Etec Monsenhor Antônio Magliano

HABILITAÇÃO PROFISSIONAL DE TÉCNICO EM INFORMÁTICA

JANELA SEGURA: sistema de automatização para janelas

Maria Julia Siqueira Felix *

Rebeca Corrêa **

Vinícius Henrique Donato de Marco ***

Yasmin Kashimura Herédia de Souza ****

Resumo: O presente projeto visa auxiliar as pessoas acerca do esquecimento e preocupações com as janelas abertas. Dito isso, é possível constatar que grande parte da população ao sair de casa se preocupa em fechar as janelas devido a mudança repentina do clima. Em virtude do fato, o dispositivo desenvolvido pelo grupo tem como finalidade ajudar tais pessoas através de um hardware feito pelo Arduino Uno usando sensor de chuva e servo motor, que ao entrar em contato com a água (chuva), fechará automaticamente a janela, para além disso, ajudará os clientes a terem mais praticidade no dia a dia, visto que o hardware deixará a janela automatizada, podendo ser aberta ou fechada por meio de botões executados no Android Studio.

Palavras-chave: Janela; Sensor; Automatização; Chuva.

* Aluna do curso Técnico em Informática, da Etec Monsenhor Antônio Magliano – maria.felix37@etec.sp.gov.br

** Aluna do curso Técnico em Informática, da Etec Monsenhor Antônio Magliano – rebeca.correa4@etec.sp.gov.br

*** Aluno do curso Técnico em Informática, da Etec Monsenhor Antônio Magliano – vinicius.marco01@etec.sp.gov.br

**** Aluna do curso Técnico em Informática, da Etec Monsenhor Antônio Magliano – yasmin.souza90@etec.sp.gov.br

Abstract: This project aims to develop an automated solution to assist people who often forget to close their windows when leaving home, especially due to sudden

weather changes. A significant portion of the population expresses concern about leaving windows open on rainy days, which motivated the creation of a device based on the Arduino Uno platform. The system consists of a rain sensor and a servo motor that automatically closes the window upon detecting water (rain). Additionally, users can manually open or close the window using a mobile application developed in Android Studio, bringing more convenience to their daily routines.

Keywords: Window; Sensor; Automation; Rain.

1 INTRODUÇÃO

Atualmente o clima no Brasil tem sido bem instável em diversas regiões. Dito isso, um problema recorrente entre os indivíduos é o esquecimento de fechar as janelas, quando deixadas abertas para gerar ventilação, para o aproveitamento da luz natural ou, para a redução de odores e umidade. Assim, principalmente em locais que costumam ficar fechados durante o dia, como apartamentos ou estabelecimentos que só abrem no período noturno. Dessa maneira, caso fosse esquecida alguma janela aberta teria um dispositivo para garantir a segurança do local, se porventura inicia-se a ocorrência de chuva.

Por vez, o ato de fechar as janelas não seja considerado algo de muita relevância, logo, é comum tal esquecimento. No entanto, como consequência desse descuido, pode ocorrer a entrada de água na residência, o que pode danificar móveis, pisos e até mesmo eletrodomésticos. Com base nos problemas apresentados, deseja-se uma solução para a problemática, visando segurança, praticidade e conforto.

O projeto tem por objetivo evitar a ocorrência desse cenário, com a automação residencial surgindo como uma solução eficaz para mitigar esses danos. Esse dispositivo foi projetado para detectar a presença de água, acionando automaticamente o fechamento das janelas e evitando danos materiais significativos.

2 REVISÃO LITERÁRIA

Visando alcançar bons resultados, com o objetivo de ajudar as pessoas e reduzir a preocupação gerada pela mudança climática repentina, a equipe concentrou seus esforços em pesquisas e na melhor forma de integrar isso à realidade do público.

Componentes tecnológicos do projeto:

Placa ESP32

Segundo Marcio Mello (2023) “A ESP32 é uma placa programável via Wi-Fi e Bluetooth, que é integrada com a plataforma de programação das placas Arduino”. O ESP32 é um microcontrolador criado pela Espressif Systems, ele contém múltiplas funcionalidades que o fazem uma das escolhas mais fortes no mercado.

Arduino Uno

O Arduino Uno é uma plataforma eletrônica open source, que foi lançado em 2010 por Massimo Banzi, juntamente com David Cuartielles, Tom Igoe, Gianluca Martino e David Mellis. Essa versão (Uno) foi surgida como uma evolução e padronização das versões anteriores da plataforma. A ideia era criar uma placa de prototipagem eletrônica de baixo custo e fácil de usar, inicialmente voltada para estudantes e artistas interativos. Tendo a finalidade de integrar hardware e software de maneira mais simplificada, podendo servir para controlar elementos mecânicos como luzes e motores, criando mecanismos automatizados. “O Arduino também é capaz de receber e enviar informações pela internet com a ajuda de várias placas de circuitos denominadas shields para Arduino” (Louis, 2018).

Servo Motor

O Servo Motor é um dispositivo eletromecânico que converte sinais de controle em movimento mecânico preciso, serve para controlar com mais exatidão o posicionamento, a velocidade, o torque em sistemas industriais automatizados e é usado para acionar as articulações.

Figma

Figma, foi desenvolvido em 2012 por Dylan Field e Evan Wallace tendo como objetivo criar uma ferramenta de design acessível e gratuita a todos. É um recurso de

design para construção de interfaces digitais, com o objetivo de tornar a prática do design mais acessível.

Por Kellison Ferreira

> “O Figma é uma plataforma online de criação de interfaces, wireframes e protótipos. Seu papel é oferecer recursos de design de telas para aplicações variadas, permitindo que times de Design trabalhem em conjunto no mesmo projeto remotamente e simultaneamente”.

Firebase Authentication

O Firebase é uma plataforma de autenticação desenvolvida pela Google que foi lançado oficialmente em 23 de junho de 2016. O mesmo permite aos desenvolvedores acrescentar funções de login e registro em seus aplicativos de forma rápida e segura, servindo para gerenciamento da autenticação dos usuários em aplicativos web e mobile.

Visual Studio Code

O Visual Studio Code chamado também de IDE é um editor de códigos totalmente gratuito. Ele foi elaborado pela Microsoft e lançado em 2015.

Segundo Ketlin Gonçalves (2024) “O Visual Studio Code é um dos editores de código-fonte mais utilizados na internet. Tanta popularidade vem do fato de ser uma ferramenta extremamente leve, personalizável e eficiente”.

Android Studio

O Android Studio é chamado de Ambiente de Desenvolvimento Integrado, um programa utilizado com o objetivo de elaborar os aplicativos para o sistema operacional Android. Ele foi criado pelo Google e é o sistema operacional móvel mais popular no mundo, usado por diversas marcas.

“As funções do software incluem a edição inteligente de códigos, recursos para design de interface de usuário e análise de performance, entre outras coisas” (ROCHA, 2014).

3 DESENVOLVIMENTO

Site

O site da Janela Segura foi desenvolvido com o objetivo de divulgar o produto e apresentar de forma clara e atrativa, os benefícios que ele oferece. Para isso, foi criada uma landing page completa, estruturada em seções bem definidas que facilitam a navegação e a compreensão das informações: Início, Benefícios, Download App, Planos e Dúvidas.

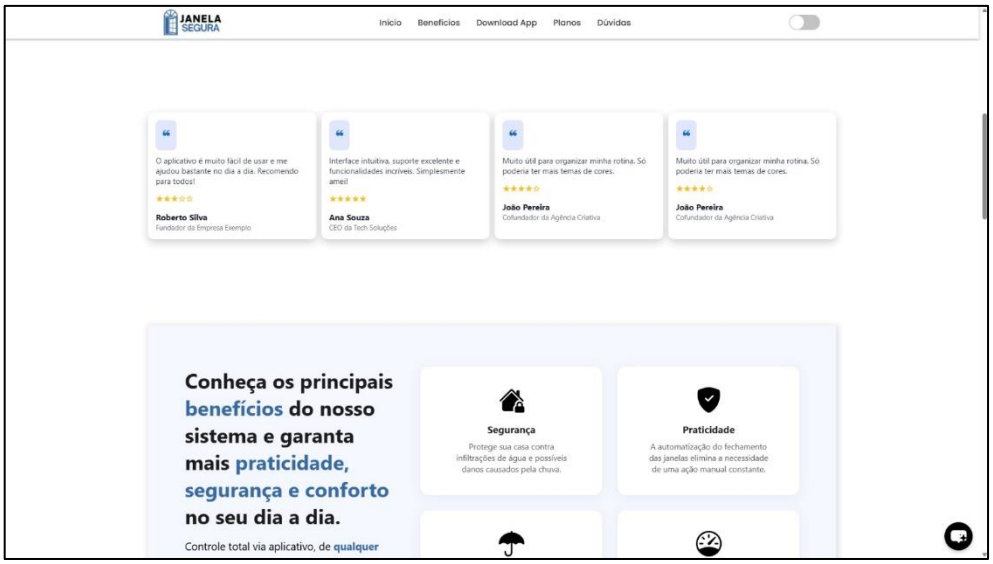
Início:

Figura 1 – Tela Início

Fonte: Elaborada pelos alunos

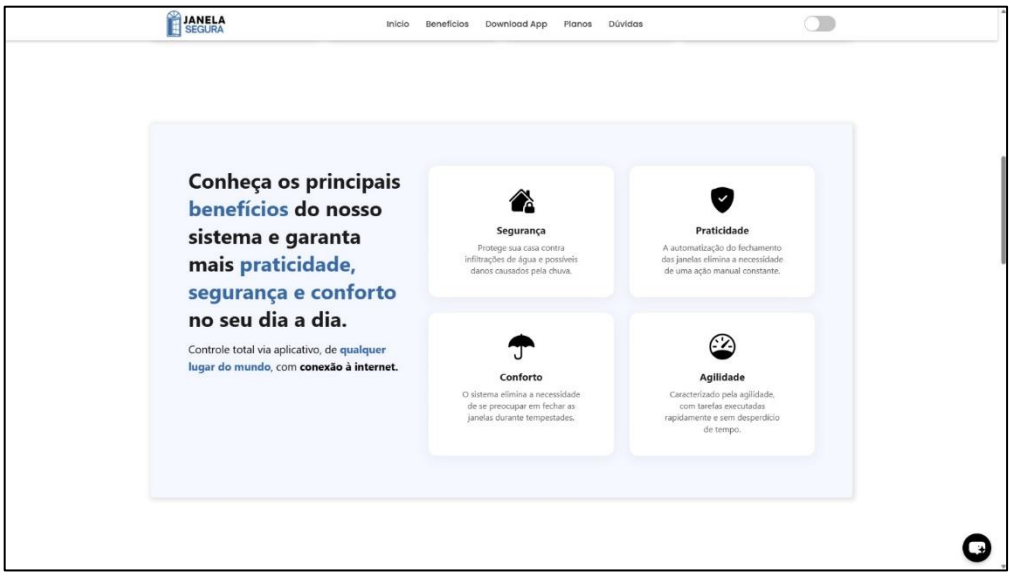
Benefícios:

Figura 2 – Tela Benefícios



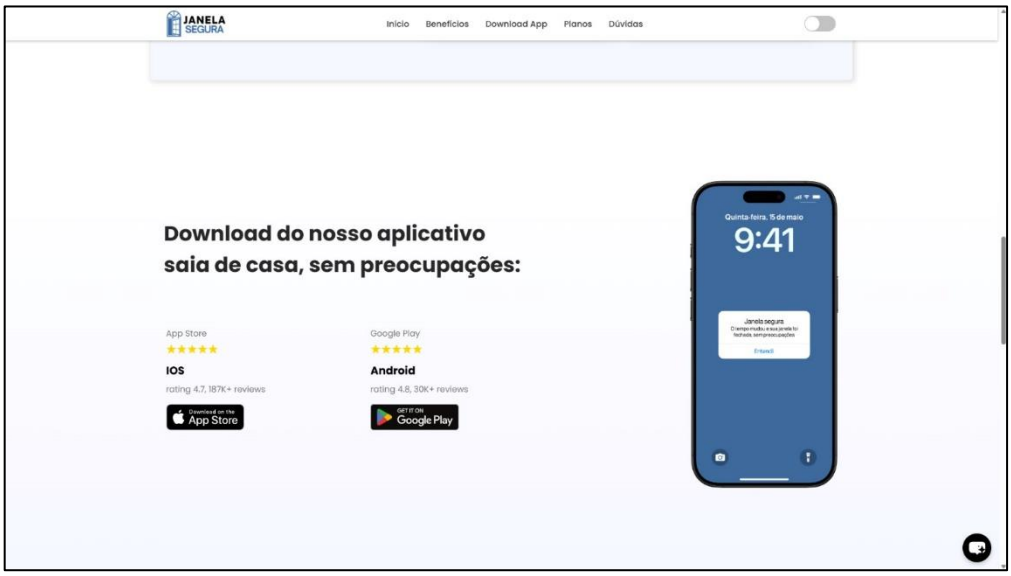
Fonte: Elaborada pelos alunos

Figura 3 – Tela Benefícios



Fonte: Elaborada pelos alunos

Figura 4 – Tela Download App

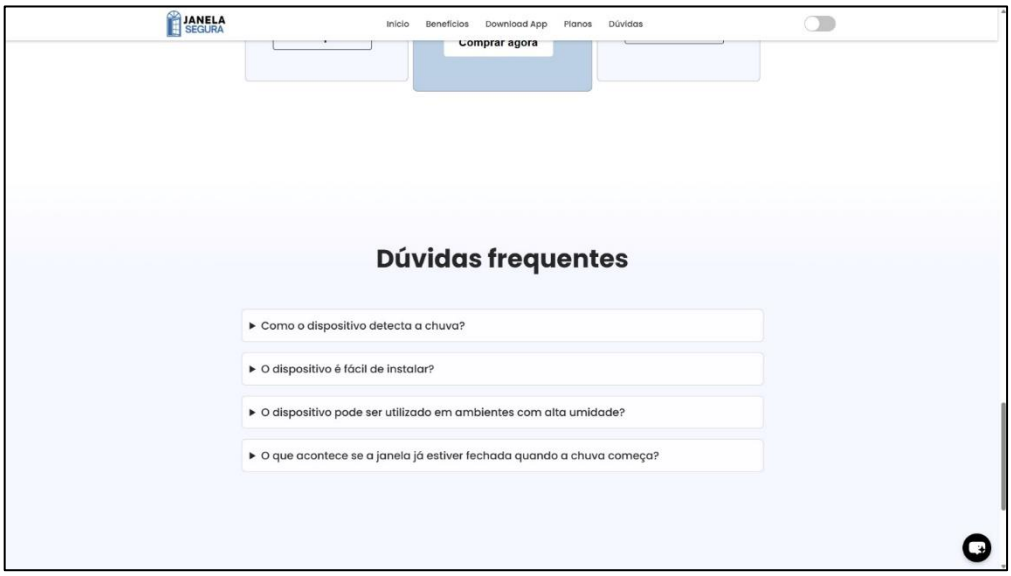


Fonte: Elaborada pelos alunos

Figura 5 – Tela Planos

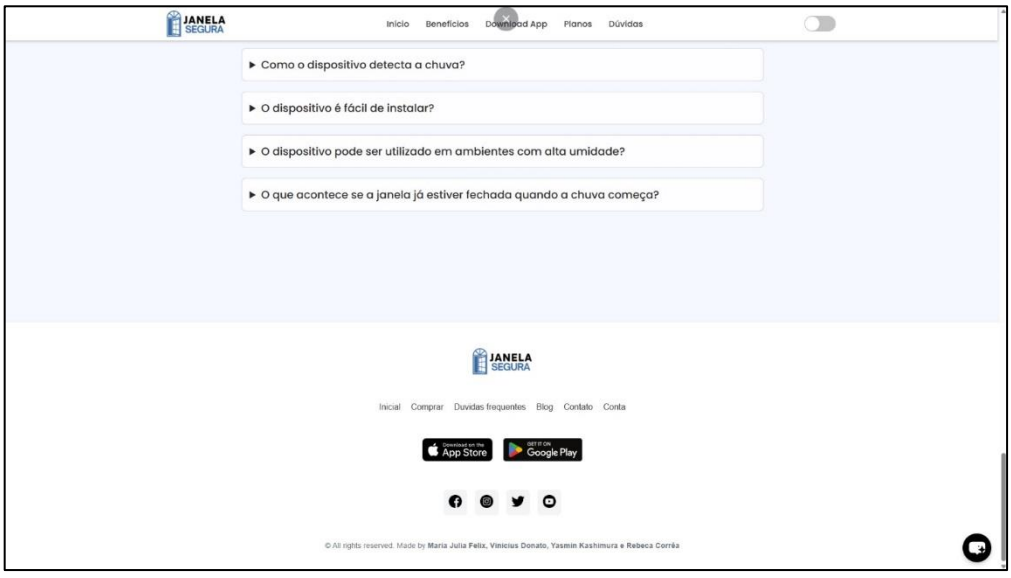
Fonte: Elaborada pelos alunos

Figura 6 – Tela Dúvidas Frequentes



Fonte: Elaborada pelos alunos

Figura 7 – Tela Footer



Fonte: Elaborada pelos alunos

Logo ao acessar o site, o visitante é recebido por um chatbot inteligente, treinado especialmente para responder dúvidas frequentes e auxiliar durante a navegação. Esse recurso foi integrado por meio da API do Chatbase, da OpenAI, garantindo um atendimento ágil, eficiente e personalizado.

Figura 8 - Chatbot

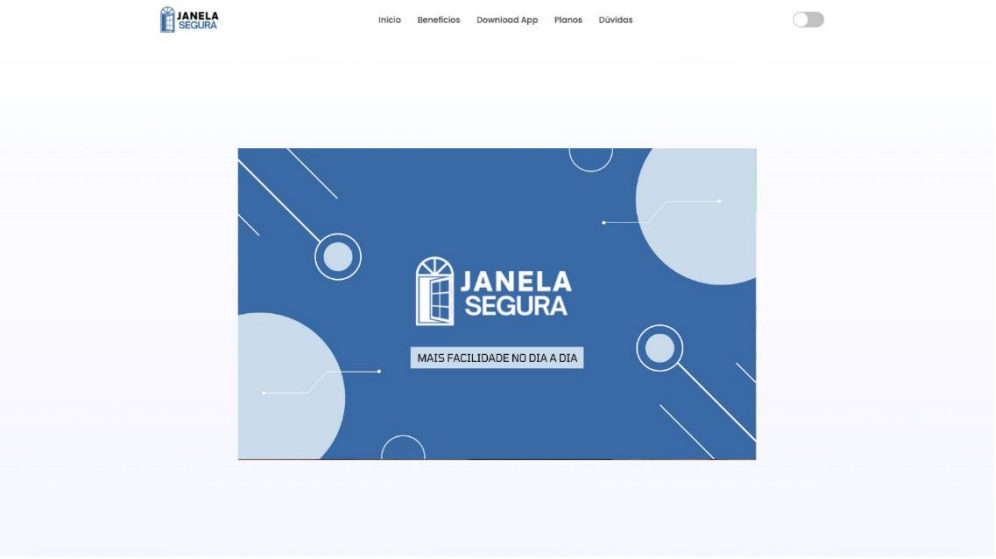
Fonte: Elaborada pelos alunos

Figura 9 - Chatbot

Fonte: Elaborada pelos alunos

Além da landing page principal, o site conta com uma segunda página, acessada por meio do botão “Saiba mais”. Nela, é exibido um vídeo demonstrativo do dispositivo em funcionamento, permitindo uma compreensão mais visual e prática do produto.

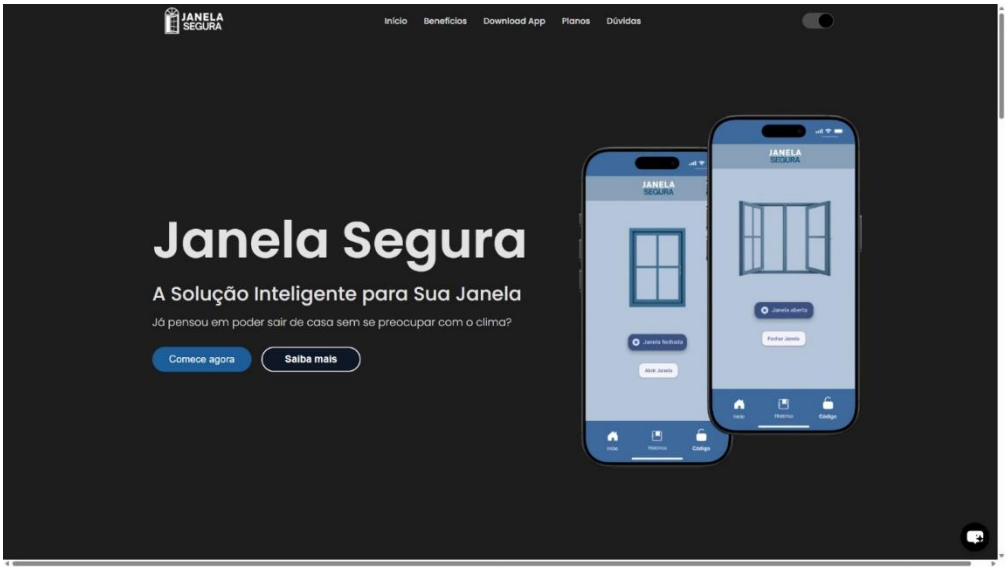
Figura 10 – Tela Saiba mais



Fonte: Elaborada pelos alunos

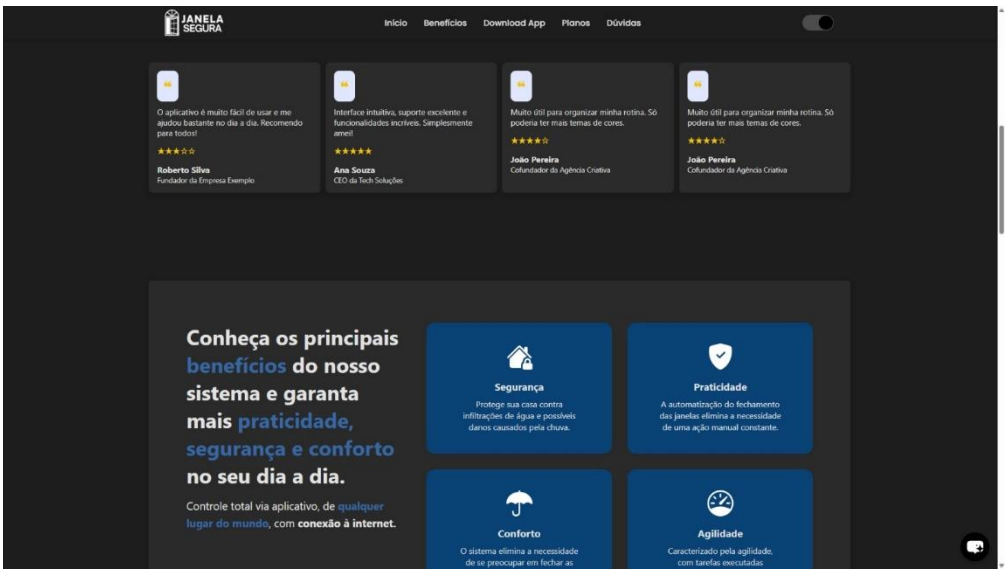
Pensando na experiência do usuário, foi implementado um botão de modo noturno, que permite alternar entre os temas claro e escuro com facilidade. Esse recurso melhora a acessibilidade e proporciona mais conforto visual, especialmente em diferentes horários do dia.

Figura 11 – Tela Início - Noturno



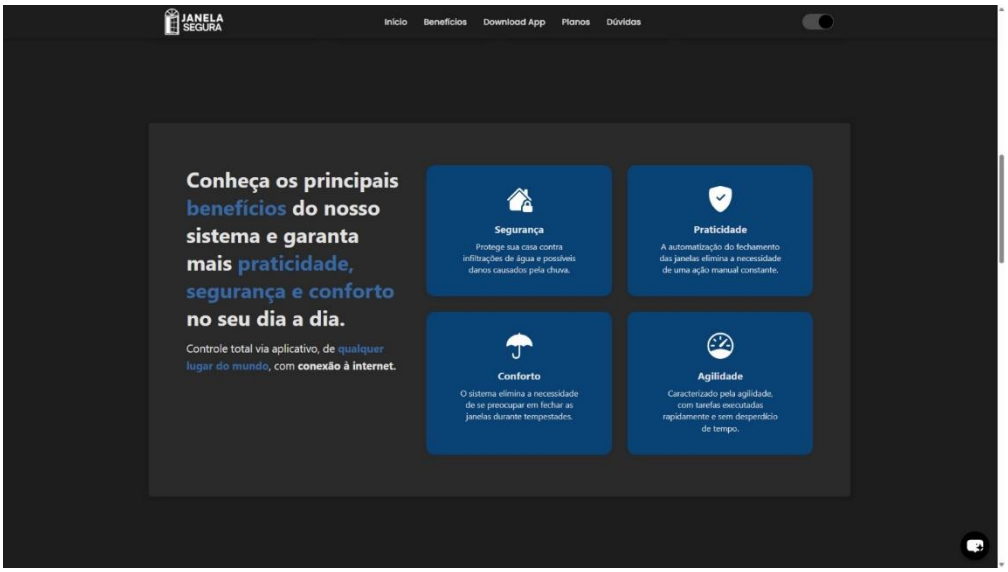
Fonte: Elaborada pelos alunos

Figura 12 – Tela Benefícios – Noturno



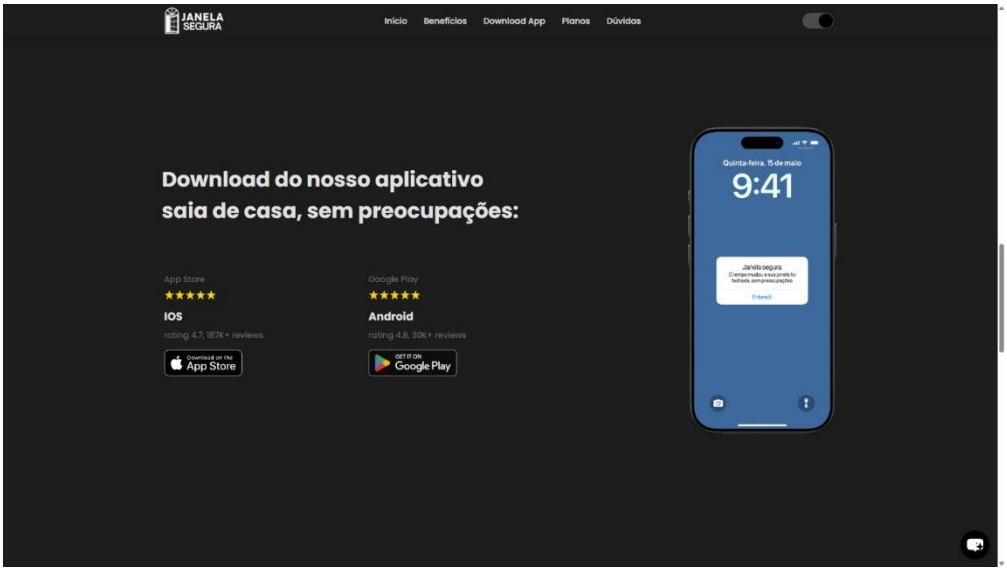
Fonte: Elaborada pelos alunos

Figura 13 – Tela Benefícios - Noturno



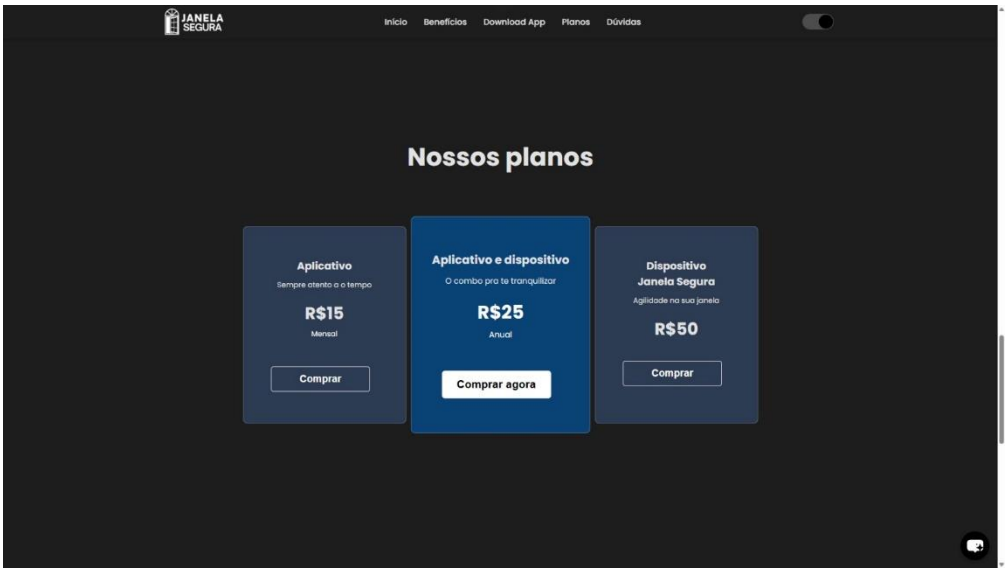
Fonte: Elaborada pelos alunos

Figura 14 – Tela Download App - Noturno



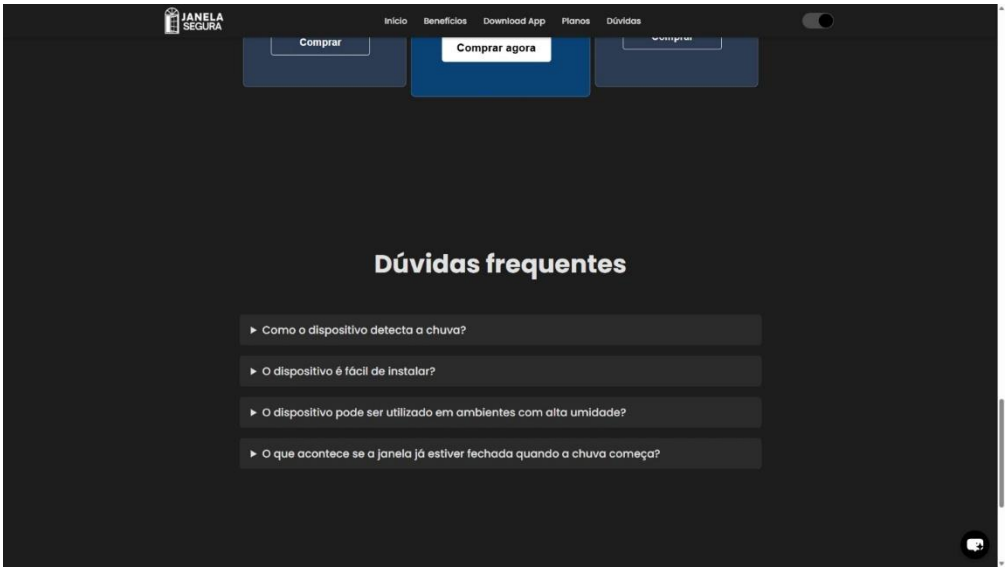
Fonte: Elaborada pelos alunos

Figura 15 – Tela Planos - Noturno



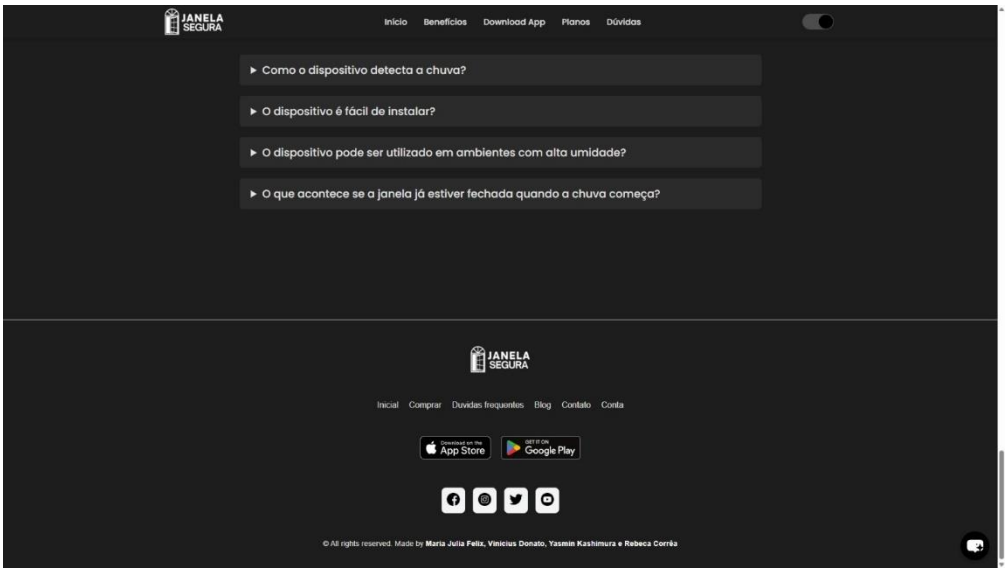
Fonte: Elaborada pelos alunos

Figura 16 – Dúvidas Frequentes - Noturno



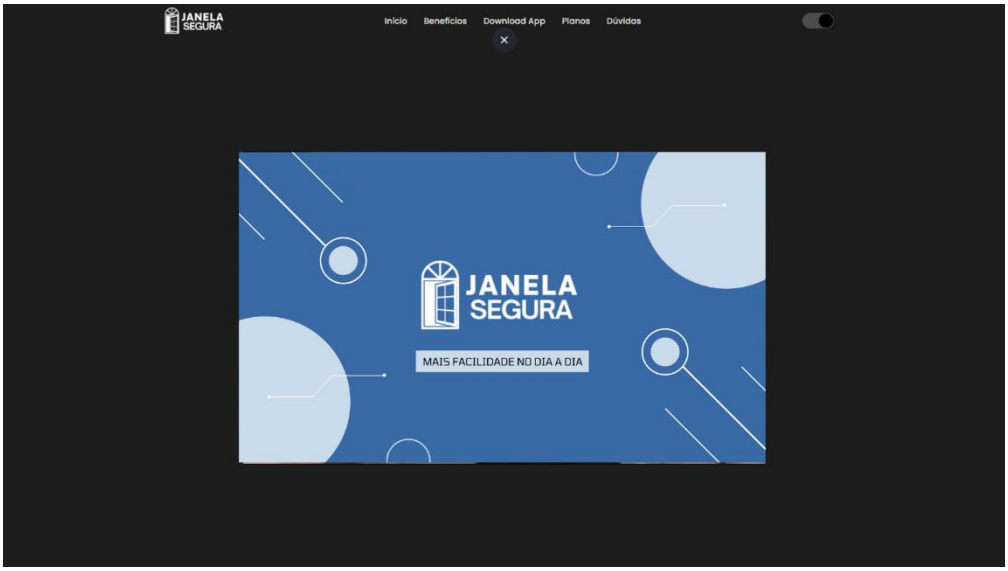
Fonte: Elaborada pelos alunos

Figura 17 – Tela Footer - Noturno



Fonte: Elaborada pelos alunos

Figura 18 – Tela Saiba Mais – Noturno



Fonte: Elaborada pelos alunos

Arduino

Este projeto será desenvolvido utilizando o Arduino. Os componentes serão conectados por meio de uma protoboard, o que permitirá a montagem do circuito de forma prática. O sistema funcionará da seguinte forma: o sensor de chuva será responsável por detectar a presença de água. Quando identificar que está molhado, ele enviará um sinal ao Arduino, que, por sua vez, acionará o servo motor. O servo motor será programado para fechar automaticamente a janela sempre que houver chuva. Para a montagem, serão utilizados cabos jumper do tipo macho-macho e macho-fêmea, os quais permitirão a ligação entre o Arduino, a protoboard e os demais componentes eletrônicos.

O código de programação será desenvolvido na plataforma Arduino IDE utilizando linguagem C/C++. O algoritmo será estruturado para ler os dados do sensor de chuva, interpretar a presença de água, e comandar o movimento do servo motor conforme a necessidade de abrir ou fechar a janela.

Figura 19 – Código Arduino Uno

```
#include <Servo.h>

#define leituraAnalogica A5
#define leituraDigital 7

int valorAnalogico;
int valorDigital;
int estadoAnterior = -1;
int angulo = 0;
int valorAnteriorAnalogico = -1;

Servo meuServo1;
Servo meuServo2;

void setup() {
  Serial.begin(9600);
  pinMode(leituraAnalogica, INPUT);
  pinMode(leituraDigital, INPUT);

  meuServo1.attach(9);
  meuServo2.attach(10);

  meuServo1.write(angulo);
  meuServo2.write(angulo);
}

```

Fonte: Elaborada pelos alunos

Nesta parte do código, é incluída a biblioteca Servo.h, responsável por possibilitar o controle dos servomotores utilizados no projeto. Em seguida, são definidos os pinos utilizados para as leituras analógica (A5) e digital (7) do sensor de chuva, por meio de diretivas #define. Também são declaradas variáveis destinadas ao armazenamento dos valores lidos do sensor, bem como ao controle do ângulo dos servomotores. A variável angulo, por exemplo, é inicializada com valor 0, indicando a posição inicial dos servos (janela fechada).

Na função setup(), é iniciada a comunicação serial com o comando Serial.begin(9600), possibilitando a troca de informações com o monitor serial durante a execução do sistema. Os pinos dos sensores são configurados como entrada, e os dois servomotores são associados aos pinos digitais 9 e 10 por meio do método attach(). Por fim, é definido que ambos os servos devem iniciar na posição estabelecida pela variável angulo, ou seja, 0 graus.

Figura 20 – Segunda parte do código Arduino Uno

```
27  void loop() {
28    valorAnalogico = analogRead(leituraAnalogica);
29    valorDigital = digitalRead(leituraDigital);
30
31    if (valorAnalogico != valorAnteriorAnalogico) {
32
33      valorAnteriorAnalogico = valorAnalogico;
34    }
35
36    if (valorDigital != estadoAnterior) {
37      estadoAnterior = valorDigital;
38
39      if (valorDigital == 0) {
40        Serial.println("Chuva detectada! Fechando janela...");
41        angulo = 0;
42      } else {
43        Serial.println("Sem chuva! Abrindo janela...");
44        angulo = 90;
45      }
46      meuServo1.write(angulo);
47      meuServo2.write(angulo);
48    }
49
50    if (Serial.available()) {
51      String comando = Serial.readStringUntil('\n');
52      comando.trim();
53
54      if (comando == "abrir") {
55        meuServo1.write(180);
56        meuServo2.write(180);
57        Serial.println("status:aberto");
58      } else if (comando == "fechar") {
59        meuServo1.write(90);
60        meuServo2.write(90);
61        Serial.println("status:fechado");
62      } else {
63        Serial.println("status:invalido");
64      }
65
66      delay(100);
67    }
68
69    delay(100);
70  }
```

Fonte: Elaborada pelos alunos

A função loop() representa o ciclo principal do programa, sendo executada repetidamente durante toda a operação do sistema. Inicialmente, são realizadas as leituras dos valores analógico e digital provenientes do sensor de chuva, por meio das funções analogRead() e digitalRead(), respectivamente. O valor analógico é armazenado na variável valorAnalogico, e o digital em valorDigital.

Em seguida, o programa verifica se o valor analógico atual é diferente do anterior, armazenado em valorAnteriorAnalogico. Caso haja alteração, essa nova leitura é registrada na variável correspondente. Apesar disso, neste trecho específico,

não é realizada nenhuma ação com base na leitura analógica — o código apenas detecta a mudança.

Logo após, é feita a comparação entre o valor digital atual do sensor e o valor anterior (estadoAnterior). Caso tenha ocorrido uma alteração, o novo valor é atualizado. Se for detectado o valor 0, que indica a presença de chuva, o sistema envia a mensagem "Chuva detectada! Fechando janela..." ao monitor serial, e define o ângulo dos servomotores como 0, fechando a janela. Caso contrário, ou seja, quando valorDigital for diferente de 0, é enviada a mensagem "Sem chuva! Abrindo janela...", e os servomotores são ajustados para o ângulo de 90 graus, abrindo parcialmente a janela. Os ângulos são aplicados aos dois servos por meio do comando write().

Além do controle automático com base na leitura do sensor, a função loop() também verifica se há algum comando recebido pela porta serial. Caso exista, o comando é lido e tratado. Se for recebido o texto "abrir", os servos são movidos para 180 graus, representando uma abertura total da janela, e é exibida a mensagem "status:aberto". Se o comando for "fechar", os servos são movidos para 90 graus, e é exibida a mensagem "status:fechado". Para qualquer outro comando inválido, é exibida a mensagem "status:invalido".

Por fim, são utilizados pequenos atrasos (delay(100)) para evitar leituras e atualizações excessivamente rápidas, permitindo a estabilidade da leitura dos sensores e do controle dos servomotores.

ESP32

A placa ESP32 será responsável pela conexão com a rede Wi-Fi e pela comunicação entre o Arduino e o aplicativo móvel, utilizando o protocolo MQTT. O ESP32 ficará monitorando o broker MQTT, aguardando um comando enviado pelo aplicativo. Esse comando será publicado no tópico janela/comando com os valores "abrir" ou "fechar". Assim que um comando for recebido, o ESP32 repassará a instrução ao Arduino Uno, que irá acionar os servos motores para abrir ou fechar a janela conforme solicitado.

A comunicação entre o ESP32 e o Arduino Uno será feita através dos pinos GND, transmissão (TX) e recepção (RX) de ambos os dispositivos, utilizando uma protoboard. O ESP32 enviará informações ao Arduino por meio do pino TX, enquanto o Arduino receberá as informações pelo pino RX. O processo será invertido para a comunicação de retorno: o Arduino enviará o status ("aberto" ou "fechado") pelo pino TX, e o ESP32 receberá essa informação pelo pino RX.

O ESP32 se conectará ao broker MQTT disponível em tcp://test.mosquitto.org:1883, onde publicará o status da janela no tópico janela/status, permitindo que o aplicativo móvel acesse e mostre em tempo real o estado da janela.

A implementação do código será realizada na plataforma Arduino IDE, utilizando a linguagem C++.

Figura 21 – Código ESP32

```
#include <WiFi.h>
#include <PubSubClient.h>

const char* nomewifi = "wifi";
const char* senhawifi = "senha";

const char* mqtt_server = "test.mosquitto.org";
const char* mqtt_comando = "janela/comando";
const char* mqtt_status = "janela/status";


WiFiClient espClient;
PubSubClient client(espClient);

void setup() {
  Serial.begin(115200);
  Serial2.begin(9600);

  conectarWiFi();
  client.setServer(mqtt_server, 1883);
  client.setCallback(recebeComando);
}

```

Fonte: Elaborada pelos alunos

No início do código, são incluídas duas bibliotecas essenciais para o funcionamento do ESP32: a WiFi.h, que permite ao ESP32 se conectar a redes Wi-Fi, e a PubSubClient.h, responsável pela comunicação MQTT, um protocolo eficiente para troca de mensagens entre dispositivos. Com essas bibliotecas, o ESP32 será

capaz de se conectar à rede e interagir com um servidor MQTT para enviar e receber comandos.

Em seguida, são definidas algumas constantes que armazenam as credenciais da rede Wi-Fi e os tópicos do servidor MQTT. As variáveis nomewifi e senhawifi guardam, respectivamente, o nome da rede e a senha para conectar o ESP32 à internet. As constantes mqtt_server, mqtt_comando e mqtt_status armazenam o endereço do servidor MQTT e os tópicos que o ESP32 utilizará para enviar e receber informações. O tópico janela/comando será usado para enviar comandos de controle da janela, como "abrir" e "fechar", enquanto o tópico janela/status será utilizado para enviar o status atual da janela.

Logo após, são criados dois objetos importantes: o WiFiClient espClient, que gerencia a conexão Wi-Fi do ESP32, e o PubSubClient client(espClient), que permite ao ESP32 se conectar e interagir com o servidor MQTT.

A função setup() é executada uma única vez quando o dispositivo é ligado ou reiniciado. Dentro dela, a comunicação serial é inicializada, tanto para depuração com o computador quanto para comunicação com o Arduino, usando o Serial2. A função conectarWiFi() é chamada para conectar o ESP32 à rede Wi-Fi configurada. Em seguida, a função client.setServer() configura o servidor MQTT com o endereço e a porta 1883, que é o padrão para MQTT. Por fim, a função client.setCallback() define a função que será chamada sempre que uma mensagem for recebida no servidor MQTT, permitindo que o ESP32 reaja a comandos recebidos, como abrir ou fechar a janela.

Figura 22 – Segunda parte do código ESP32

```
24  void loop() {
25    if (WiFi.status() != WL_CONNECTED) {
26      conectarWiFi();
27    }
28
29
30    if (!client.connected()) {
31      reconectarMQTT();
32    }
33
34    client.loop();
35
36
37    if (Serial2.available()) {
38      String status = Serial2.readStringUntil('\n');
39      status.trim();
40
41      if (status.startsWith("status:")) {
42        Serial.print("Enviando status para MQTT: ");
43        Serial.println(status);
44        client.publish(mqtt_status, status.c_str());
45      }
46    }
47
48    delay(10);
49  }
50
51
52  void conectarWiFi() {
53    Serial.print("Conectando-se a ");
54    Serial.println(nomewifi);
55    WiFi.begin(nomewifi, senhawifi);
56    while (WiFi.status() != WL_CONNECTED) {
57      delay(500);
58      Serial.print(".");
59    }
60    Serial.println("\nWi-Fi conectado!");
61    Serial.print("IP: ");
62    Serial.println(WiFi.localIP());
63  }
64
```

Fonte: Elaborada pelos alunos

Essa parte do código é responsável por garantir a manutenção da conexão Wi-Fi e MQTT, além de ler e enviar o status da janela para o servidor MQTT. A função loop() é executada continuamente e mantém o sistema funcionando de maneira dinâmica, verificando a conexão de rede e processando dados em tempo real.

O código começa verificando o status da conexão Wi-Fi. Caso o ESP32 não esteja conectado (WiFi.status() != WL_CONNECTED), a função conectarWiFi() é chamada para tentar restabelecer a conexão com a rede. Essa função tenta se reconectar à rede Wi-Fi até que a conexão seja bem-sucedida.

Em seguida, o código verifica se a conexão MQTT está ativa. Se o cliente MQTT não estiver conectado, a função reconectarMQTT() é chamada para realizar

uma nova tentativa de conexão com o servidor MQTT. Isso garante que o ESP32 consiga se reconectar automaticamente sempre que a conexão MQTT for perdida.

A função client.loop() é chamada dentro do loop() principal para garantir que a comunicação MQTT continue ativa e processando as mensagens que chegam ao ESP32. Esta função é fundamental para a manutenção da conexão e para a recepção de mensagens no tópico MQTT.

Depois, o código verifica se há dados disponíveis para leitura na comunicação serial com o Arduino (via Serial2). Caso haja dados, ele lê a string até a quebra de linha e remove quaisquer espaços em branco ao redor do texto com o comando trim(). Se o status lido começar com a palavra "status:", o código envia esse status ao servidor MQTT no tópico janela/status. Isso permite que o sistema atualize continuamente o status da janela no servidor MQTT.

Por fim, o código faz uma pausa breve com o comando delay(10) para não sobrecarregar o processador e garantir que o sistema opere de forma estável, permitindo também a execução das funções de conexão e comunicação de forma eficiente.

A função conectarWiFi() realiza a conexão do ESP32 à rede Wi-Fi. Ela começa exibindo no monitor serial o nome da rede à qual o dispositivo está tentando se conectar. A função WiFi.begin(nomewifi, senhawifi) é chamada para iniciar a conexão. A função entra em um laço (while) e tenta se conectar à rede até que a conexão seja bem-sucedida. Durante esse processo, pontos de status são exibidos no monitor serial. Quando a conexão for estabelecida, o IP local do ESP32 é exibido, confirmando que o dispositivo está conectado à rede.

Figura 23 – Terceira parte do código ESP32

```
65  void reconectarMQTT() {
66    while (!client.connected()) {
67      Serial.print("Conectando ao MQTT...");
68      String clientId = "ESP32Client-" + String(random(0xffff), HEX);
69      if (client.connect(clientId.c_str())) {
70        Serial.println("Conectado!");
71        client.subscribe(mqtt_comando);
72      } else {
73        Serial.print("Falha, rc=");
74        Serial.print(client.state());
75        Serial.println(" Tentando novamente em 2 segundos...");
76        delay(2000);
77      }
78    }
79  }
80
81
82  void recebeComando(char* topic, byte* payload, unsigned int length) {
83    String comando = "";
84    for (unsigned int i = 0; i < length; i++) {
85      comando += (char)payload[i];
86    }
87    comando.trim();
88
89    Serial.print("Comando recebido: ");
90    Serial.println(comando);
91
92    if (comando == "abrir" || comando == "fechar") {
93      Serial2.println(comando);
94      Serial.println("Comando enviado ao Arduino via Serial2");
95    } else {
96      Serial.println("Comando inválido ignorado.");
97    }
98  }
```

Fonte: Elaborada pelos alunos

Essa parte do código trata da reconexão do ESP32 ao servidor MQTT caso a conexão seja perdida, e também da recepção e tratamento dos comandos enviados para o controle da janela, via MQTT.

A função reconectarMQTT() é responsável por garantir que o ESP32 se conecte ao servidor MQTT. Quando o cliente MQTT não está conectado, o código entra em um laço while que tenta estabelecer a conexão até que ela seja bem-sucedida. Dentro desse laço, um ID único é gerado para o cliente MQTT utilizando a função random(0xffff) e convertendo-o para hexadecimal. Esse ID é necessário porque o servidor MQTT exige que cada cliente tenha um identificador único.

Após gerar o ID, o código tenta se conectar ao servidor MQTT com a função client.connect(). Se a conexão for bem-sucedida, o ESP32 se inscreve no tópico janela/comando através da função client.subscribe(mqtt_comando), o que permite

que o ESP32 receba mensagens enviadas para esse tópico. Caso a conexão falhe, o código exibe o código de erro (usando client.state()) e tenta novamente após um intervalo de 2 segundos.

A função recebeComando() é chamada sempre que o ESP32 recebe uma mensagem no tópico MQTT ao qual está inscrito. O payload (conteúdo da mensagem) é recebido como um vetor de bytes, e a função converte esse vetor para uma string. A string resultante é armazenada na variável comando, e o método trim() é usado para remover quaisquer espaços extras ao redor do texto.

Após a conversão do payload, o código verifica se o comando recebido é "abrir" ou "fechar". Se for um comando válido, ele é enviado ao Arduino via Serial2.println(comando), que fará o Arduino executar a ação correspondente (abrir ou fechar a janela, dependendo do comando). O ESP32 também imprime no monitor serial a mensagem "Comando enviado ao Arduino via Serial2" para indicar que o comando foi repassado para o Arduino.

Se o comando recebido não for válido, o código imprime a mensagem "Comando inválido ignorado" no monitor serial, informando que o comando não foi reconhecido e não será executado.

Aplicativo Mobile

O presente projeto apresenta por sua vez, o desenvolvimento de um aplicativo que visa auxiliar o usuário no dia a dia contendo formas práticas e automatizadas, com o intuito de efetuar o fechamento da janela. A partir disso, o aplicativo realizado no Android Studio contém quatro páginas, sendo o “login”, “histórico”, “cadastro do código” e a “tela principal”.

Em continuidade ao exposto, o aplicativo se inicia com a página de “login” que tem como objetivo adicionar o usuário por meio de seu e-mail e a criação de uma senha. Logo é avançado para a “tela principal”, que servirá para o acionamento da abertura e fechamento da janela a partir de botões e sem a necessidade do contato com a água. Após essa tela, contém o “Histórico”, que terá como finalidade registrar e guardar informações sobre as atividades feitas no hardware, como por exemplo o horário e data em que a janela foi aberta ou fechada, se foi acionada pelo contato com

a água ou se foi por acionamento pelo próprio aplicativo. Por fim, contém a tela “Código” que tem como objetivo desempenhar a função de registrar o código único que contém embaixo do hardware para realizar a conexão do mesmo com o aplicativo.

Figura 24 – Tela Entrada

Fonte: Elaborada pelos alunos

Figura 25 – Tela Login e Cadastro

Fonte: Elaborada pelos alunos

Figura 26 – Tela Botão de acionamento

Fonte: Elaborada pelos alunos

Figura 27 – Tela Histórico

Fonte: Elaborada pelos alunos

Figura 28 – Tela Código

Fonte: Elaborada pelos alunos

4 CONSIDERAÇÕES FINAIS

O objetivo geral da proposta apresentada, teve como finalidade facilitar o dia a dia da sociedade visando a praticidade, conforto, segurança e diminuição da preocupação. A ideia central foi o fechamento automático das janelas ao entrar em contato com a água (chuva), porém, também foi proposto que elas pudessem ser acionadas para serem fechadas ou abertas a partir do aplicativo desenvolvido.

Considerando atentamente os objetivos propostos para o projeto no primeiro momento, foi possível observar que houve diversas mudanças ao decorrer do trabalho como, a criação de um Chatbot no site que servirá para o esclarecimento de possíveis dúvidas, a inclusão dos botões de acionamento de abertura e fechamento das janelas, a tela do histórico que visa armazenar os horários e datas em que a janela foi acionada – tanto por meio do aplicativo, quanto pelo fechamento a partir de chuva – e, também os valores de venda do hardware no site.

Conclui-se, portanto, que os objetivos propostos foram alcançados obtendo mudanças ao decorrer do desenvolvimento do projeto, atingindo o objetivo final.

REFERÊNCIAS

CARDOSO, Alan. Fim do La Niña deixa clima instável e irregular no Brasil, diz previsão. **CNN Brasil**, São Paulo, 10 de abr. de 2025. Disponível em: <https://www.cnnbrasil.com.br/nacional/brasil/fim-do-la-nina-deixa-clima-instavel-e-irregular-no-brasil-diz-previsao/?utm_source>. Acesso em: 22 de abr. de 2025.

CHARLEAUX, Lupa; LIMA, Lucas. O que é um sistema operacional? Conheça os tipos existentes. **Tecnoblog**, 2025. Disponível em: <https://tecnoblog.net/responde/o-que-e-um-sistema-operacional/>. Acesso em: 10 de jun. de 2025.

DEVMEDIA. Guia completo do Visual Studio Code, 2023. Disponível em: <https://www.devmedia.com.br/guia-completo-do-visual-studio-code/43827>. Acesso em: 10 de jun. de 2025.

DEVMEDIA. Introdução ao Visual Studio Code, 2016. Disponível em: < https://www.devmedia.com.br/introducao-ao-visual-studio-code/34418>. Acesso em: 10 de jun. de 2025.

EBAC ONLINE. O que é Figma e como usar?, 2023. Disponível em: <https://ebaconline.com.br/blog/o-que-e-figma-e-como-usar>. Acesso em: 21 de maio de 2025.

GONÇALVES, Ketlin. Como usar o Visual Studio Code?. **HostGator**, 2024. Disponível em: < https://www.hostgator.com.br/blog/visual-code-studio/>. Acesso em: 10 de jun. de 2025.

GUSE, Rosana. Controlando um servo motor com Arduino: 6 modos diferentes. **Maker Hero**, 2024. Disponível em: < https://www.makerhero.com/blog/controlando-um-servo-motor-com-arduino/?srsltid=AfmBOoo_7D4ZfasEkYBlNr6nQNIhkeZ1szrSDtyICb3-1rZeoNL6ue5E >. Acesso em: 22 de maio de 2025.

HARADA, Eduardo Yukio. O que é o Android Studio, ferramenta criada para desenvolver apps mobile. **Tecmundo**, 2019. Disponível em: <https://www.tecmundo.com.br/software/146361-o-android-studio-ferramenta-criada-desenvolver-apps-mobile.htm>. Acesso em: 10 de jun. de 2025.

MELLO, Marcio. Placa ESP32: Descubra o que é, para que serve e muito mais!. **Victor Vision**, 2023. Disponível em: < https://victorvision.com.br/blog/placa-esp32/#:~:text=A%20ESP32%20%C3%A9%20uma%20placa,para%20criar%20seus%20pr%C3%B3prios%20projetos.>. Acesso em: 10 de jun. de 2025.

ROCHA, Leonardo. Android Studio: ferramenta de criação de apps da Google ganha versão 1.0. **Tecmundo**, 2014. Disponível em: <https://www.tecmundo.com.br/android/69111-android-studio-ferramenta-criacao-apps-google-ganha-versao-1-0.htm>. Acesso em: 10 de jun. de 2025.

ROCK CONTENT. Conheça Firebase: a ferramenta de desenvolvimento e análise de aplicativos mobile, 2021. Disponível em: <https://rockcontent.com/br/blog/firebase/>. Acesso em: 21 de maio de 2025.

RODRIGUES, Raísa. O primo rico do Arduino: conheça o ESP32. **3E UNICAMP**, 2025. Disponível em: <https://www.3eunicamp.com/post/o-primo-rico-do-arduino-conhe%C3%A7a-o-esp32?gad_source=1&gad_campaignid=20527797266&gbraid=0AAAADMRGQG3YGIfG9QlOE6WBkv7EcFDI&gclid=EAIaIQobChMIxJvvo_LnjQMV0YBaBR27AwK7EAAYASAAEgI3YfD_BwE>. Acesso em: 10 de jun. de 2025.

UNIVERSAL ROBOTS. Servo Motor: o que é, como funciona e seu uso na robótica industrial, 2023. Disponível em: <https://www.universal-robots.com/br/blog/servo-motor-o-que-%C3%A9-como-funciona-e-seu-uso-na-rob%C3%B3tica-industrial/>. Acesso em: 20 de maio de 2025.