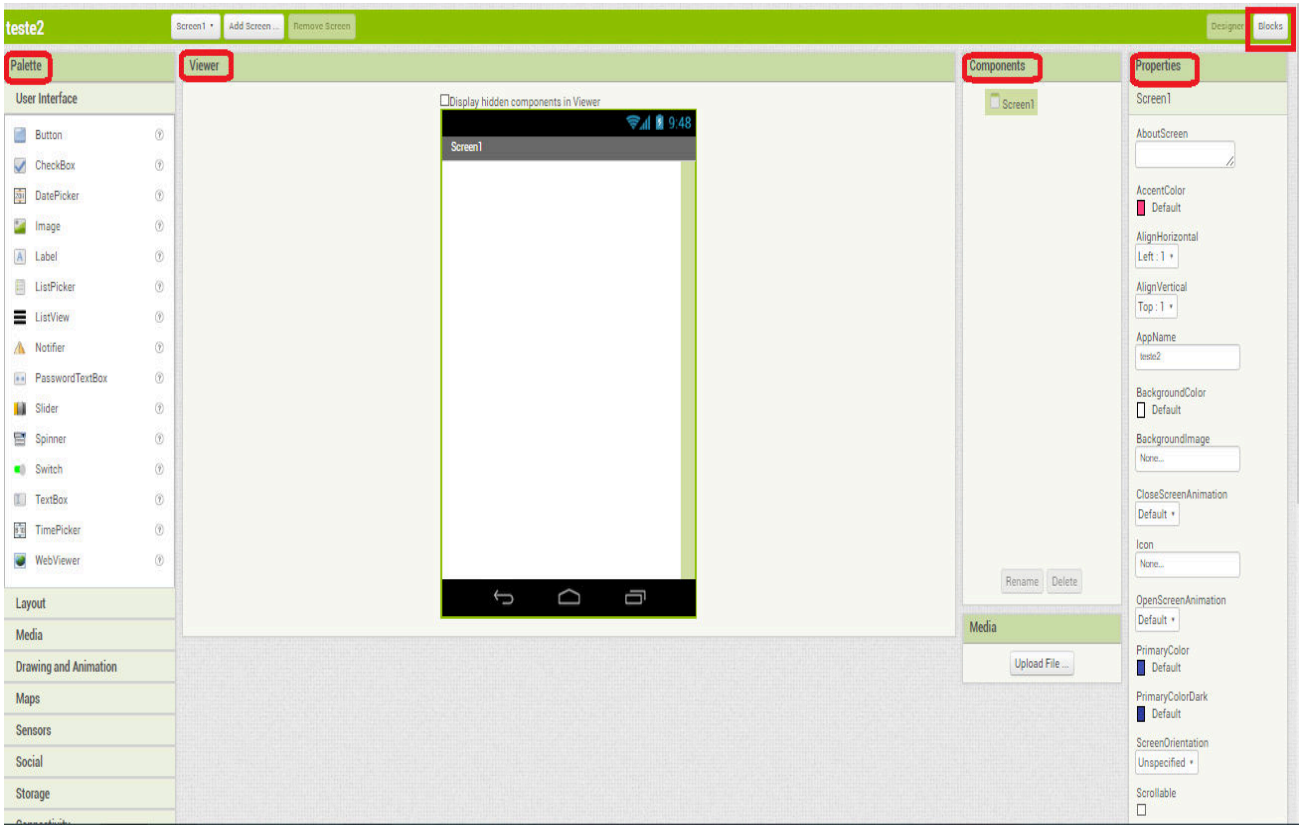
O App inventor é dividido em duas partes: *App Inventor Designer e o App Inventor Blocks Editor*, cada uma com uma função específica.

3.2.1 App Inventor Designer

A parte *designer* aparece logo após clicar no botão *"create apps!"*. Nessa parte é produzido o *designer* do aplicativo, ou seja, é feito o desenho e formato do aplicativo, adicionando imagens, botões, textos, *checkboxes* e entre outras opções que a plataforma disponibiliza.

A parte do *designer* é dividida em quatro colunas: *palette* (paleta), *viewer* (visualizador), *componentes* (componentes), *properties* (propriedades). Tais elementos estão circulados e apresentadas na figura 2:

Figura 2 - App inventor designer



Fonte: site oficial do *App inventor*

- *Palette* (paleta).

Nessa coluna ficam os componentes que podem ser utilizados no aplicativo. São divididas em sessões para facilitar a criação e a localização dos componentes como botões, imagens, textos e outros. Para utilizar qualquer componente dessa coluna é necessário arrastá-lo até a coluna *"viewer"* (visualizador).

- *Viewer* (Visualizador).

A presente coluna é onde os componentes são inseridos e organizados da forma que o criador desejar. Nessa coluna será apresentado como o aplicativo ficará na tela do *smartphone.*

- *Components* (Componentes)

Os itens adicionados da *"Palette"* ao *"Viewer"* são apresentados na coluna *"Components"* (Componentes). Dessa forma, fica mais fácil configurar cada componente, pois eles ficam listados de forma ordenada. Nessa coluna também é possível renomear os itens adicionados, como por exemplo um aplicativo com dois botões, pode-se nomear o primeiro botão "botão1" e o outro "botão2". Dessa maneira, facilita a criação da programação do aplicativo. Clicando no botão *"Add"* (presente na parte inferior da coluna), é possível importar fotos, sons e vídeos do computador para o *App inventor*, tudo que é importado estará disponível para ser utilizado no aplicativo.

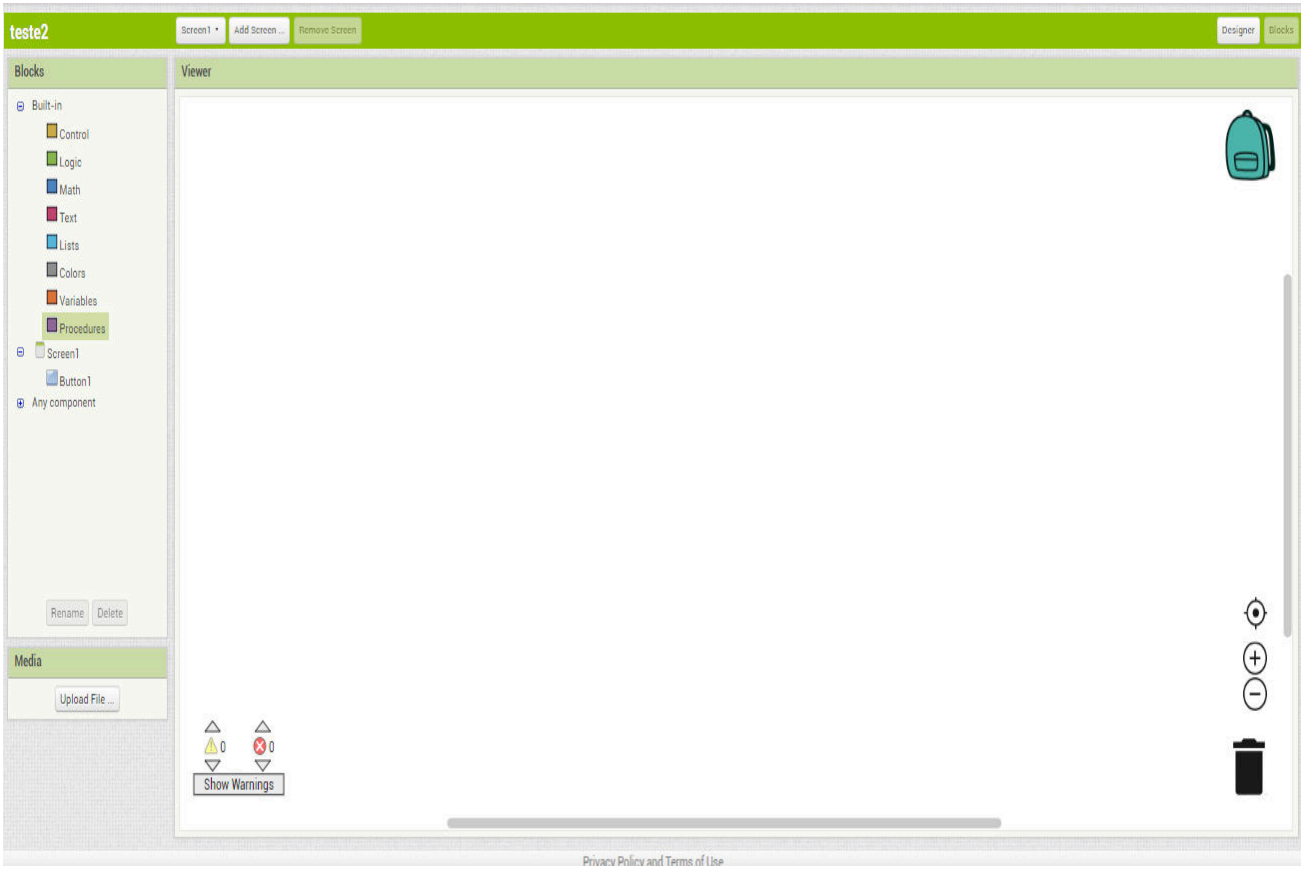
- *Properties* (propriedades)

Quando qualquer item da coluna *"Components"* é clicado, a coluna propriedades permite editar os detalhes sobre esse item. Nessa coluna é possível definir o tamanho, formato, cores e outras propriedades dos textos, botões, caixas de informações e qualquer outro componente adicionado. Essas configurações são vistas na coluna *"viewer"*, permitindo uma visualização das mudanças que estão sendo feitas na interface do aplicativo.

3.2.2 App Inventor Blocks Editor

Após a criação do design do aplicativo, o próximo passo é atribuir funções aos itens adicionados, dando início a programação do aplicativo. Para iniciar, é preciso clicar o botão *"Blocks"* (blocos), que leva a tela apresentada na figura 3:

Figura 3 - App inventor Blocks Edit

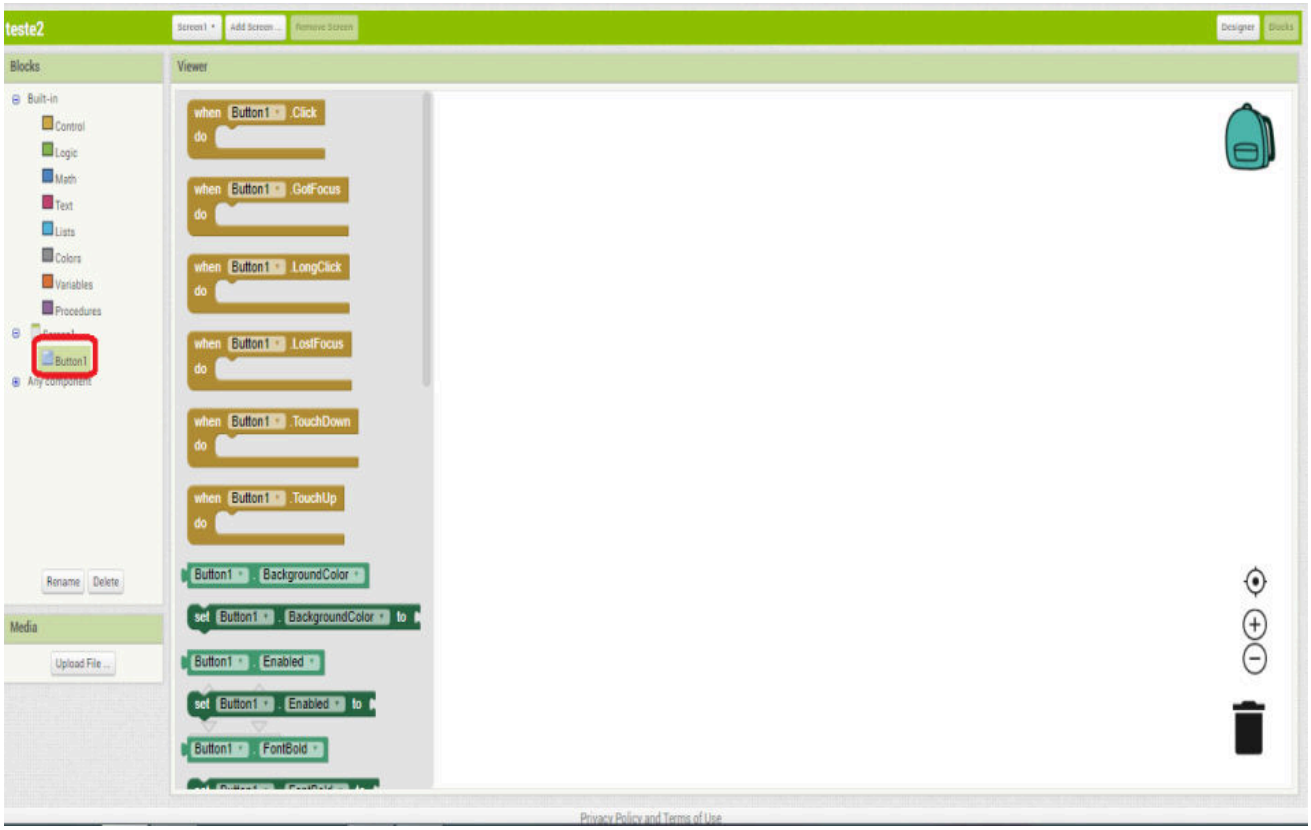


Fonte: site oficial da *App inventor*

Essa parte é onde o código será criado, e com ele é possível associar ações para os itens selecionados no designer. O lado esquerdo da figura 3 apresenta uma coluna dividida em duas partes: *"Built- in"* (onde ficam os blocos) e *"screen1"* (onde ficam os componentes inseridos em *"viewer").* É através dos blocos que a programação é feita, funcionando como uma espécie de "quebra-cabeças" que encaixa as ações e funções a cada componente inserido anteriormente no design.

Todos os itens adicionados no *design* aparecem de forma ordenada na parte *"screen1"* abaixo dos comandos *control, logic, math, text, lists, colors, variables e procedures* presentes em *"Built- in"*. Para associar alguma ação ao item desejado, basta clicar no componente e os blocos com as possíveis combinações apareceram. Por exemplo, caso um botão seja inserido, ao clicar no componente será apresentado as possibilidades de combinações que pode se fazer com o mesmo. Os blocos de combinações são apresentados na figura 4.

Figura 4 - Blocos de combinações para o botão



Fonte: site oficial do *App inventor*

Os comandos *control, logic, math, text, lists, colors, variables e procedures*, servem para auxiliar nas combinações dos componentes, sendo que cada um apresenta funções diferentes.

Control (controle) – Esses blocos servem pra controlar ações de um determinado componente. Como por exemplo a função “if then” (se, então ...), que significa: se um determinado componente estiver em determinado estado, então será atribuído algo a ele.

Logic (Lógica) – Os blocos de lógica são usados para dizer que algo é verdadeiro ou falso, quando uma função é igual a outra e entre outras funções lógicas.

Math (Matemática) – Esses blocos são usados para somas, multiplicações e outras operações matemáticas.

Text (texto) – Blocos para inserir textos, comparar textos e entre outros.

Lists (listas) – Blocos para criar listas, adicionar algo a listas e entre outros.

Colors (cores) – Blocos de cores

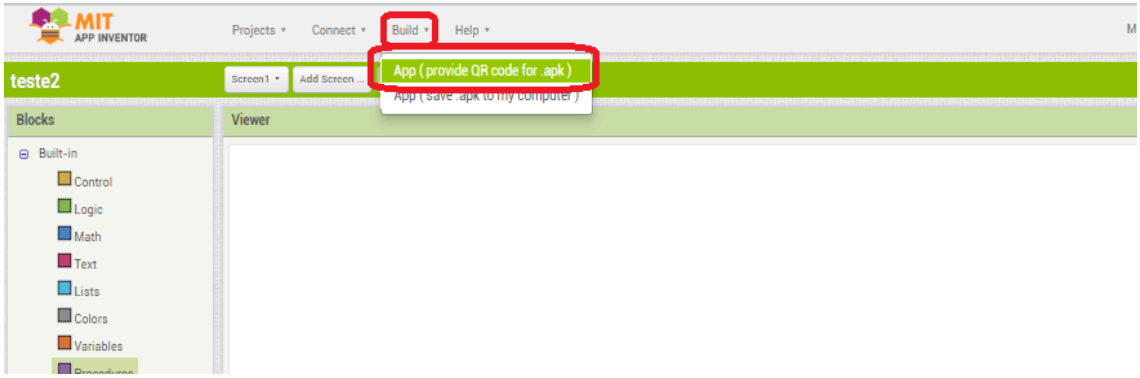
Variables (Variáveis) – Blocos para criar variáveis (algo que pode reter ou representar um valor).

Procedures (Procedimentos) – Determinar resultados a um procedimento.

Após criar o *design* e associar ações aos componentes do aplicativo, é necessário gerar um QR *code* no site do *App Inventor*. Para obter o aplicativo no smartphone, é necessário baixar o aplicativo "*MIT AI2 Companion*" ou um leitor de QR *code* na *Play store* do *smartphone*, para que seja possível ler o código criado. Após a leitura, o aplicativo será instalado no *smartphone*.

Para gerar o QR *code* basta clicar em *"build"* e em seguida "App(provide QRcode for .apk)" e o QR *code* será criado, como mostra a figura 5 a seguir:

Figura 5 - Como gerar QR code no App inventor



Fonte: site oficial do *App inventor*

Para controlar as lâmpadas e manter uma comunicação com o aplicativo pode ser utilizado um microcontrolador, ou algo como um pequeno computador que sirva para auxiliar no projeto. Como a ideia é proporcionar um sistema de baixo custo, o Arduino pode se encaixar no presente protótipo, por apresentar um bom preço e ser uma ferramenta de qualidade.

3.3 Arduino

O Arduino foi criado por Massimo Banzi, David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis, com o intuito de criar um dispositivo de baixo custo, preciso e fácil de programar. De forma que fosse acessível a estudantes, projetistas ou simplesmente amadores. A plataforma apresenta várias aplicações e opções para a criação de diversos projetos (THOMSEN, 2014).

O Arduino funciona de forma semelhante a um pequeno computador, com ele é possível criar sistemas automáticos interpretando as entradas e saídas da placa. É dividido em duas partes: o *hardware* (parte física do sistema) e o *firmware* (software específico). O *hardware* é a placa do Arduino, para controlá-la é necessário programá-la utilizando o *firmware,* que é a

plataforma IDE, onde é criado um código usando linguagem de programação própria do Arduino. Esse código é compilado e enviado para a placa fazendo com realize as tarefas descritas no código.

Existem diversos tipos de placas do Arduino, cabe ao criador do projeto decidir qual a melhor para se utilizar. Alguns exemplos são as placas:

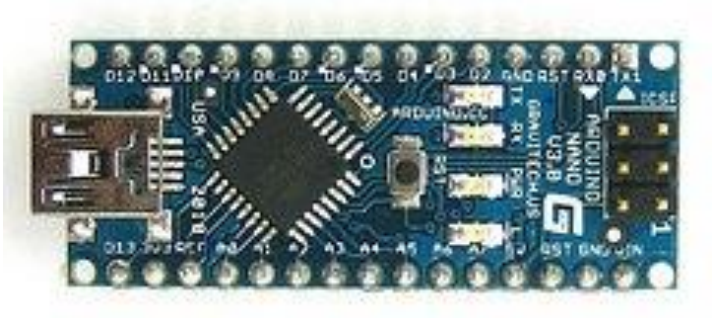
- Arduino UNO – Oferece várias opções e possibilita a utilização de acessórios e componentes.

Figura 6 - Arduino UNO

Fonte: Vieira, 2011(https://sejalivre.org/saiba-tudo-sobre-arduino/).

- Arduino NANO – Não utiliza acessórios facilmente e apresenta um tamanho bem menor que a placa uno.

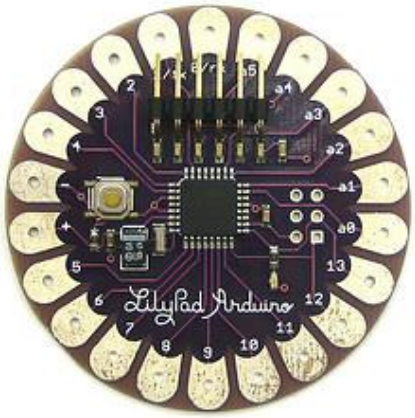
Figura 7 - Arduino NANO



Fonte: Vieira, 2011(https://sejalivre.org/saiba-tudo-sobre-arduino/)

- Arduino *Lilypad* **–** Apresenta um *design* arredondado e pode ser costurada em tecidos.

Figura 8 - Arduino Lilypad



Fonte: Vieira, 2011(https://sejalivre.org/saiba-tudo-sobre-arduino/).

- Arduino mega 2560 – Apresenta uma quantidade maior de memória e de pinos de entrada e saída.

Figura 9 - Arduino mega 2560

Fonte: Vieira, 2011(https://sejalivre.org/saiba-tudo-sobre-arduino/).

As placas variam, apresentando características e pinos diferentes. No geral apresentam pinos de entradas e saídas digitais, pinos de saídas analógicas, saídas PWM (*Pulse Width Modulation* – Modulação por Largura de Pulso). As entradas (recebem tensão) e saídas (fornece tensão) digitais trabalham com dois valores de tensão: nível alto (5V) ou nível baixo (0V), podendo adicionar uma corrente a um determinado componente fazendo com que seja possível mudar o seu estado, como mudar o estado de um relé, ou a função de um botão. As analógicas são as usadas para medir posição, temperatura ou pressão, isto é, medir algo que varie com o tempo. O processador do Arduino é digital, logo há uma conversão embutida no próprio

Arduino para transformas as grandezas digitais em analógicas e vice-versa. As saídas PWM são responsáveis por obter resultados analógicos através de digitais (ARDUINOPORTUGAL.PT, 2017).

A alimentação da placa pode ser feita pela entrada USB conectada a um computador, ou por uma alimentação externa.

3.3.1 IDE

O IDE (*Integrated Development Environment*) é uma plataforma de desenvolvimento. A IDE do Arduino é um ambiente de desenvolvimento integrado. Ou seja, é onde o programador tem o necessário para programar a placa. Nessa plataforma são criados códigos que são enviados diretamente para a placa, para que ela realize a tarefa descrita no código (TORRES,2013). O código para programar o Arduino é feito em uma linguagem de programação própria, que é semelhante a linguagem C/C++. Para obter essa ferramenta é necessário baixá-lo no site oficial do Arduino.

A interface IDE Arduino está apresentada na figura 10:

Figura 10 - Interface IDE Arduino

Fonte: Interface IDE Arduino

Os ícones que estão na parte superior da interface servem para:

 Verificar se o código está correto e se é possível compilá-lo.

 Carregar o código para a placa.

 Abrir uma nova janela.

 Abrir um arquivo.

 Salvar o código criado.

 Abrir monitor serial (monitor presente na plataforma para se comunicar com o código criado através da comunicação serial).

 Apresentar opções para adicionar esboço ao projeto atual.

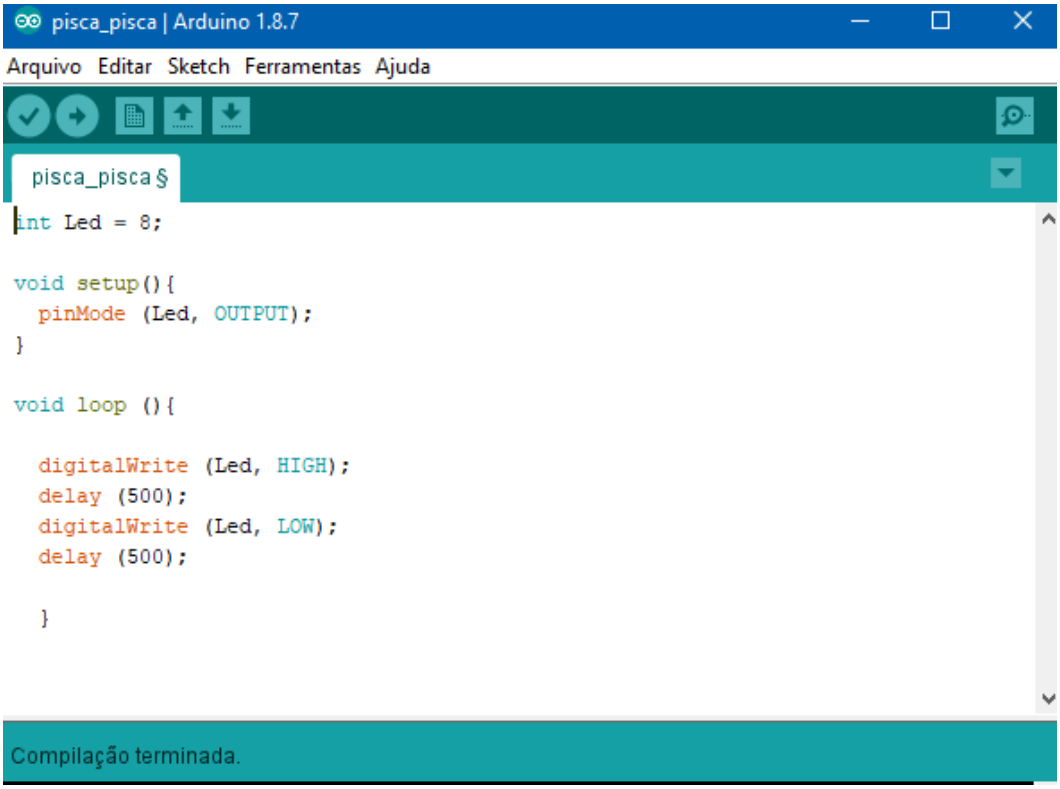
A interface é dividia em duas principais funções: *void setup e void loop*. Na função *void setup* é colocado o que será utilizado na configuração de código, essa função será executada apenas uma vez. Nessa função é possível criar variáveis, determinar os pinos de entradas e saídas, e também determinar os pinos que serão utilizados como entrada serial, ou seja, entradas de comunicação. Isto significa que todas as informações que necessitar configurar para usar no código, deve ser configurado na função *void setup*.

Na função *void loop*, é onde escreve-se o código de processamento de dados. Nessa função o código funciona de forma repetida e contínua, ou seja, funciona como um loop que processa informações sem parar.

O código criado nessas funções devem estar entre as chaves "{ }", pois elas determinam o início e fim de cada função do código. As barras "//" servem para escrever informações sobre o código, como se fossem anotações que servem para melhor compreensão do programador, ou leitor do código. O que é escrito após as barras, não é compilado para a placa do Arduino, serve apenas para explicar o código.

Para melhor compreensão da linguagem de programação do Arduino, a figura 11 apresenta um exemplo de programa criado para piscar um LED a cada 500 millisegundos:

Figura 11 - Exemplo de código criado na Interface IDE Arduino



Fonte: Interface IDE Arduino

Para a criação do código são utilizados os seguintes comandos:

- "int" - Responsável por criar variáveis.
- led = 8 - A variável "led" é igual a 8, isso significa que o pino 8 será de entrada ou saída tensão do Arduino, e será utilizado para ascender ou apagar o LED.

Na função *void setup*

- pinMode – Esse comando indicará se o pino 8 será entrada (*INPUT*) ou saída (*OUTPUT*) de tensão.
- O pino "led" é colocado em modo "OUTPUT", isto é, modo saída, pois a ideia é enviar tensão para o LED.

Na função *void loop*

- digitalWrite - Essa função indicará a tensão que será enviada para o pino, podendo ser nível alto (5V) ou nível baixo (0V).

- Inicialmente é enviado “HIGH” (Nível alto) para o pino “led”, isso faz com que envie 5V para o LED, logo ele ascenderá.
- delay() – Esse comando serve para determina um tempo. Após esse tempo a função posterior é acionada. O tempo é dado em milissegundos.
- digitalWrite(LOW) - Nível baixo (0V) é enviado para o pino “led”, e ele será apagado.
- “delay()” de 500 milissegundos é colocado novamente, e o código da função *void loop* funcionará constantemente, fazendo com que aconteça um loop e o LED ascenda e apague a cada 500 milissegundos.

A comunicação do Arduino com o computador é através de um código criado no *firmware* IDE, essa comunicação é possível devido um microcontrolador presente na placa. Ele é responsável por possibilitar a compilação do programa criado. Para enviar o código para a placa do Arduino utiliza-se um cabo com entrada USB, que envia o código criado na plataforma IDE direto para a placa do Arduino, que armazena os dados recebidos.

3.4 Comunicação serial

É através da comunicação serial que o Arduino consegue se comunicar com um computador ou outro dispositivo. Isso possibilita a interação entre o aplicativo criado na plataforma *App inventor* e a placa do Arduino.

Na placa do Arduino existem dois pinos de comunicação serial que são Rx (envia dados) e Tx (recebe dados) que estão ligados ao microcontrolador presente no Arduino. Esses pinos são ligados a um módulo que possa se comunicar com um *smartphone*, seja via web ou *bluetooth*. Nesses módulos também há a comunicação Rx e Tx. O Rx do módulo conectado ao Tx do Arduino, o Tx do módulo no Rx do mesmo. Dessa forma, há uma comunicação entre o *smartphone* e a placa, fazendo com que seja possível controlar o sistema montado no Arduino via *smartphone*.

O IDE possibilita um terminal serial que auxilia no recebimento e envio de dados, isto é, possibilita a comunicação serial sem a necessidade smartphone ou uma ferramenta externa. Para utilizar o terminal serial da própria plataforma IDE, basta ir na plataforma IDE e clicar em “Tools” e em seguida “Serial Monitor”.

4 MATERIAS E MÉTODOS

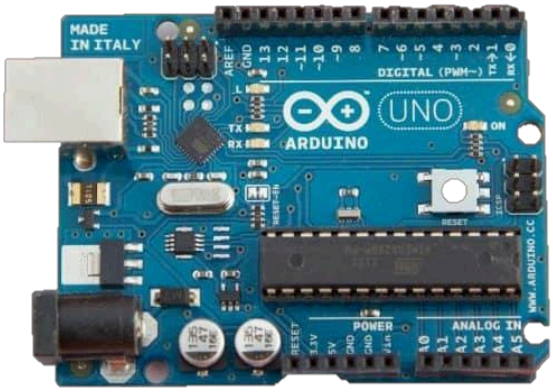
Neste capítulo serão apresentados os materiais utilizados no presente protótipo de automação residencial, buscando entender o conceito e aplicação do mesmo. Além de apresentar a metodologia em que o protótipo foi feito.

4.1 Materiais

Para a implementar o protótipo proposto é necessário entender o conceito e o funcionamento de alguns elementos e componentes que auxiliam em sua aplicação.

4.1.1 Arduino UNO

Figura 12 - Placa Arduino UNO



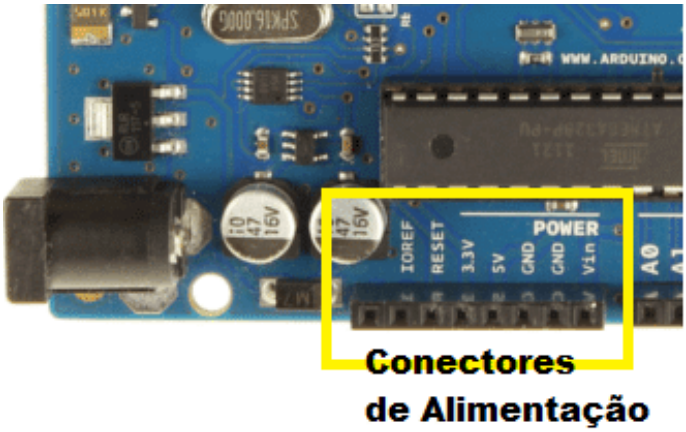
Fonte: Embarcados, 2014(https://www.embarcados.com.br/arduino-uno/).

A placa Arduino UNO possui o microcontrolador ATMEL ATMEGA328, 14 pinos de entras e saídas digitais (sendo 6 dessas saídas PWM), 6 portas analógicas, e 2 portas de comunicação serial.

A alimentação da placa pode ser feita pela entrada USB conectada a um computador, ou por uma alimentação externa. A alimentação externa pode ser por uma fonte no conector *jack*, a mesma pode variar de 6 a 20 volts. Porém, o recomendado é utilizar uma tensão entre 7 a 12 volts, pois uma tensão menor que 7V pode tornar a placa instável, e uma maior que 12V pode sobrecarregar a mesma (SOUZA,2013).

É possível conectar módulos como ledes, relés, sensores e outros, nesta placa. Esses módulos são utilizados para auxiliar nas realizações das tarefas, conforme o código criado no software do Arduino. Os pinos de alimentação para conexão de módulos são os apresentados na figura 13.

Figura 13 - Pinos de alimentação da placa



Fonte: Embarcados, 2014(https://www.embarcados.com.br/arduino-uno/)

Cada pino de alimentação apresenta uma função diferente:

IOREF - Fornece a tensão de referência que o microcontrolador está operando. Quando configurado corretamente, um circuito integrado tem a capacidade de ler a tensão do presente pino, e selecionar a fonte tensão adequada seja 5V ou 3,3V.

RESET - É utilizado para um reset externo da placa Arduino.

3,3 V. – Nesse pino será fornecido ao módulo externo uma tensão de alimentação de 3,3V, onde sua corrente máxima será de 50 mA.

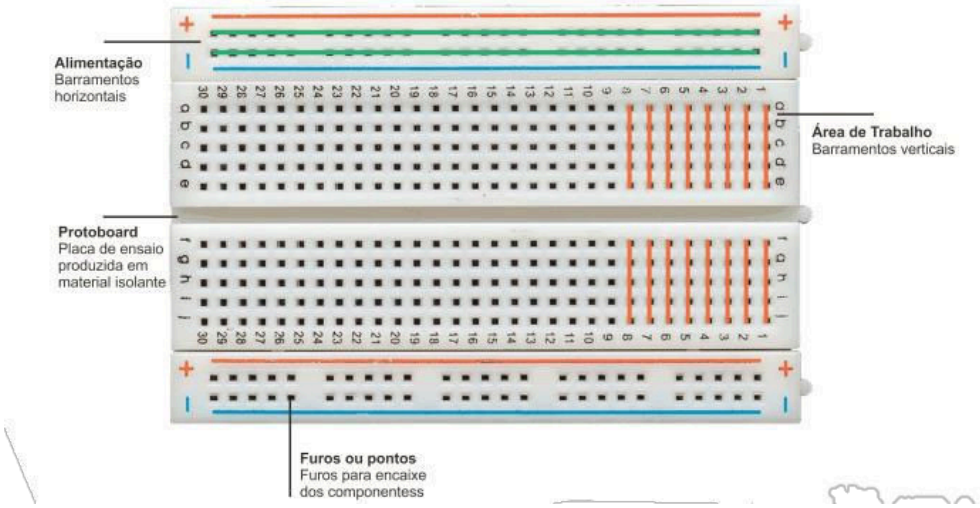
5 V – Será fornecido uma tensão de 5 V para alimentação de circuitos externos.

GND - Pinos de referência, ou aterramento.

VIN – Alimenta a placa através de uma bateria externa. Se a placa for alimentada pelo conector *jack*, a tensão dessa alimentação estará nesse pino.

4.1.2 Protoboard

Figura 14 - Protoboard



Fonte: Mota, 2018(https://portal.vidadesilicio.com.br/protoboard/).

É uma placa que possibilita a construção de circuitos sem a necessidade de soldar os componentes, fazendo com que os mesmos sejam móveis, dessa forma é possível montar diversos circuitos diferentes na mesma placa. Para que a corrente circule no circuito são utilizados *jumpers*, que são fios de conexão que auxiliam nas ligações do circuito, podendo então ligar componentes a *protoboard* e criar o circuito desejado.

Os pinos entre as linhas vermelhas e azul são normalmente usados para alimentação. Apresentam duas linhas horizontais na parte superior e duas na parte inferior. A linha verde abaixo da linha vermelha apresentada na figura 10, apresentam todos os pontos ligados entre si. O mesmo ocorre com a linha acima da linha azul, na parte inferior e superior da placa.

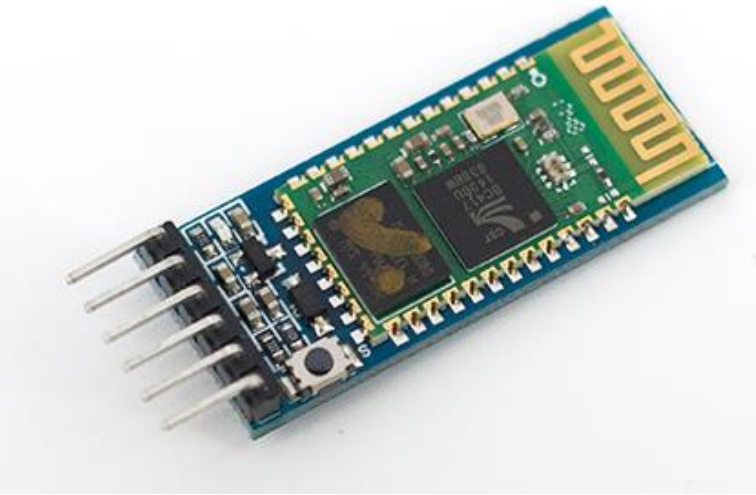
Na área de trabalho é possível inserir componentes e criar o circuito. Os pinos na vertical de cada linha laranja da figura 10 estão ligados entre si, ou seja, apresentam o mesmo ponto. Já as linhas horizontais apresentam pontos diferentes. Com isso é possível montar circuito em série e em paralelo.

Entre a parte superior e inferior há um espaço que divide a placa. Logo, os pontos das linhas verticais da parte superior da área de trabalho, é diferente dos pontos verticais inferiores a esse espaço.

4.1.3 Módulo *bluetooth*

O *bluetooth* é uma forma de comunicação muito utilizada atualmente, em fones de ouvido ou em smartfones para enviar dados. No Arduino, o módulo *bluetooth* é uma das formas possíveis de enviar e receber informações. A sua conexão é simples e facilita a comunicação com o *smartphone*. O módulo utilizado para o protótipo foi o HC-05.

Figura 15 - Módulo *bluetooth*



Fonte: THOMSEN, 2015(https://www.filipeflop.com/blog/tutorial-modulo-bluetooth-com-arduino/).

O módulo possui quatro pinos de conexão: o GND que é ligado no GND do Arduino, isto é, no 0V e o VCC ligado nos 5V da placa do Arduino. Os outros dois pinos Tx E Rx são para enviar e receber dados. O Tx do módulo é ligado no Rx da placa, e o Rx ligado no Tx da mesma. O pino Tx é responsável por transmitir as informações e o Rx por receber as mesmas. Dessa maneira há uma comunicação entre a placa e *bluetooth*, fazendo com as informações e dados passem de um para o outro. É importante destacar que para ligar o Rx do *bluetooth* na placa é necessário fazer um divisor de tensão com o uso de resistores, pois o pino do mesmo funciona em 3,3V, e uma tensão maior do que a que o pino suporta pode danificar o componente.

4.1.4 Módulo relé

O relé é um componente eletrônico que possibilita a interface da eletrônica de baixa potência com a eletrônica de alta potência. O módulo utilizado no protótipo foi o relé 5V para Arduino.

Figura 16 - Módulo relé

Fonte: A autora.

O presente módulo, apresentam três pinos. Dois deles são de alimentação, o GND ligado no 0 V, e o VCC ligado no 5V. O terceiro pino é o IN, responsável por ligar o relé. O outro lado do relé (lado esquerdo com parafusos), apresentam as entradas que estarão ligadas a lâmpadas, ou a um equipamento de alta potência. Para ligar os cabos a ele, basta desparafusar com o auxílio de uma chave de fenda, inserir o cabo na entrada e parafusar novamente para que fique fixo. As entradas são NO (normalmente aberto) estão abertos quando o relé não está energizado, mas quando energizado fecham e passam corrente pela bobina presente no interior do relé, NC (normalmente fechado) funciona de forma contrária ao NC e COM é usado para ligar o dispositivo de alta potência, conectado ao dispositivo desejado.

4.1.5 Interruptor paralelo

Figura 17 - Interruptor paralelo

Fonte: A autora.

Diferente de um interruptor comum, o interruptor paralelo apresenta três terminais. O terminal do meio é o conectado ao fio fase ou retorno fase. Esse interruptor é utilizado para casos em que deseja ligar ou desligar a mesma lâmpada em lugares diferentes, podendo ser com dois interruptores paralelo em lugares diferentes, ou até mesmo de outro dispositivo como o smartphone.

Os terminais localizados nas laterais esquerda e direita do interruptor são ligados a outro interruptor nos mesmos pontos, ou a um relé por exemplo, para que haja uma ligação entre os mesmos, possibilitando o controle de uma lâmpada de forma simultânea. Já o terminal do meio é o de fase ou retorno de fase, que serve para enviar tensão a lâmpada.

4.1.5 Resistores

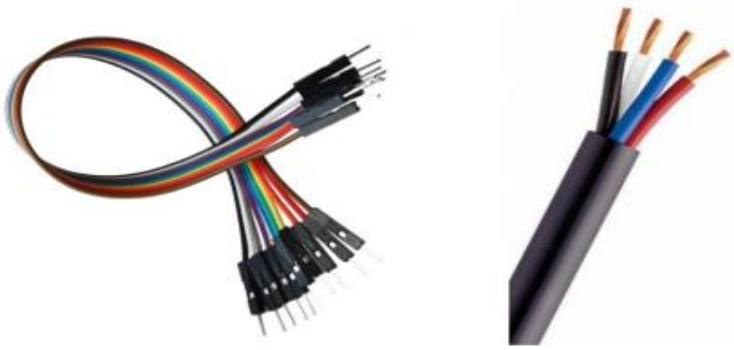
Os resistores são componentes eletrônicos utilizados para limitar a corrente elétrica, possibilitando o controle de tensão ou corrente que chegará a determinado componente, com o intuito de “proteger” o componente.

Figura 18 - Resistores

Fonte: https://brasilescola.uol.com.br/o-que-e/fisica/o-que-sao-resistores.htm.

4.1.6 Cabos e *jumpers*

Figura 19 - Jumpers e Cabos de conexão



Fonte: Google imagens.

A imagem do lado esquerdo da figura 15 apresenta os jumpers, que são peças plásticas com um fio de metal por dentro que servem para conduzir corrente elétrica. Já a imagem do lado direito da figura 15 apresentam os cabos que são vários fios unidos com o mesmo objetivo de conduzir corrente, porém, a sua condutividade é maior por apresentar vários fios juntos, sendo assim os cabos são usados para lâmpada e relé, e os jumpers para os componentes na protoboard, pois os mesmos não necessitam de uma tensão ou corrente alta para funcionar.

4.1.7 Lâmpada

Figura 20 - Lâmpada

Fonte: http://tudo.extra.com.br/ferramentas/tudo-o-que-voce-precisa-saber-sobre-lampadas/

A lâmpada é o dispositivo elétrico que transforma energia elétrica em luminosa. Popularmente conhecida e encontrada nas residências.

4.2 Método

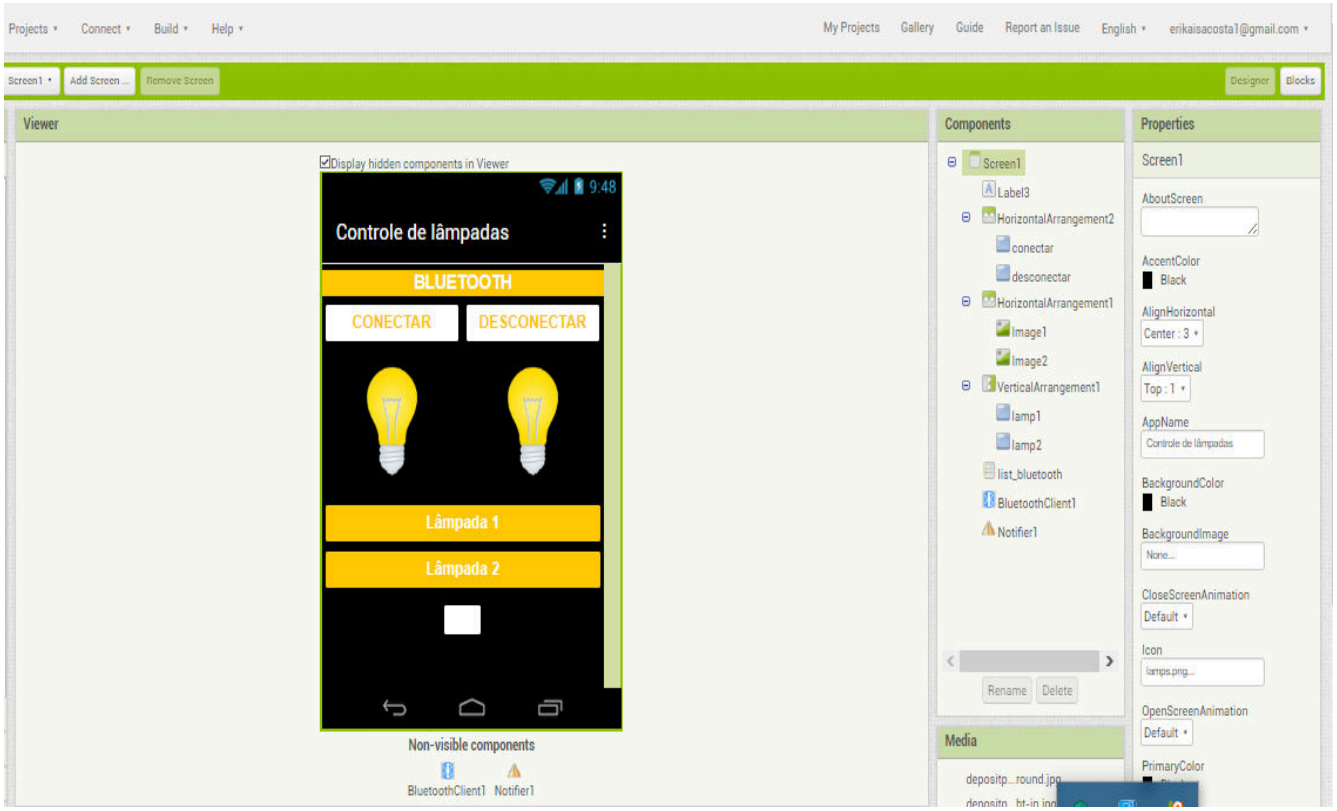
A criação do protótipo é dividida em três partes: o código criado no IDE Arduino, o aplicativo criado no *App inventor* e a montagem do circuito.

4.2.1 Código na plataforma IDE

Inicialmente é criado o código na plataforma IDE. No código são determinados os pinos onde os relés são conectados. Além disso, são determinadas também as funções através dos comandos. Em seguida, o código é enviado para a placa através de um cabo USB, esse código fica armazenado na placa, e a placa realiza as tarefas inseridas no código.

4.2.2 Aplicativo

Figura 21 - Design do aplicativo



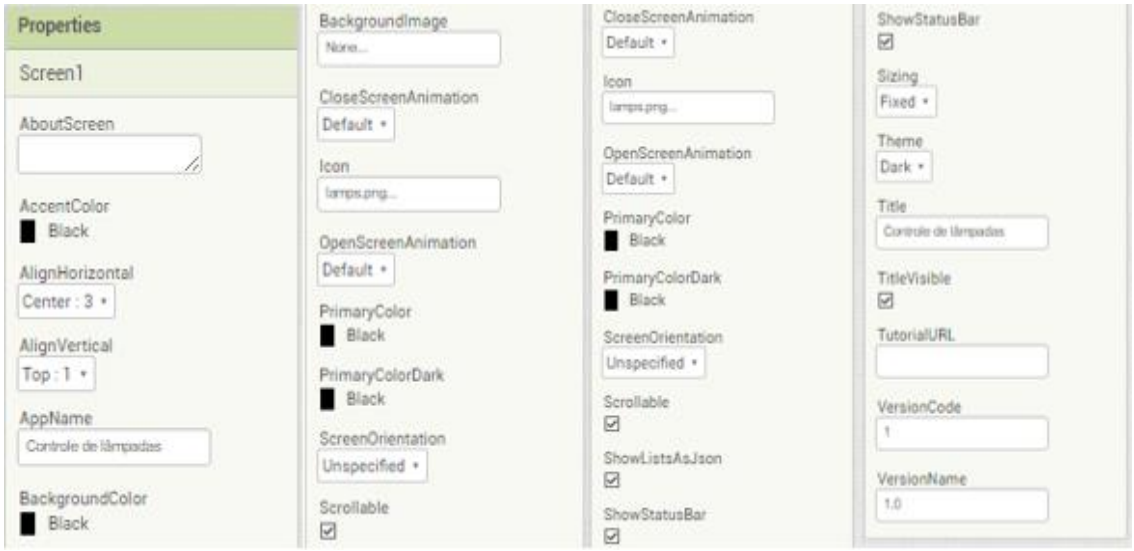
Fonte: a autora.

Na coluna de componentes apresentada na figura 21, é possível visualizar todos os componentes inseridos para a criação do aplicativo que são: Label3, HorizontalArrangment2 com os botões “conectar” e “desconectar” inseridos no mesmo, o HonrizontalArrangement1 com duas imagens de lâmpadas inseridas uma ao lado da outra, VerticalArrangement1 com os botões lamp1 e lamp2, o list_bluetooth, BluetoothClient1 e Notifier1.

O Screen1 é a parte onde será organizado como o aplicativo ficará na tela do *smartphone*. As propriedades inseridas a ele estão apresentadas na figura 20. A parte do fundo do aplicativo está configurada para ficar totalmente preta, por isso todos as possíveis propriedades de cor estão com a cor preta selecionada. Na propriedade AppName é onde é adicionado o nome do aplicativo. O aplicativo recebe o nome de Controle de lâmpadas. A configuração AlighHorizontal determina como ficará o aplicativo do smartphone, o “Center:3” selecionado significa que a tela ficará centralizada no smartphone. A imagem de uma lâmpada é inserida em “Icon” que será o ícone do aplicativo.

As propriedades dos principais componentes inseridos no *designer* estão apresentas na figura 22 abaixo:

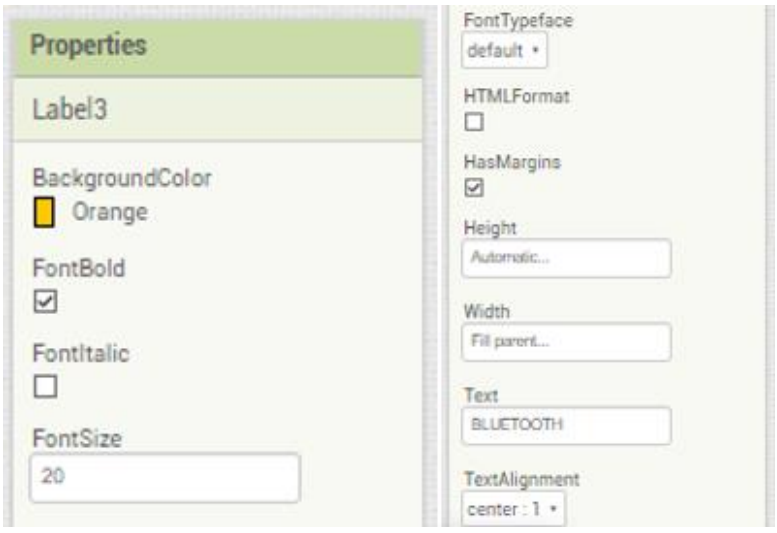
Figura 22 - Propriedades do Screen 1



Fonte: a autora.

O componente Label3 é inserido. A cor desse componente será “Orange” apresentada nas propriedades mostrada na figura 21. Esse componente possibilita inserir texto. Para identificar que os botões “conectar” e “desconectar” são para ligações via *bluetooth*, o nome BLUETOOTH será inserido com uma fonte (FontSize) de 20.

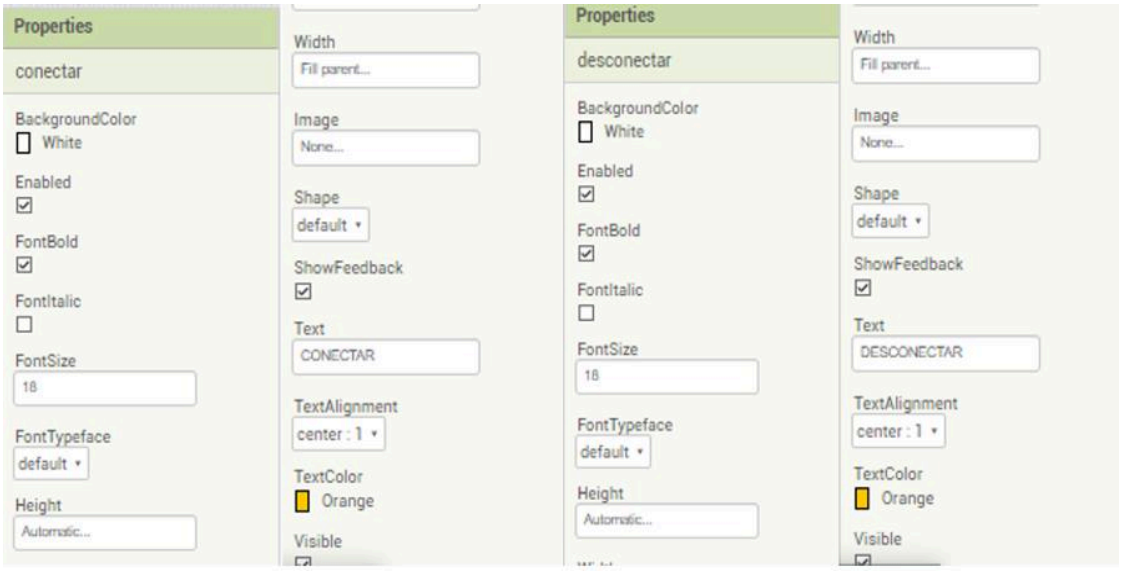
Figura 23 - Propriedades do texto "BLUETOOTH" escrito na parte superior



Fonte: a autora.

As propriedades dos botões “conectar” e “desconectar” são apresentadas na figura 24 abaixo. Ambos botões apresentam a cor branca (White) selecionada em BackgroundColor, estão com textos centralizados e a cor do texto escrito é laranja (Orange) selecionada em TextColor com fonte 18. Os botões estão inseridos dentro do HorizontalArrangment2, que possibilita que os botões fiquem horizontais para melhor organização do aplicativo.

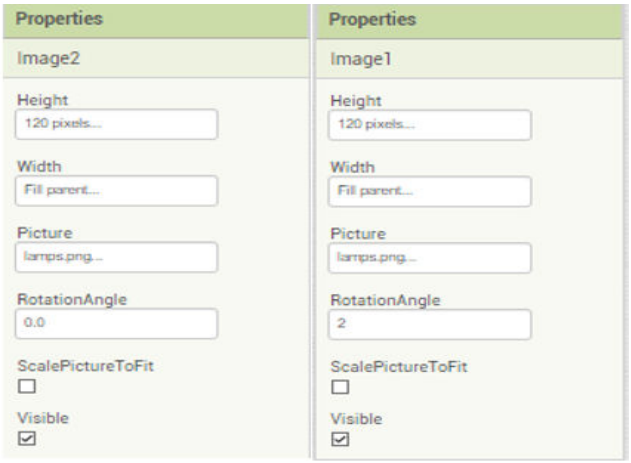
Figura 24 - Propriedades dos botões conectar e desconectar



Fonte: a autora.

As propriedades das imagens das lâmpadas presentes na tela são apresentadas na figura abaixo. As imagens foram selecionadas na configuração Picture. As imagens estão inseridas em HorizontalArrangment1, para manter as imagens na horizontal de forma alinha e visualmente mais interessante.

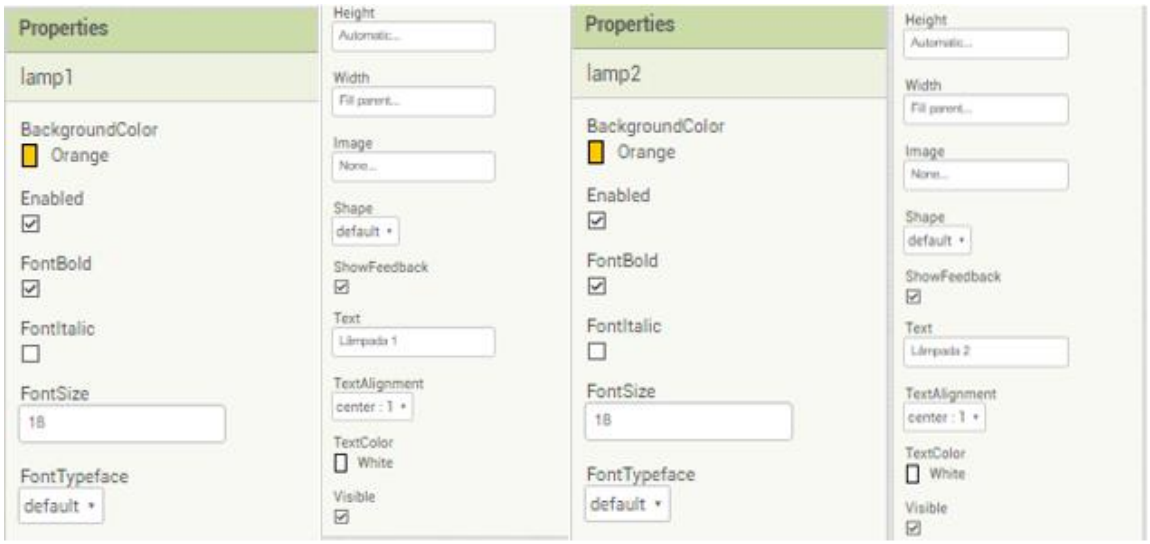
Figura 25 - Propriedade das imagens



Fonte: a autora.

As propriedades para os botões lâmpada 1 e lâmpada 2 são apresentadas na figura 24. Ambos botões apresentam a coloração laranja (*orange*) selecionada em BackgroundColor. O tamanho da letra é 18 e a cor do texto é branco (White). Os botões estão inseridos em VerticalArrangment, que serve para organizá-los verticalmente.

Figura 26 - Propriedades dos botões lâmpada 1 e lâmpada 2



Fonte: a autora.

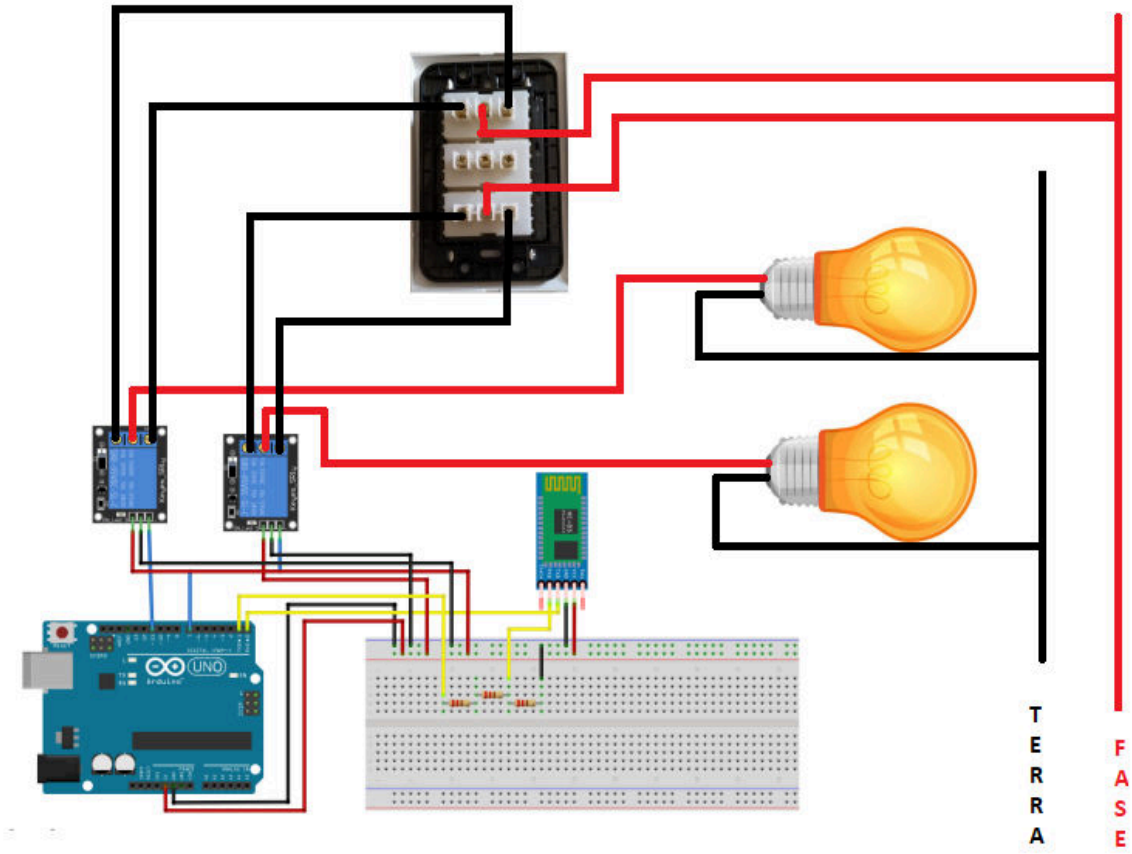
Os componentes o list_bluetooth, BluetoothClient1 e Notifier1 não são visíveis na tela do smartphone, mas servem para ser utilizado na programação do aplicativo. O list_bluetooth serve para apesentar a lista de dispositivos *bluetooth* disponíveis. O BluetoothClient1 serve para utilizar a conexão via *bluetooth* e Notifier1para apresentar notificações no aplicativo.

Após a criação do *designer* são inseridos comandos aos componentes. Esses comandos são feitos através da junção dos blocos. Os blocos são encaixados e são determinadas as funções para que o aplicativo funcione e se comunique com os componentes ligados a placa do Arduino.

4.2.3Circuito

As ligações do circuito estão apresentadas no diagrama criado na ferramenta Fritzing, mostrado na figura 27:

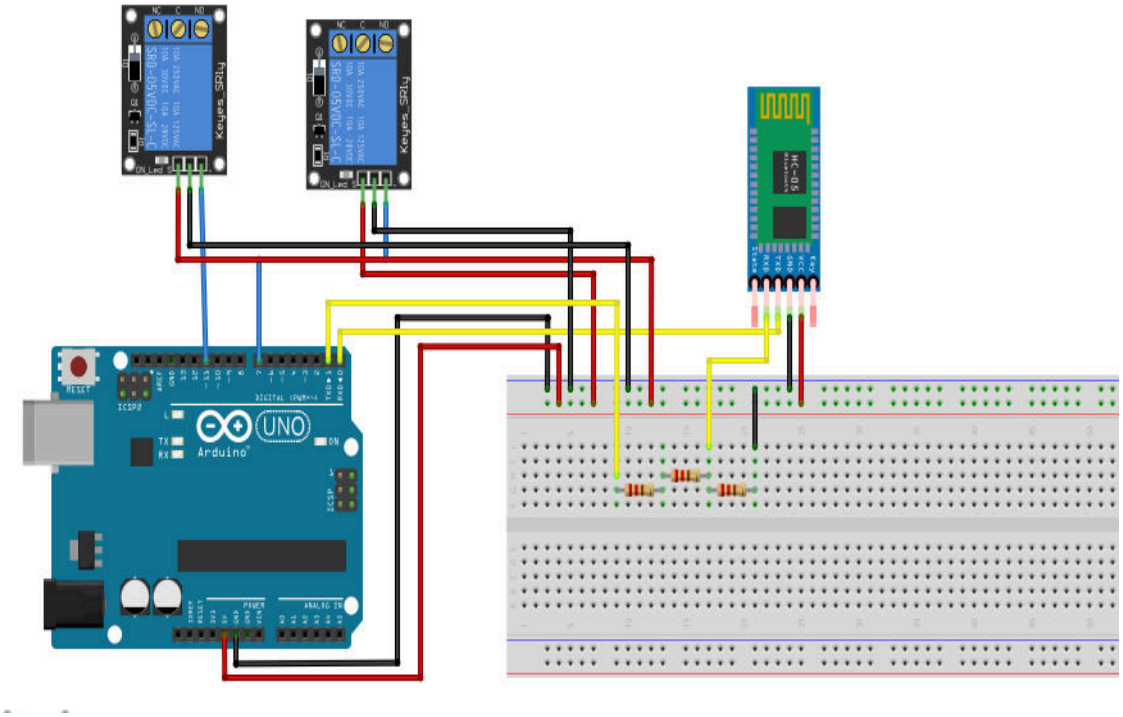
Figura 27 - Circuito do protótipo



Fonte: a autora.

O circuito apresenta as ligações dos componentes no Arduino, e as ligações do relé com as lâmpadas e interruptor. Para melhor visualização e compreensão, será explicado por partes a montagem do circuito.

Figura 28 - Primeira parte do circuito



Fonte: a autora.

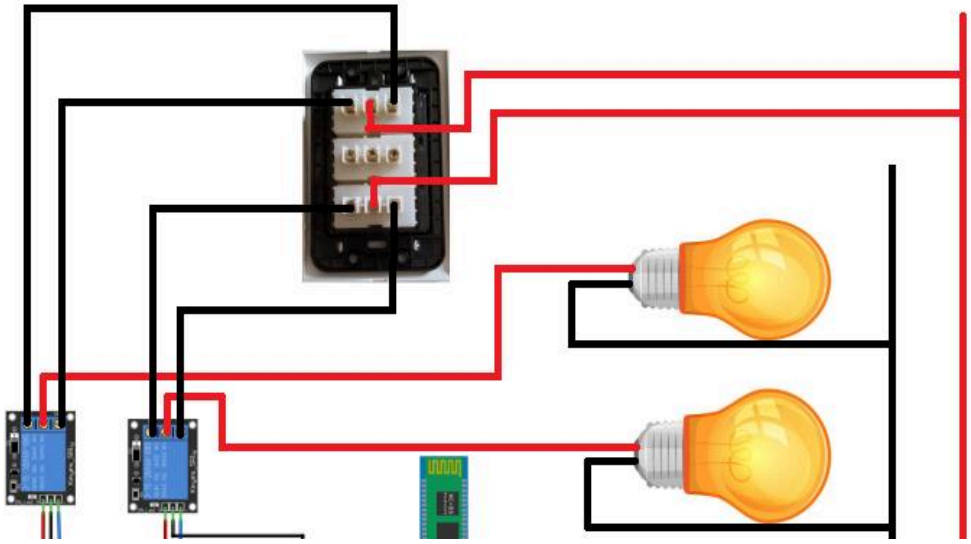
Inicialmente os pinos do Arduino UNO 5V (jumper vermelho) e GND (jumper preto) são ligados a protoboard, determinando que qualquer jumper ligado a linha superior onde o jumper vermelho está terá 5V, e ligado a linha onde o jumper preto está ligado terá 0V. Logo o VCC e GND dos dois módulos relés e do módulo *bluetooth* são ligados a esses pontos, energizando então os componentes para que o sistema funcione.

Os pinos Tx e Rx são os de comunicação, neles foram usados os jumpers amarelos para demonstração. O Tx do módulo é ligado no Rx da placa do Arduino, e o Rx no Tx da placa. Porém, no Rx há um divisor de tensão feito com três resistores de resistências iguais, pois a tensão de alimentação do Rx do módulo é de 3,3V, e já que a placa fornece 5v, o divisor serve para diminuir a tensão e enviar uma tensão que o Rx do componente suporta, evitando que o componente queime.

O módulo relé é ligado no VCC e GND na protoboard, e o pino “IN” do primeiro relé é colocado no pino 7 da placa do Arduino, e o segundo relé no pino 11, pois foram os pinos escolhidos para saída de energia no código criado para o protótipo.

A segunda parte a ser montada no circuito é a ligação ente o relé, a lâmpadas e o interruptor paralelo, que é apresentada na figura 29 seguir:

Figura 29 - Segunda parte do circuito



Fonte: a autora.

Os cabos das entradas NO e NC do primeiro relé são ligados nas extremidades do interruptor, para ser possível controlar tanto pelo smartphone quando pelo interruptor. Essa ligação possibilita essa interação. O pino COM do relé é ligado no fio de fase da lâmpada, para que seja possível ligar a lâmpada pelo smartphone via *bluetooth*. O pino do meio do interruptor é ligado a fase vindo da tomada (representado pela linha vermelha na lateral direita da figura 29), para que seja energizado e para que esteja ligado a lâmpada. Por fim, o fio terra (representado pela linha preta na lateral da figura 29) vindo da tomada é ligado no terminal negativo da lâmpada, para fechar circuito. O mesmo procedimento se repete para o segundo relé e para a segunda lâmpada.

5 RESULTADOS E DISCUSSÃO

5.1 Códigos

5.1.1 Arduino

O código criado na plataforma:

```
#include <SoftwareSerial.h> // Uma nova porta serial é criada
int rele1 = 7;
int rele2 = 11;
SoftwareSerial comSerial(0,1); // pinos RX/TX para o modulo

void setup()
{
Serial.begin(9600); //porta serial para monitorar os caracteres enviados
Serial.println("comunicação"); // mensagem de inicializacao

comSerial.begin(9600); // inicializacao da porta serial para o modulo

pinMode(rele1, OUTPUT); //  Pino 7 é definido como saída de energia para o Rele 1
pinMode(rele2, OUTPUT); // Pino 11 é definido como saída de energia para o Rele 2
}
void loop()
{
if (comSerial.available()){ // Se a comunicação serial estiver disponivel, há uma caracter para ler
char caracter = comSerial.read(); // A caracter é salva na variável
Serial.write(caracter); // escreve o caracter recebido via bluetooth na serial do PC para conferencia
if (caracter=='A'){ digitalWrite(rele1, HIGH); } // Liga Rele 1
if (caracter=='a'){ digitalWrite(rele1, LOW); } // Desliga Rele 1
if (caracter=='B'){ digitalWrite(rele2, HIGH); } // Liga Rele 2
if (caracter=='b'){ digitalWrite(rele2, LOW); } // Desliga Rele 2

}
}
```

- *#include <SoftwareSerial.h>*

Inicialmente o #include <SoftwareSerial.h> é digitado, indicando que será criado uma nova porta serial.

- *int rele1 = 7; int rele2 = 11;*

São criadas duas variáveis “rele1” e “rele2”. Essas variáveis são usadas para ligar as lâmpadas, são apresentadas pelo comando “int” para apresentar um valor a variável. “rele1” = 7 e “rele2” =11, significa que o primeiro relé estará ligado no pino 7, e o segundo no pino 11 da placa do Arduino.

- *SoftwareSerial comSerial(0,1);*

A porta serial criada inicialmente recebe o nome de “comSerial” que está apresentada na quarta linha do código, representada pelos pinos de comunicação serial 0 e 1 da placa do Arduino.

Na função *void setup:*

- *Serial.begin(9600);*

Indica a inicialização da comunicação serial com 9600 bits por segundo. A partir desse comando será monitorado os caracteres enviados para a porta serial.

- *Serial.println("comunicação");*

É digitado para obter uma mensagem no monitor serial do IDE. A mensagem escrita dentro do parêntese “(Comunicação)” é apresentada no monitor serial. Essa mensagem serve apenas para casos em que o controle de lâmpada seja feito por meio do monitor serial presente na própria plataforma IDE.

- *comSerial.begin(9600);*

Ao inserir esse comando, inicia-se a comunicação serial com a o módulo *bluetooth*.

- *pinMode(rele1, OUTPUT);pinMode(rele2, OUTPUT);*

O comando “pinMode” é utilizado para definir que as variáveis “rele1” e “rele2” serão determinadas como pinos de saída, por isso o comando “OUTPUT”. Essa declaração indica que os pinos estarão enviando corrente, para energizar o relé e fazer com que as lâmpadas acendam.

Na função *void loop:*

- *if (comSerial.available()*

Na parte “*void loop*”, é inserido o comando “if (se) comSerial.available()”. Isso significa que será verificado se há algum dado na porta serial.

- *char caracter = comSerial.read(); // A caracter é salva na variável*

Em seguida, é criada a variável com o comando “char” (comando usado para determinar que uma variável é igual a apenas um caractere), que recebe o nome de *caracter*. Essa variável será igual a “comSerial.read()”, ou seja, será igual a caractere enviada pela comunicação serial.

- *Serial.write(caracter);*

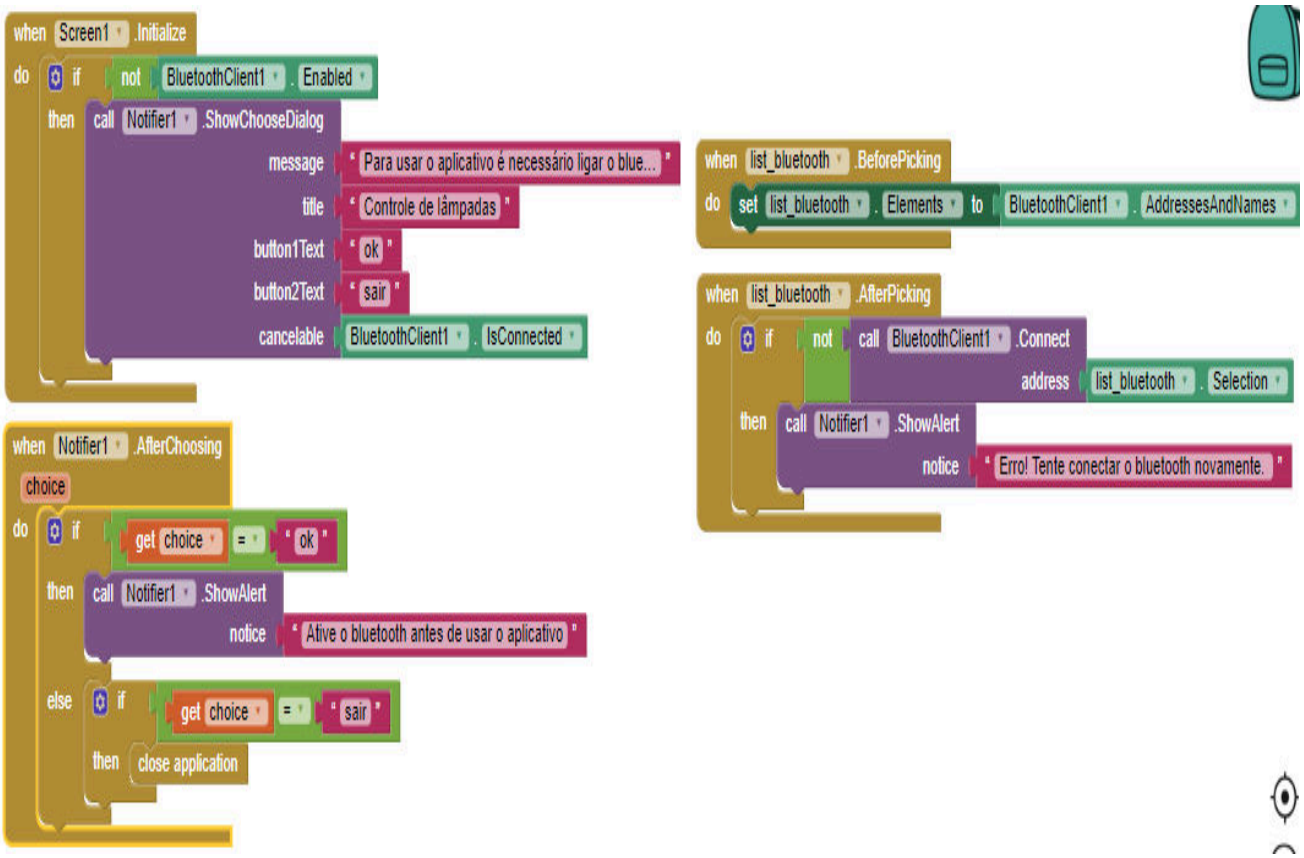
Esse comando escreve o caracter recebido via *bluetooth* na serial do PC para conferencia.

- *if (caracter=='A'){ digitalWrite(rele1, HIGH);*
- *if (caracter=='a'){ digitalWrite(rele1, LOW);*
- *if (caracter=='B'){ digitalWrite(rele2, HIGH);*
- *if (caracter=='b'){ digitalWrite(rele2, LOW);*

As funções *if (caracter) e digitalWrite* são responsáveis por ligar ou desligar a lâmpada. O “digitalWrite” que ativa o LOW (nível baixo) ou HIGH (nível alto), isso significa que pode enviar 5V (HIGH) ou 0V (LOW) para o relé, e o relé energizará a lâmpada que requer uma potência maior. No primeiro caso *if (caracter == ‘A’) digitalWrite (rele1, HIGH).* Isso significa que quando a caractere enviada pela comunicação serial for ‘A’, a lâmpada conectada ao relé 1 ascenderá, caso a variável *caracter* seja igual a ‘a’ a lâmpada aparará. O mesmo se repete para o segundo relé, quando a variável *caracter* envia pela comunicação serial os valores “B” para ligar, e “b” para desligar a lâmpada”.

5.1.1 App inventor

Figura 30 - Programação do aplicativo (primeira parte)



Fonte: a autora.

Na primeira parte da programação do aplicativo (apresentada na figura 30), é onde inicialmente é determinado os comandos iniciais para utilização do aplicativo.

- Bloco 1 (When screen1.initialize do)

O primeiro bloco é o “When do” onde é colocado “screen1.initialize”. O bloco é usado para determinar que quando o Screen1 inicializar, ou seja, quando abrir o aplicativo, “do” (fazer) realizar os comandos inseridos dentro do bloco, que é o “if then”. No “if” é colocado o bloco de lógica “not” e dentro desse bloco o “bluetoothClient1 Enabled”, isso significa que se o *bluetooth* do smartphone estiver desligado, “then” (então) aparecerá a notificação do bloco “call notfler.ShowChooseDialog”. A mensagem que a notificação apresentará é a presente no bloco de texto em frente a “messege” com o texto “Para usar o aplicativo é necessário ligar *o bluetooth* do seu smartphone”, o “title” (título) da notificação é “Controle de lâmpadas”, e dois botões estarão presentes na notificação com “ok” e “sair”. Caso o *bluetooth* esteja ligado no smartphone, a notificação não aparecerá no smartphone.

- Bloco 2 (When notifier1.AftsChoosing do)

No segundo bloco de comando é usado novamente “When do” dessa vez usado para a notificação. No “if” (se) é colocado uma lógica (bloco verde) que determina que se o botão da notificação “ok” for clicado, “then” (então) aparece a notificação na tela do aplicativo com o texto “Ative o bluetooth antes de usa o aplicativo”. Por fim, o bloco apresenta o comando “else” que é usado para quando o usuário clicar em “sair”. Ao clicar em “sair’ o aplicativo será fechado (bloco “closed application”).

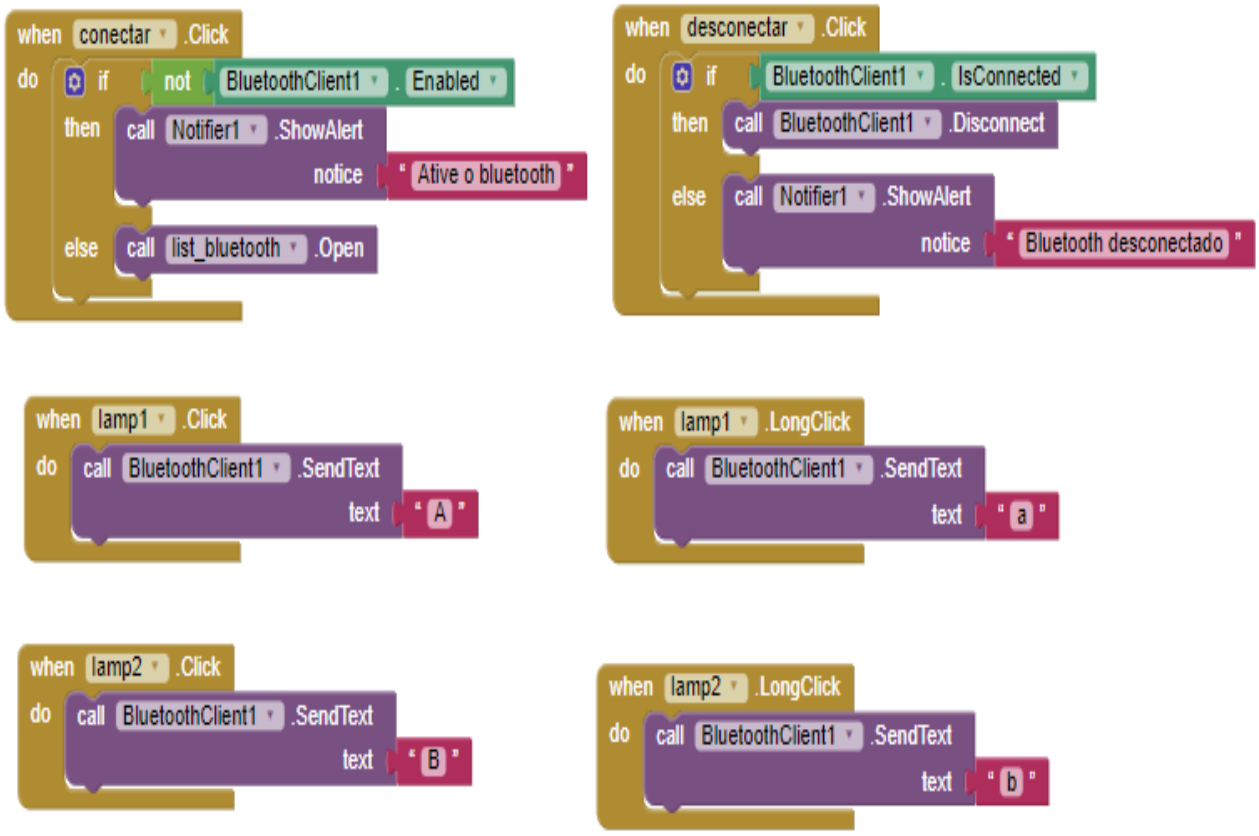
- Bloco 3 (When list_bluetooth.BeforePickng do)

O bloco “when do” é inserido novamente para “list_bluetooth.BeforePickng” com o bloco “set list_bluetooth elements to” que é usado para aparecer na tela do usuário as listas de dispositivos *bluetooth* disponíveis para conexão.

- Bloco 4 (When list_bluetooth.AfterPicking do)

O comando “when do” é inserido para “list_bluetooth.AfterPicking”(depois de escolher algum dispositivo *bluetooth* da lista) novamente é usado com “if” (se) com blocos de lógica “not” (negação), ou seja, caso o *bluetooth* não seja conectado ao dispositivo escolhido na lista de *bluetooth* disponíveis, “then” na tela do usuário aparece o texto “ERRO! Tende conectar o bluetooth novamente”.

Figura 31 - Programação do aplicativo (segunda parte)



Fonte: a autora.

- Bloco 5 (Botão conectar)

O botão “conectar” está no bloco “when do”, ou seja, “when” (quando) o botão é clicado, “do” (fazer): “if” (se) o bluetooth não estiver conectado, “then” (então) aparece a notificação com o texto “Ative o bluetooth”. Caso o bluetooth do smartphone esteja ligado, o comando “else” com os blocos “list_bluetooth.Open” faz com que apresente a lista de todos os bluetooh disponíveis para conexão ao clicar no botão conectar.

- Bloco 6 (Botão desconectar)

O botão “desconectar” está no bloco “when” (quando), que significa que quando o botão é clicado, “do” (fazer): “if” (se) o *bluetooth* conectado, “then” (então) desconectar o *bluetooth*, e “else” apresenta a notificação com um bloco com o texto “Bluetooth desconectado”, que aparece na tela para o usuário quando o botão for clicado.

- Blocos para os botões lamp1e lamp2 :

Ao clicar o botão “lamp1”, será enviado pelo *bluetooth* o texto “A”, caso seja inserido um “long.click”, ou seja, um click longo, o texto enviado pelo *bluetooth* é “a”. Para o segundo botão “lamp2”, repete o mesmo processo, a única mudança é que em um click normal no botão o *bluetooth* enviará “B” e caso seja um longo click enviará “b”. A letras “A”, “a”, “B” e “b” são as enviadas pois o código criado no IDE Arduino utiliza essas letras para ligar ou deligar as lâmpadas, logo é possível notar que haverá uma comunicação entre o smartphone e os componentes ligados ao Arduino.

5.2 Aplicativo

O controle das lâmpadas por meio do smartphone deve ser feito pelo aplicativo criado, que apresentou os seguintes resultados:

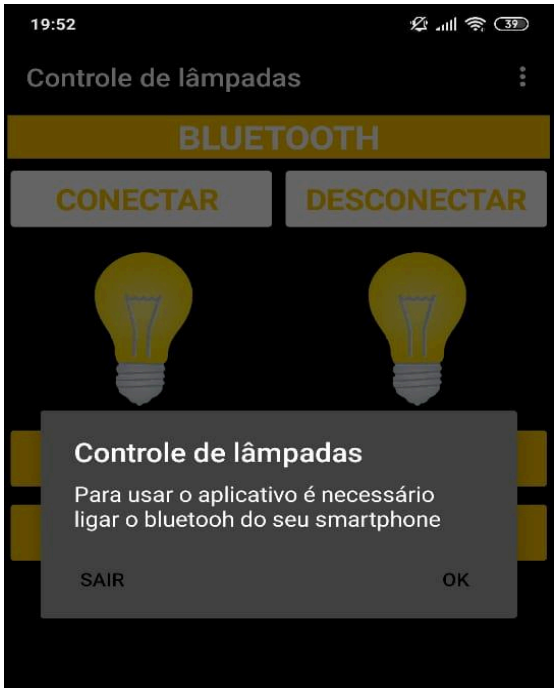
O aplicativo no smartphone tem como título “Controle de lâmpadas” e o ícone do aplicativo é um desenho uma lâmpada selecionada durante a criação do mesmo:

Figura 32 - Ícone e nome do aplicativo

Fonte: a autora.

Ao abrir o aplicativo aparece na tela do usuário a notificação apresentada na figura 33 seguir:

Figura 33 - Tela inicial do aplicativo



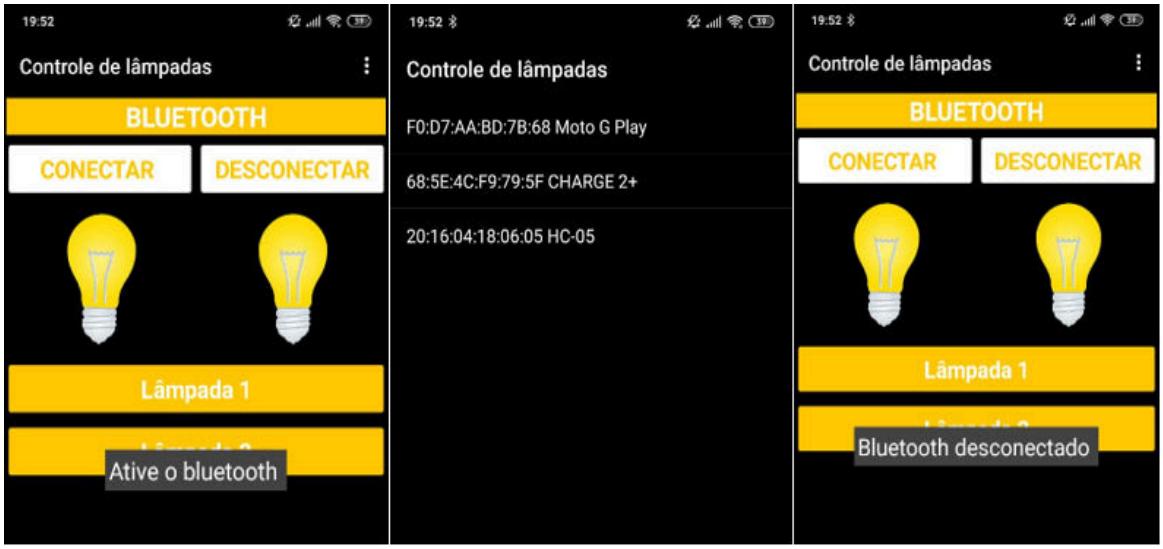
Fonte: a autora.

Caso o usuário clique “sair”, o aplicativo será fechado. Caso click “ok” aparece na tela do usuário a mensagem apresentada na figura 34.

Figura 34 - Notificação do aplicativo

Fonte: a autora.

Figura 35 - Funções dos botões conectar e desconectar



Fonte: a autora.

O lado esquerdo da figura 35 é apresentado ao usuário quando o botão “CONECTAR” é clicado e o *bluetooth* do *smartphone* não está ligado, a notificação “Ative o bluetooth” aparece na tela do *smartphone*. Quando o usuário liga o *bluetooth* do *smartphone*, a imagem do meio da figura 33 aparece na tela apresenta a lista dos possíveis dispositivos *bluetooth* para conexão. O módulo usado é o HC-05, que está apresentado na lista de *bluetooth*. Para conectar basta clicar no mesmo. O *bluetooth* será pareado, e logo é possível obter uma conexão via *bluetooth.* O botão “DESCONECTAR” é usado para desconectar o usuário do dispositivo *bluetooth*, caso seja clicado este botão a notificação que aparece para o usuário é apresentada no lado direito na figura 35.

Para ligar a lâmpada, basta clicar no botão “lâmpada 1” que aciona o primeiro relé e liga a primeira lâmpada. Para desligar basta dar um click longo, ou seja, pressionar o botão por um pequeno tempo e a lâmpada será desligada. O mesmo se procedimento é feito para o relé 2 e a lâmpada 2.

5.3 Protótipo

O protótipo apresentou os seguintes resultados:

Figura 36 - Protótipo proposto

Fonte: a autora.

As ligações foram feitas seguindo a metodologia apresentada anteriormente, com as ligações do Arduino para o relé, e do relé para as lâmpadas e interruptores. Cada lâmpada foi colocada em um bocal, para segurança em relação aos fios de conexão, e para melhor visualização dos resultados com as lâmpadas.

Na figura 36 é possível perceber que os botões do interruptor não foram clicados. Ao clicar no botão lateral do lado direito, a lâmpada da direita ascende, o mesmo se repete para a esquerda quando o botão esquerdo do interruptor é clicado. Para desligar a lâmpada basta apertar nos botões do interruptor novamente. Esse resultado está apresentado na figura 37:

Figura 37 - Lâmpadas controladas via interruptor paralelo

Fonte: a autora.

Para controlar a lâmpada via smartphone, inicialmente o smartphone deve ser conectado ao módulo *bluetooh* para que haja a comunicação entre o aplicativo e o circuito. Para verificar se o *bluetooth* está conectado, basta verificar o módulo *bluetooth*. Quando o módulo não está conectado a luz vermelha presente no mesmo pisca com uma frequência mais alta, e quando conectado pisca com uma frequência menor. Após a conexão com *bluetooth,* o protótipo pode ser controlado via *smartphone*. Para isso, é clicado o botão "lâmpada 1" na tela do aplicativo e a primeira lâmpada ascende. O mesmo se repete para a segunda lâmpada, ao clicar o botão "lâmpada 2" do aplicativo. Com isso é possível controlar as lâmpadas via smartphone e também pelo interruptor paralelo.

O protótipo apresenta diversas vantagens, uma das suas principais vantagens é que para a implementação do mesmo, não é necessário trocar todo o sistema elétrico da residência. Por exemplo, a maioria dos projetos de automação residencial disponíveis no mercado possuem seus próprios interruptores, com isso, é necessário trocar todos os interruptores da residência junto com algumas implementações do sistema. O protótipo proposto não necessita fazer a troca dos interruptores, desde que os interruptores da residência sejam paralelos. Caso não sejam paralelos, a implementação do protótipo ainda apresenta um custo menor pois os interruptores paralelos convencionais custam menos do que os disponíveis nos sistemas de automação.

Logo, é possível concluir que o presente protótipo apresenta um custo bem menor por utilizar ferramentas como Arduino que é um microcontrolador de baixo custo, alguns módulos como relé, *bluetooth*, resistores, entre outras ferramentas que apresentam bons resultados e bom preço.

6 CONSIDERAÇÕES FINAIS

O presente trabalho apresentou a elaboração de um protótipo de automação residencial, buscando controlar lâmpadas por um interruptor e via smartphone simultaneamente. O auxílio das ferramentas Arduino e App inventor foram de grande importância para a criação do protótipo, por apresentar baixo custo e por serem mais simples de manusear se comparado a outras plataformas que requer um pouco mais de conhecimento em programação. Apesar de serem mais simples e de baixo custo, os resultados na utilização dessas ferramentas se mostraram bastante satisfatórios. Baseado nos presentes resultados é notório que o protótipo cumpre com o que foi proposto, mostrando que é possível criar um sistema de baixo custo com um bom funcionamento.

Desse modo, é possível concluir que o protótipo pode ser algo a ser aplicado, e com as inúmeras possibilidades que o Arduino proporciona é possível acrescentar mais funções ao sistema de automação residencial, podendo controlar mais lâmpadas ou até mesmo outros aparelhos elétricos de uma residência. O protótipo também pode ser algo a ser implementado no mercado, por proporcionar conforto e economia aos que o utilizam.

7. REFERÊNCIAS BIBLIOGRÁFICAS

ANGELONI, Guilherme Campos. **AUTOMAÇÃO RESIDENCIAL.** 91 f. TCC (Graduação) - Curso de Engenharia de Controle e Automação, Universidade Federal de Santa Catarina, Santa Catarina, 2013.

Arduino. Disponível em: < https://www.arduino.cc>. Acesso em: 25 jul. 2019.

ARDUINOPORTUGAL.PT. **Qual a diferença entre entradas Digitais, Analógicas e PWM.** Disponível em: <https://www.arduinoportugal.pt/grandezas-digitais-e-analogicas-e-pwm/>. Acesso em: 06 jul. 2019.

BEGHINI, Lucas Bragazza. **Automação Residencial de baixo custo por meio de dispositivos móveis com sistema operacional Android.** 2013. 78 f. TCC (Graduação) - Curso de Engenharia Elétrica, Universidade de São Paulo - Escola de Engenharia de São Carlos, São Carlos, 2013.

CASA INTELIGENTE – SISTEMA DE AUTOMAÇÃO RESIDENCIAL. 74 f. TCC (Graduação) - Curso de Tecnologia em Análise e Desenvolvimento de Sistemas, Fundação Educacional do Município de Assis – Fema, Assis, 2014.

CASA, Smart. **Domótica, o que é?** Disponível em: <http://smartcasa.pt/domotica-o-que-e/>. Acesso em: 16 jul. 2019.

FONEPLAN. **O QUE É AUTOMAÇÃO RESIDENCIAL?** Disponível em: <http://foneplan.com.br/blog/automacao-residencial/o-que-e-automacao-residencial/>. Acesso em: 06 jun. 2019.

FUNÇÃO Setup e Loop ? | Curso de Arduino. 2017. P&B.

GUISS, Alexandre. **Google App Inventor: o criador de apps para Android para quem não sabe programar.** Disponível em: <https://www.tecmundo.com.br/google/11458-google-app-inventor-o-criador-de-apps-para-android-para-quem-nao-sabe-programar.htm>. Acesso em: 04 jun. 2019.

Hardware, o que é. Disponível em: <http://pt.shvoong.com/internet-and-technologies/372953-hardware-que-é/> Acesso em 10 de jun. de 2019.

INTEL. **Arduino * IDE tutorial.** Disponível em: <https://www.intel.com.br/content/www/br/pt/support/articles/000006321/boards-and-kits/intel-galileo-boards.html>. Acesso em: 10 jun. 2019.

JÚNIOR, Joab Silas Da Silva. "O que são resistores?"; Brasil Escola. Disponível em: https://brasilescola.uol.com.br/o-que-e/fisica/o-que-sao-resistores.htm. Acesso em 15 de julho de 2019.

MARCHESAN, Marcelo. **SISTEMA DE MONITORAMENTO RESIDENCIAL UTILIZANDO A PLATAFORMA ARDUINO.** 62 f. TCC (Graduação) - Curso de Tecnologia em Redes de Computadores, Universidade Federal de Santa Maria, Santa Maria, 2012.

MELLO, Alex Vizeu Lopes de. **AUTOMAÇÃO RESIDENCIAL UTILIZANDO O ARDUÍNO.** 2016. 70 f. TCC (Graduação) - Curso de Tecnologia em Sistemas de Computação, Universidade Federal Fluminense, Niterói, 2016.

MIT (Massachusetts Institute of Technology). Disponível em:< http://appinventor.mit.edu/explore/>. Acesso em: 25 jul. 2019.

MOTA, Allan. **COMO USAR UMA PROTOBOARD?** Disponível em: <https://portal.vidadesilicio.com.br/protoboard/>. Acesso em: 22 jul. 2019.

MOTA, Allan. **O que é Arduino e como funciona?** Disponível em: <https://portal.vidadesilicio.com.br/o-que-e-arduino-e-como-funciona/>. Acesso em: 20 mai. 2019.

MURRELEKTRONIK, RedaÇÃo. **Qual é a Historia da Automação Industrial no Brasil?** Disponível em: <http://blog.murrelektronik.com.br/qual-e-a-historia-da-automacao-industrial-no-brasil/>. Acesso em: 09 jul. 2019.

ORLETTE, Felipe Seguichi. **SISTEMA DE CONTROLE DE ACESSO VIA CELULAR USANDO ARDUINO E MÓDULO GSM.** 2016. 60 f. TCC (Graduação) - Curso de Engenharia de Controle e Automação, Universidade Federal de Ouro Preto, Ouro Preto, 2016.

PINTO, Mateus. **O QUE É AUTOMAÇÃO? ENTENDA COMO ELA PODE TRANSFORMAR SUA EMPRESA.** Disponível em: <https://guiaempreendedor.com/automacao-o-que-e/>. Acesso em: 09 jun. 2019.

PULCINELLI, Márcio. **Arduino pra que te quero.** Disponível em: <https://www.tiespecialistas.com.br/arduino-pra-que-te-quero/>. Acesso em: 08 jun. 2019.

QUINDERÉ, Patrick Romero Frota. **Casa Inteligente – Um Protótipo de Sistema de Automação Residencial de Baixo Custo.** 69 f. TCC (Graduação) - Curso de Ciência da Computação, Faculdade Farias Brito, Fortaleza, 2009.

SOARES JÚNIOR, Wellington. **O que é um interruptor paralelo?** Disponível em: <https://www.bluelux.com.br/o-que-e-um-interruptor-paralelo/>. Acesso em: 06 jul.2019.

SOUZA, Fábio. **Arduino - Comunicação Serial.** Disponível em: <https://www.embarcados.com.br/arduino-comunicacao-serial/>. Acesso em: 07 jun. 2019.

SOUZA, Fábio. **Arduino UNO.** Disponível em: <https://www.embarcados.com.br/arduino-uno/>. Acesso em: 28 jul. 19.

THOMSEN, Adilson. **O que é Arduino?** Disponível em: <https://www.filipeflop.com/blog/o-que-e-arduino/>. Acesso em: 20 mai. 2019.

THOMSEN, Adilson. **Tutorial Módulo Bluetooth com Arduino.** Disponível em: <https://www.filipeflop.com/blog/tutorial-modulo-bluetooth-com-arduino/>. Acesso em: 06 jul. 2019.

TORRES, Victor. **Socorro: o que é a IDE do Arduino?!** Disponível em: <http://www.natalnet.br/ura/?p=438>. Acesso em: 28 mai. 2019

VIEIRA, Vinícius. **Saiba tudo sobre Arduino!** Disponível em: <https://sejalivre.org/saiba-tudo-sobre-arduino/>. Acesso em: 16 jun. 2019.

WIKIPÉDIA. Lâmpada incandescente. Disponível em: <https://pt.wikipedia.org/wiki/Lâmpada_incandescente>. Acesso em: 23 jul. 2019.