Universidade Federal de Campina Grande
Centro de Engenharia Elétrica e Informática
Departamento de Engenharia Elétrica

Trabalho de Conclusão de Curso

Desenvolvimento de uma interface gráfica para o monitoramento, automação e controle de ar-condicionado

Bruna Eduarda Cruz Maciel

Campina Grande - PB
Outubro de 2024

Bruna Eduarda Cruz Maciel

Desenvolvimento de uma interface gráfica para o monitoramento, automação e controle de ar-condicionado

Trabalho de Conclusão de Curso submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletricista.

Universidade Federal de Campina Grande - UFCG

Centro de Engenharia Elétrica e Informática - CEEI

Departamento de Engenharia Elétrica - DEE

Coordenação de Graduação em Engenharia Elétrica - CGEE

Jaidilson Jó da Silva, D.Sc.

(Orientador)

Campina Grande - PB

Outubro de 2024

Bruna Eduarda Cruz Maciel

Desenvolvimento de uma interface gráfica para o monitoramento, automação e controle de ar-condicionado

Trabalho de Conclusão de Curso submetido à Coordenação de Graduação em Engenharia Elétrica da Universidade Federal de Campina Grande como parte dos requisitos necessários para a obtenção do grau de Engenheiro Eletricista.

Aprovado em _____ /_____ /_____

Georgina Karla de Freitas Serres
Universidade Federal de Campina Grande
Avaliadora

Jaidilson Jó da Silva
Universidade Federal de Campina Grande
Orientador

Campina Grande - PB
Outubro de 2024

Dedico este trabalho a Deus,que me proporcionou sabedoria para superar desafios e aos meus pais, pela educação, amor e cuidado ao longo de toda a minha vida e me deram oportunidades de chegar onde cheguei.

Agradecimentos

À Deus por me proporcionar a benção de poder realizar um curso superior, e me manter perseverante na elaboração deste trabalho, me trazendo foco e me direcionando no caminho correto. Em cada desafio, pude perceber Sua providência cuidando dos detalhes e abrindo portas que jamais imaginei. É pela Sua infinita graça que esta caminhada foi possível, e eu reconheço que tudo o que alcancei é fruto da Sua vontade e destinado à Sua glória.

À minha família, um agradecimento especial. Vocês sempre estiveram presentes em todos os momentos da minha vida. O apoio incondicional e a fé que depositaram em mim foram essenciais para que eu alcançasse este marco. Em especial, sou eternamente grata à meus pais, Marta e Jailton, por todo sacrificio, apoio e nunca deixarem faltar nada,principalmente o mais importante,amor.

Aos amigos que compartilharam esta jornada comigo na universidade, dividindo noites em claro e celebrando conquistas, saibam que os laços que criamos são valiosos e levarei comigo por toda a vida. Agradeço por sempre acreditarem em mim.

Aos amigos que me acompanham desde a escola e aos que fiz ao longo do caminho, sou grata por estarem sempre presentes e por serem fontes de apoio e amizade verdadeira. Obrigada por me mostrarem que posso contar com vocês em qualquer momento.

A todos que contribuíram para este trabalho, oferecendo auxílio e ideias valiosas, meu sincero agradecimento por seu apoio fundamental.

Por fim, agradeço aos professores que me guiaram durante esta trajetória acadêmica. Agradeço pela orientação compartilhada e pela inspiração que me proporcionaram.

"Dificuldades preparam pessoas comuns para destinos extraordinários",
(C.S Lewis)

Resumo

Neste trabalho é apresentado o desenvolvimento de um aplicativo para controle e monitoramento de aparelhos de ar-condicionado, visando a redução do consumo desnecessário de energia. Para tal projeto, foi criado um sistema que permite o monitoramento e controle dos ar-condicionados, utilizando uma placa de desenvolvimento ESP32. O sistema foi projetado para operar tanto remotamente quanto localmente, integrando-se a plataformas de Internet das Coisas (IoT) para um gerenciamento eficiente. Além disso, o aplicativo, desenvolvido em React Native, proporciona uma interface intuitiva que facilita o controle dos aparelhos, tornando a experiência acessível a todos os usuários. O backend utiliza Firebase para autenticação de usuários e monitoramento em tempo real dos dados dos aparelhos, garantindo segurança e eficiência na troca de informações. Os resultados demonstram a eficácia do sistema na otimização do uso de energia elétrica.

Palavras-chave: Internet das Coisas (IoT), Automação, Ar-Condicionado, ESP32, Aplicativo,React Native,Firebase.

Abstract

This work presents the development of an application for controlling and monitoring air conditioning devices, aiming to reduce unnecessary energy consumption. For this project, a system was created that allows the monitoring and control of air conditioners, using an ESP32 development board. The system is designed to operate both remotely and locally, integrating with Internet of Things (IoT) platforms for efficient management. Furthermore, the application, developed in React Native, provides an intuitive interface that facilitates device control, making the experience accessible to all users. The backend uses Firebase for user authentication and real-time monitoring of device data, ensuring security and efficiency in the exchange of information. The results demonstrate the effectiveness of the system in optimizing the use of electrical energy.

Keywords: Internet of Things (IoT), Automation, Air Conditioning, ESP32, Application, React Native, Firebase.

Lista de ilustrações

Lista de abreviaturas e siglas

UFCG	Universidade Federal de Campina Grande
IoT	*Internet of Things*
SDK	*Sofware devolopment kit*
NoSQL	*Not Only Structured Query Language*
API	*Application Programming Interface*
IR	*Infrared*
FCM	*Firebase Cloud Messaging*
AI	*Artificial Intelligence*
Wi-Fi	*Wireless Fidelity*
IDE	*Integrated Development Environment*
JSON	*JavaScript Object Notation*
CLI	*Command-Line Interface*

Sumário

1 Introdução

A demanda por energia elétrica cresce continuamente em escala global, impulsionada, sobretudo, pelo progresso econômico. Indicadores socioeconômicos, como o PIB (Produto Interno Bruto), revelam uma correlação significativa entre o aumento econômico e a demanda de energia de um país. No Brasil, essa tendência é clara, evidenciada pelo recorde histórico de consumo de energia registrado em março de 2022, conforme relatado pela Empresa de Pesquisa Energética (EPE).

Segundo o relatório da EPE, houve um aumento no consumo de energia elétrica, impulsionado pelas altas temperaturas em várias partes do país. Esse contexto resultou em um crescimento na aquisição de equipamentos elétricos, especialmente ar-condicionados. A EPE também aponta que a melhora nos indicadores econômicos teve impacto no aumento do consumo residencial. O aumento da quantidade de unidades vendidas permitiu que os condicionadores de ar apresentassem um ganho relevante na participação do consumo total de eletricidade nos últimos anos, cuja participação passou de 7% em 2005 para 14% em 2017. De acordo com a nota técnica 030/2018 da EPE, os ar-condicionados são um dos maiores consumidores de energia nas residências. (EPE, 2024).

Em 2018, a Agência Internacional de Energia divulgou o relatório "The Future of Cooling", que alertava sobre uma iminente crise de resfriamento global devido à crescente demanda por ar-condicionado, prevendo que essa demanda poderia triplicar até 2050, representando 10% do consumo global de eletricidade.(IEA, 2018).

Em 2021, o Brasil enfrentou uma crise elétrica provocada por baixos índices de chuva, resultando em aumento das tarifas de eletricidade e ressaltando a necessidade de economia de energia. Para combater esse desperdício de maneira econômica e sustentável, vários estudos apresentam soluções que utilizam sistemas de monitoramento e controle, com o intuito de reduzir gastos energéticos, ao mesmo tempo em que oferecem conforto aos usuários.

Nesse contexto, a importância de aplicativos dedicados para o controle de dispositivos é fundamental. Esses aplicativos oferecem aos usuários uma interface intuitiva para monitorar o consumo energético em tempo real, programar horários de funcionamento e ajustar configurações conforme a necessidade. Com essas ferramentas, é possível evitar o uso excessivo de energia, otimizando o funcionamento dos aparelhos e resultando em economia significativa nas contas de eletricidade. Além disso, a capacidade de receber notificações e relatórios sobre o uso de energia permite que os usuários façam escolhas mais conscientes e sustentáveis em relação ao consumo energético em seus lares.

1.1 Objetivos

Esta seção tem por finalidade apresentar e descrever os objetivos gerais e específicos deste trabalho.

1.1.1 Objetivo Geral

Estudar e desenvolver um aplicativo para controlar um dispositivo IoT que automatize o controle de ar-condicionado, assegurando a compatibilidade do sistema em diversos ambientes e promovendo a redução do consumo de energia. A proposta envolve a exploração de recursos e técnicas para criar um dispositivo de fácil utilização, além de garantir que o aplicativo permita um monitoramento e controle eficientes dos ar-condicionados. O aplicativo será projetado para oferecer uma interface intuitiva, facilitando a interação do usuário com o dispositivo e assegurando um gerenciamento eficaz do consumo energético.

1.1.2 Objetivos Específicos

Integração com Firebase:

- Implementar a leitura e gravação de dados no Realtime Database do Firebase para monitorar o status dos ar-condicionados (ligado/desligado, temperatura, modo de operação).
- Garantir a atualização em tempo real das informações no aplicativo, permitindo que os usuários vejam o estado atual dos dispositivos.

Interface do Usuário (UI):

- Projetar uma interface no figma intuitiva para React Native, que permita fácil navegação e controle dos ar-condicionados.

Controle Remoto:

- Desenvolver funcionalidades que permitam aos usuários controlar os ar-condicionados remotamente (ligar/desligar, ajustar temperatura e modo).

Segurança e Autenticação:

- Implementar um sistema de autenticação de usuários no Firebase, garantindo que apenas usuários autorizados possam acessar e controlar os dispositivos.
- Proteger os dados dos usuários, assegurando que informações sensíveis sejam armazenadas e transmitidas de forma segura.

1.2 Justificativa

O desenvolvimento deste aplicativo para controle de ar-condicionados via IoT justifica-se pela crescente demanda por eficiência energética e conforto em ambientes residenciais e comerciais. A automação desses dispositivos não apenas contribui para a redução do consumo de energia, mas também oferece conveniência ao permitir o controle remoto.

1.3 Metodologia

Inicialmente foi realizado uma revisão bibliográfica baseada em documentações, artigos e vários outros meios de informação que sejam confiáveis. Enquanto isso, foi desenvolvido o figma e logo após a construção de um banco de dados que é implementado na plataforma Google Firebase, alimentando o dispositivo com os dados capturados. Após isso, foi desenvolvido o aplicativo, o qual foi desenvolvido por meio da IDE visual studio code.

1.4 Resultados Esperados

Espera-se com este trabalho que os resultados obtidos contribuam para estudos de IoT, para o desenvolvimento, crescimento e aprimoramento das soluções de tecnologia voltadas para a área comercial, especialmente as que envolvem smart home, com ênfase em automação comercial junto a aplicativos móveis.

1.5 Estrutura do documento

O trabalho está organizado em cinco capítulos, incluindo este introdutório, conforme descrito a seguir.

O **Capítulo 2** Fundamentação Teórica, expondo os conceitos teóricos para a contextualização e entendimento do trabalho desenvolvido.

O **Capítulo 3** aborda os materiais e metodologias utilizadas para o desenvolvimento do trabalho bem como também a integração das partes.

No **Capítulo 4** são apresentados os resultados dos testes realizados.

O **Capítulo 5** apresenta as conclusões do trabalho, expondo os resultados alcançados, assim como uma análise dos pontos falhos e propostas futuras visando complementar as atividades desenvolvidas.

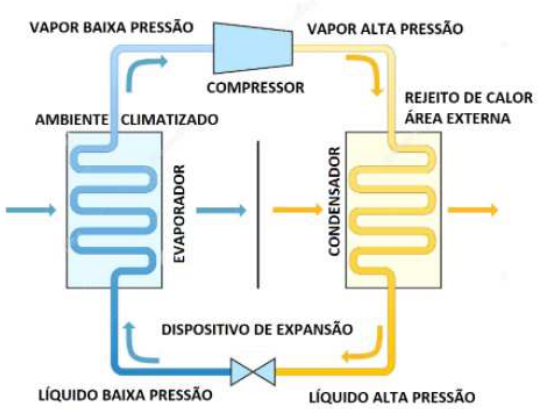
2 Referencial Teórico

Este capítulo aborda os principais fundamentos e conceitos inerentes ao protótipo desenvolvido, com o intuito de fornecer um maior entendimento ao assuntos tratados nos capítulos posteriores. Esta seção apresenta os conceitos de diodos emissores de luz infravermelha, diodos emissores de luz infravermelha, que representam um dos métodos de comunicação utilizados nos controles remotos dos aparelhos de ar-condicionado. Além disso, será apresentada uma visão geral do desenvolvimento do aplicativo em React Native, que permite o controle do dispositivo IoT. O backend será estruturado utilizando o Firebase, especificamente com o banco de dados em tempo real, que facilita a troca de informações e o monitoramento eficaz dos ar-condicionados.

2.1 Principio de Funcionamento

A principal função do ar-condicionado é proporcionar ambientes com temperaturas agradáveis, onde um ser humano sinta conforto térmico, independentemente de ser para aquecer ou resfriar. O condicionamento de ar refere-se ao processo de resfriar e desumidificar o ar em ambientes internos para garantir o conforto térmico ou atender a outras necessidades. Sendo possível regular a temperatura, a umidade e a qualidade do ar, além de oferecer funções como aquecimento, resfriamento, umidificação e desumidificação.(ARCHTOOLBOX, 2021).

Figura 1 – Ilustração de componentes de um Ar-condicionado.



Fonte: Adaptado de (SMACNA, 2024)

O ar-condicionado funciona sobre o princípio da troca térmica do ambiente. O ar passa pela serpentina do evaporador, onde é realizada a troca de calor, elevando ou

diminuindo a sua temperatura, de acordo com o ciclo em funcionamento, gerando, ainda uma redução da umidade relativa do ar. Quando o devido indicador de temperatura é atingido, o compressor do evaporador lê essa informação, desligando em seguida para que a temperatura seja mantida. A qualquer instante, na condição de ocorridos qualquer desvio para fora da faixa de atuação da temperatura encomendada, o compressor do evaporador é desligado. O compressor do evaporador interno é o responsável pela circulação do gás refrigerante.

2.2 Diodo emissor de infravermelho

Para entender o controle de um ar-condicionado, é fundamental conhecer o funcionamento do controle remoto. Ele contém um microcontrolador que armazena os códigos necessários para operar o aparelho, além de componentes eletrônicos e um diodo emissor de infravermelho, que envia os comandos.

A tecnologia dominante nos controles remotos de televisores é o infravermelho (IR). A luz infravermelha é também conhecida como "calor". A premissa básica no funcionamento de um controle remoto IR é o uso da luz para levar sinais entre um controle remoto e o aparelho a que ele controla. A luz infravermelha está na faixa invisível do espectro eletromagnético.

Figura 2 – Espectro Eletromagnético

Fonte: (LAYTON, 2021)

Um controle remoto IR (o transmissor) envia pulsos de luz infravermelha que representam códigos binários específicos. Estes códigos binários correspondem a comandos, como: liga/desliga e aumentar o volume. O receptor IR na TV, ou outro aparelho, decodifica os pulsos de luz em dados binários (1 e 0) que o microprocessador do aparelho pode entender. O microprocessador realiza então a tarefa correspondente.(LAYTON, 2021)

A comunicação via infravermelho é uma das formas mais simples de comunicação sem fio. É uma opção econômica para transmitir dados e oferece uma boa relação sinal-ruído, tornando a transmissão confiável e resistente a interferências, com baixo consumo de energia. Essa tecnologia é comumente utilizada em eletrodomésticos, além de ter aplicações em áreas como controle industrial e segurança. (LV et al., 2010).

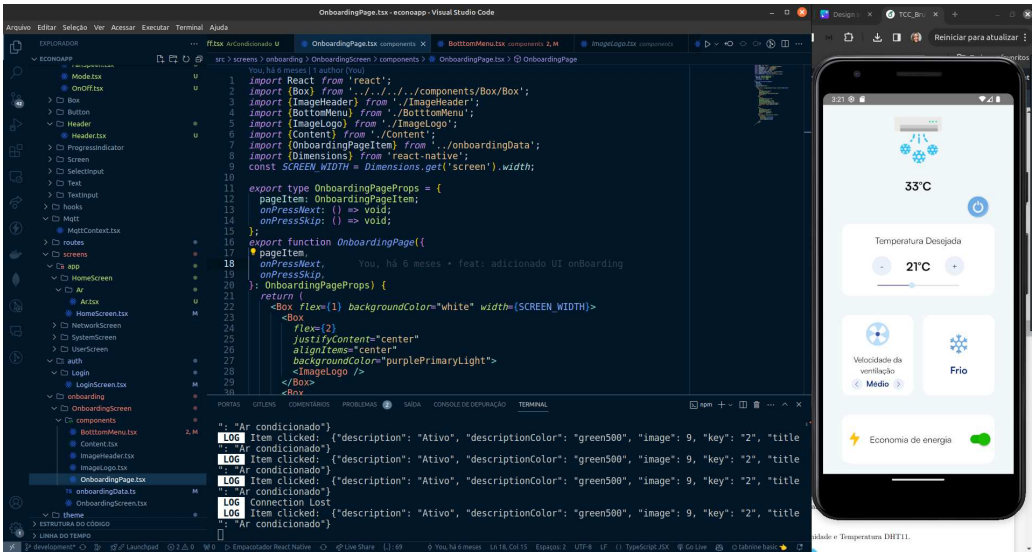
Os LEDs infravermelhos emitem luz invisível ao olho humano, geralmente entre 800 e 980 nanômetros, mas outras fontes, como lâmpadas e o sol, também produzem radiação infravermelha, complicando as comunicações (KITABAYASHI et al., 2010). Para reduzir esses problemas, os LEDs são acionados rapidamente em uma frequência de 38.000 Hz, chamada de frequência portadora. Contudo, cada fabricante utiliza protocolos de comunicação diferentes, tornando necessário que um controle remoto seja específico para cada dispositivo. Fabricantes como PHILIPS, NEC e Toshiba fornecem chips de codificação que organizam os dados para garantir que ambas as partes compreendam a mensagem (LV et al., 2010).

2.3 React Native

O React Native é um framework criado em 2015 pelo Facebook, que permite criar aplicações mobile nativas usando Javascript e utilizar as mesmas API's (Application Programming Interface) da plataforma nativa que outras aplicações utilizam. Atualmente, várias aplicações são desenvolvidas ou estão sendo migradas para o React Native, tais como: o próprio Facebook, Skype, Walmart, Instagram, entre outras. (ALURA, 2024).

O React traz um conceito de componente ao qual é possível fazer uma modularidade da interface, podendo se setar comportamento e atributo de forma 20 especifica de um componente para o outro que pode ser utilizado em vários arquivos e que mudará dependendo da plataforma, fácil de se entender, diminui a quantidade de código escrita e também possibilita que seja mais rápido o desenvolvimento de aplicações.

Figura 3 – Código react native



Fonte:Autoria Própria

Antes do React Native chegar em seu auge, o desenvolvimento para Android e iOS era bem mais complexo. Havia a necessidade de aprender várias linguagens de programação como Java, Kotlin, Objective-C, Swift. Entretanto, isso não garante reaproveitamento de código de uma plataforma para outra. Com isso, em cenários empresariais, havia a necessidade de contratação de um time de desenvolvimento para cada sistema operacional. Consequentemente o desenvolvimento do projeto acabava ficando mais caro e em raros casos mais lento. Porém, com o React Native, o código pode ser reaproveitado em até 100necessárias duas equipes de desenvolvimento, apenas uma.

2.3.1 Ferramentas de Desenvolvimento: Expo e React Native CLI

Expo: Acelerando o Desenvolvimento O Expo é uma ferramenta que simplifica o desenvolvimento no React Native, oferecendo um ambiente integrado que elimina a necessidade de configurações nativas complexas. Com o Expo, desenvolvedores podem testar rapidamente novas ideias e compartilhar seus aplicativos com facilidade, acelerando o ciclo de desenvolvimento. A plataforma também fornece acesso a uma vasta biblioteca de APIs e componentes, facilitando a incorporação de funcionalidades avançadas nos aplicativos.

React Native CLI: Controle Total sobre o Desenvolvimento Para aqueles que necessitam de uma configuração mais personalizada e controle direto sobre o ambiente de desenvolvimento, a React Native CLI é a ferramenta indicada. Ela permite aos desenvolvedores configurar seus projetos de maneira detalhada, acessando diretamente as configurações nativas e construindo funcionalidades complexas sem as limitações do Expo.

Embora exija um conhecimento mais aprofundado das plataformas nativas, a React Native CLI é poderosa para projetos que demandam uma integração nativa mais intensa.

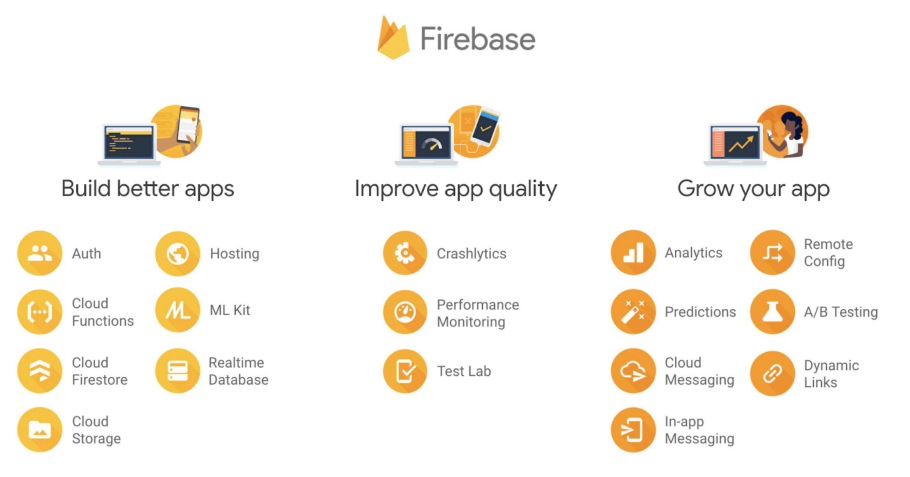
2.4 Firebase

O Firebase é um serviço disponibilizado pela Google em que toda a estrutura do backend está completamente pronta para se integrar com uma aplicação web ou mobile, devido a grande quantidade de serviços oferecidos e a facilidade de implementação. Para pequenos projetos ele pode ser a melhor opção. (DECHALERT, 2019)

Desse modo, o Firebase é mais conhecido como sendo um Realtime Database (Banco de Dados em Tempo real), o que significa que pode ser possível criar aplicações avançadas ao permitir o acesso do banco de dados executando ao lado do código do cliente. Os dados salvos no banco de dados são disponibilizados offline e localmente. Mesmo os dados sendo mantidos assim, os eventos em tempo real podem continuar fazendo requisições, conduzindo uma melhor experiência para o cliente final. Quando a conexão for estabelecida no servidor, o banco de dados em tempo real faz a sincronização das alterações feitas quando estava off-line, e recupera os dados on-line, permitindo as atualizações remotas com êxito. (FIREBASE, 2020)

Com a intenção de resolver os desafios enfrentados por desenvolvedores no início de um projeto, a Google disponibiliza vários serviços através do Firebase (GOOGLE, 2019), que incluem infraestrutura, armazenamento, segurança, autenticação, entre outros. Neste projeto, foram utilizados os serviços de armazenamento de dados em tempo real e autenticação.

Figura 4 – serviços firebase



Fonte: (FIREBASE, 2020).

2.4.1 Armazenamento de dados em tempo real

O maior benefício do Firebase Database Realtime é que ele já possui um sistema 25 de sincronização instantânea implementado, fazendo com que, caso ocorra uma modificação no banco, todos os aplicativos que tenham a referência daquele item, o atualizem automaticamente, em vez de trabalhar com requisição e resposta normalmente utilizado em outros bancos.

2.4.2 Bancos de dados NoSQL

Um banco de dados não relacional é um banco de dados que não usa o esquema de tabela de linhas e colunas encontrado na maioria dos sistemas de banco de dados tradicionais. Em vez disso, os bancos de dados não relacionais usam um modelo de armazenamento otimizado para os requisitos específicos do tipo de dados que está sendo armazenado. Por exemplo, os dados podem ser armazenados como pares chave/valor simples, como documentos JSON13 ou como um gráfico que consiste em bordas e vértices.

2.4.3 Firebase Auth

O Firebase Auth é um serviço que oferece diversos métodos de autenticação: email/senha, uso direto de sistema de contas próprio, ou provedores de terceiros, como o Google, Facebook, Microsof, Twitter, dentre outros. Os recursos deste serviço são gratuitos, exceto a autenticação por telefone. A figura 2.14 mostra o logotipo do Firebase.

3 Metodologia do Projeto

O sistema desenvolvido tem como objetivo facilitar o controle de ar-condicionados, permitindo que o usuário tenha total autonomia na programação e operação. Isso garante um conforto maior, pois ele pode ajustar as configurações conforme suas necessidades. O controle é feito por meio de um emissor infravermelho, que se comunica diretamente com o ar-condicionado, e a interação do usuário acontece através do um aplicativo desenvolvido conectado ao microcontrolador que salva os dados no banco de dados real time do Google Firebase.

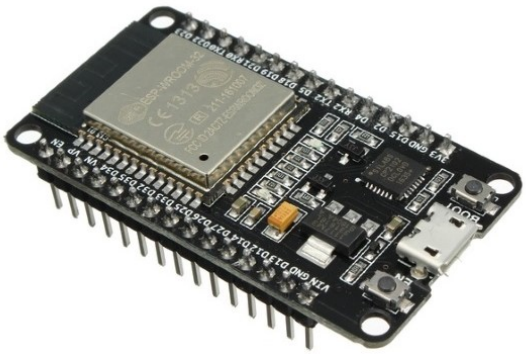
A seguir as ferramentas e as etapas realizadas para o desenvolvimento do protótipo do aplicativo de monitoramento, automação e controle de ar-condicionado.

3.1 Componentes de Hardware

3.1.1 ESP-32

ESP32 é um módulo microcontrolador (MCU) WiFi, Bluetooth, Bluetooth Low Energy (BLE), de ampla variedade de aplicações, desde de redes de sensores de baixa potência a tarefas mais exigentes, como codificação de voz, streaming de música e decodificação de MP3 (ESPRESSIF, 2012), desenvolvido pela empresa *Espressif Systems*, com módulos Wi-Fi e Bluetooth integrado, foi desenvolvido com o objetivo ser utilizado em aplicações do âmbito da internet das coisas, e é uma placa de desenvolvimento considerada de baixo custo.

Figura 5 – Representação da Placa de Desenvolvimento ESP32.



Fonte: (MAKERHERO, 2017)

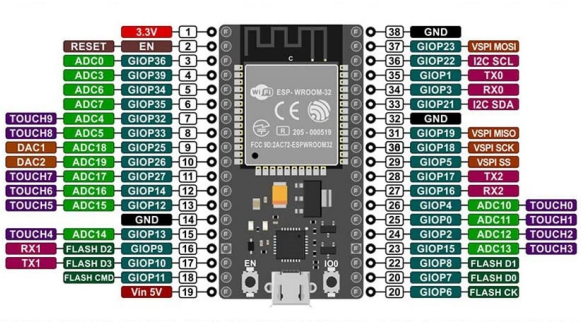
O ESP32 pode ser programado em diversas IDEs, como a SDK da Espressif, MicroPython, Visual Studio, Arduino IDE e outros editores. Programas ESP32 são escritos

em linguagem C

As principais características do ESP32 são:

- Microprocessador Xtensa® single/dual-core LX6 de 32 bits
- Processador pode operar em 80MHz / 160MHz / 240MHz
- Tensão operacional de 3,3(V)
- Alimentação fornecida através do conector Micro USB-B integrado ou diretamente com 5 à 12 VDC através do pino “VIN”
- 25 pinos de Entrada/Saída digitais (DIO)
- 6 pinos de entrada analógica (ADC)
- 2 pinos de saída analógica (DAC)
- 4MB de memória flash
- WiFi nativo padrão 802.11b/g/n
- Bluetooth BLE 4.2 BR/EDR e BLE (Bluetooth Low Energy)

Figura 6 – Pinagem placa ESP32.



Fonte: (MAKERHERO, 2017)

3.1.2 LED emissor de infravermelho

O LED emissor atua como um componente essencial no sistema, transmitindo comandos para o receptor infravermelho do ar-condicionado. Esse dispositivo converte sinais elétricos em luz visível, permitindo que o usuário execute as funções desejadas. Para essa aplicação, foi utilizado um LED Emissor Infravermelho de 5mm como mostrado na Figura 7.

Figura 7 – LED Infravermelho.

Fonte: (MAKERHERO, 2024)

3.1.3 Módulo receptor infravermelho 1838T

O sensor 1838T, apresentado na Figura 8, é um componente eletrônico utilizado para receber e decodificar sinais infravermelhos emitidos por controles remotos Pode ser encontrado em TVs, rádios, aparelhos de multimídia e até em aparelhos de ar condicionado. O módulo contém um receptor IR 1838 que conta com tensão de operação entre 2,7 e 5,5V. A comunicação com o microcontrolador se dá por meio de um único pino de saída.

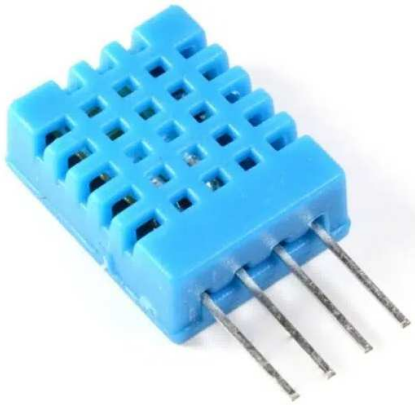
Figura 8 – Receptor Infravermelho 1838T.

Fonte: (RYTRONICS, 2024)

3.1.4 Modulo Sensor de Temperatura e Umidade DHT11

O sensor DHT11 é um dispositivo amplamente utilizado para monitorar a temperatura e a umidade do ar. Ele é composto por um sensor de umidade capacitivo e um termistor NTC (coeficiente de temperatura negativo). Esses elementos estão integrados em um único módulo, que também inclui um circuito integrado responsável por realizar as medições e fornecer os dados em formato digital. O DHT11 possui três pinos de conexão: VCC, DATA e GND.

Figura 9 – Sensor de Umidade e Temperatura DHT11.



Fonte: MAKERHERO, 2023.

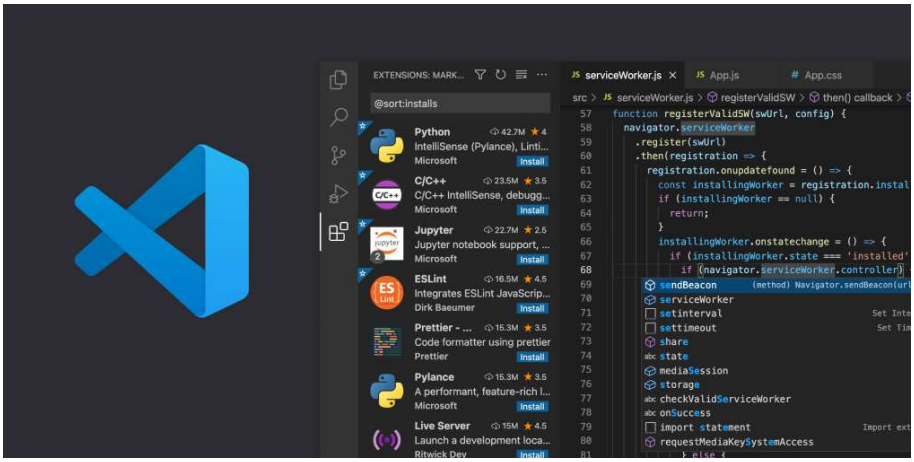
Apesar de sua popularidade e baixo custo, o sensor DHT11 possui algumas limitações. A principal delas é a sua precisão nas leituras da temperatura com uma precisão de ±2°C e a leitura da umidade com uma precisão de ±5%. Além disso, a taxa de atualização do sensor é relativamente baixa, o que significa que ele não é adequado para aplicações que exigem uma resposta em tempo real.

3.2 Desenvolvimento de Software

3.2.1 Visual Studio Code

Visual Studio Code é um editor de código-fonte leve e intuitivo para desenvolvimento, estando disponível para os sistemas operacionais macOS, Linux e Windows. Com suporte já integrado para a linguagem JavaScript. (CODE, 2024) Ele será utilizado para desenvolvimento do software, sendo utilizado hibridamente com as linguagens HTML, CSS e JS.

Figura 10 – visual studio code



Fonte: (CODE, 2024)

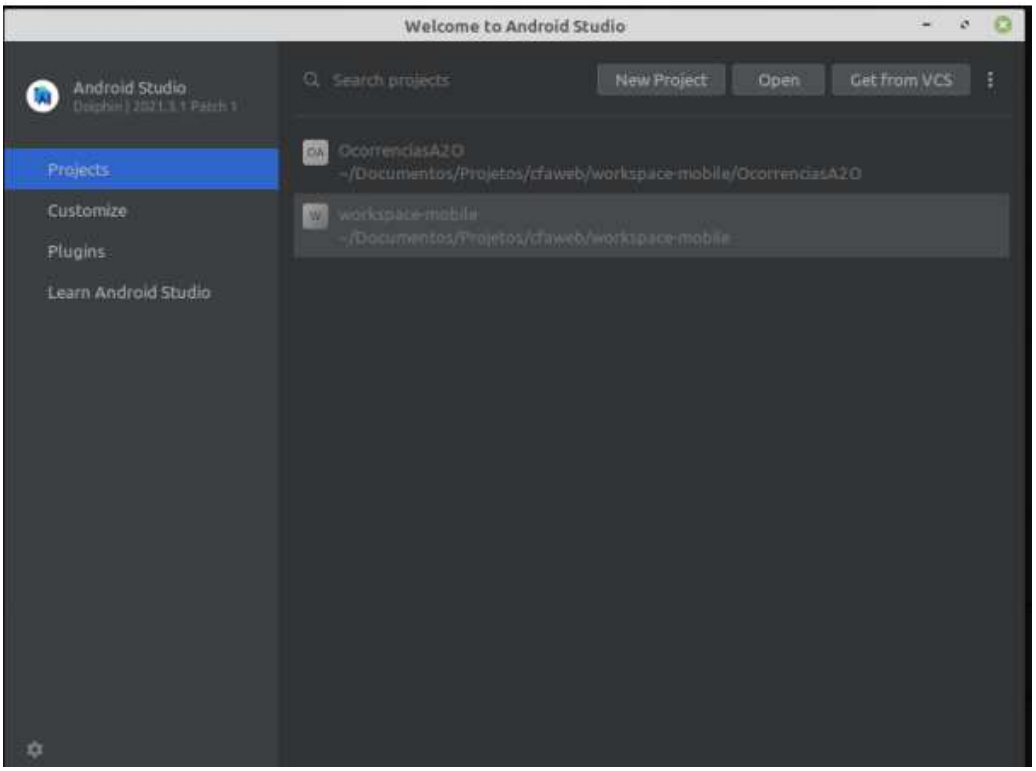
3.2.2 Android Studio

O Android Studio é uma IDE que foi criada pela Google. Esse ambiente de desenvolvimento possui várias funcionalidades, tais como preenchimento automático de comandos, auto indentação, navegador de pasta, simulação de dispositivo, dentre outras funcionalidades 23.

O Android Studio é uma plataforma de desenvolvimento poderosa e versátil, projetada para facilitar a criação de aplicativos Android. Ele é o ambiente integrado oficial para esse tipo de desenvolvimento, baseado na IDE IntelliJ IDEA. O Android Studio oferece uma série de recursos que aumentam a produtividade, como:

- Um emulador rápido com diversas funcionalidades.
- Um ambiente unificado que permite desenvolver para todos os dispositivos Android.
- A capacidade de aplicar mudanças de código diretamente em um aplicativo em execução, sem precisar reiniciá-lo.
- Ferramentas e frameworks de teste abrangentes.

Figura 11 – Android Studio



Fonte: Autoria Própria

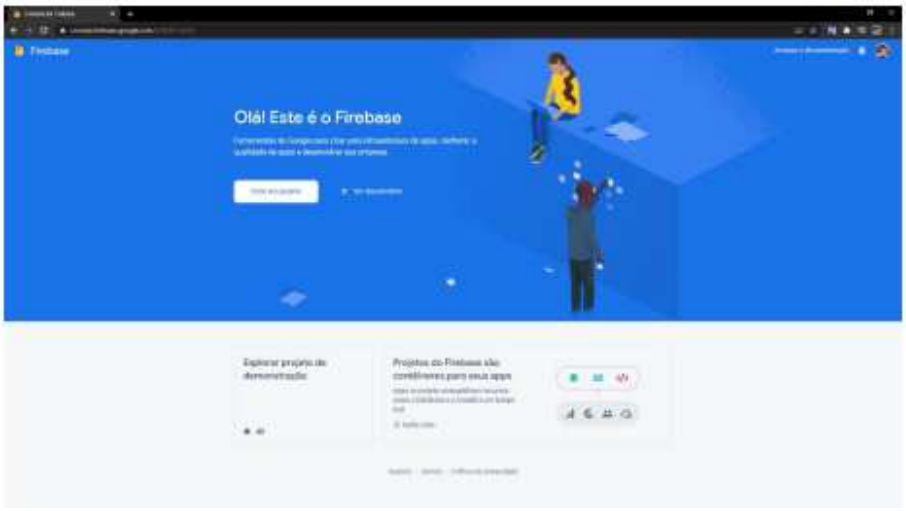
3.2.3 Google Firebase

O Firebase será utilizado como plataforma de armazenamento de dados em nuvem, oferecendo flexibilidade para softwares híbridos. A conectividade será estabelecida entre dispositivos físicos em rede e uma interface web, funcionando como uma ponte para garantir o correto funcionamento do sistema. O banco de dados suporta seis linguagens de programação: Swift, Objective-C, Java, JavaScript, C++ e Unity (FIREBASE, 2020).), sendo que este projeto utiliza JavaScript para a integração com o Firebase Realtime Database.

As funcionalidades do Firebase que serão empregadas neste projeto incluem:

- **Auth** – O Firebase Auth é um serviço que oferece diversos métodos de autenticação;
- **Cloud Firestore** – banco de dados NoSQL que armazena dados no formato JavaScript Object Notation (JSON);
- **Realtime** – permite a gravação de dados no banco em tempo real.

Figura 12 – Tela Inicial para criar um projeto no Firebase



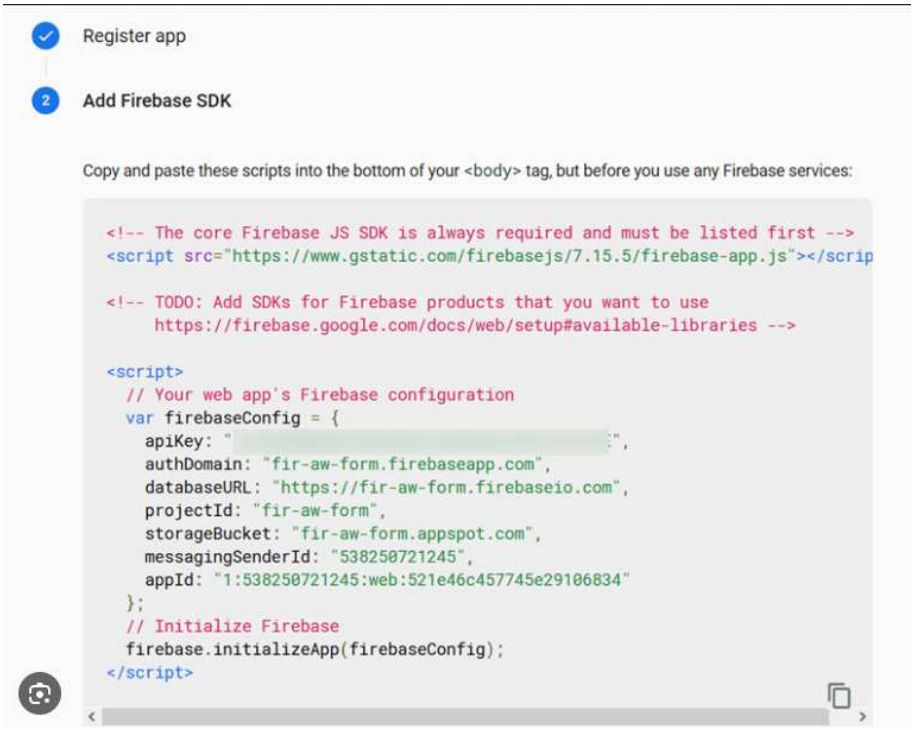
Fonte: Autoria Própria

Figura 13 – Tela para definir o nome do projeto Firebase

Fonte: Autoria Própria

Para o uso do Cloud Firestore é necessário a inicialização dele na página inicial depois da criação do projeto,será criada a coleção posteriormente, para salvar os dados dos usuários como nome, e-mail e uma identificação gerada pelo Cloud Firestore na coleção users (UID).

Figura 14 – dados para conexão o app



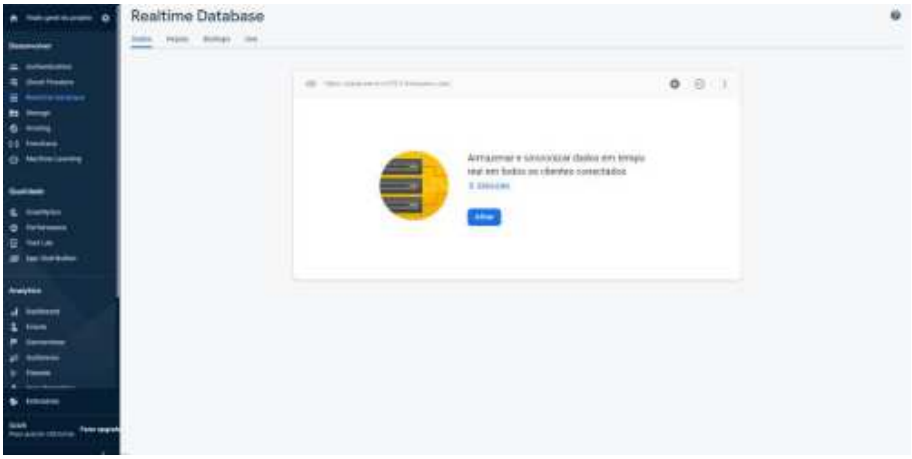
Fonte: Autoria Própria

Figura 15 – Tela do Cloud Firestore do Firebase

Fonte: Autoria Própria

Ativação também do Realtime Database, que é necessário para obter as gravações em tempo real dos dados dos sensores para controle do ar.

Figura 16 – Tela Realtime Database do Firebase



Fonte: Autoria Própria

3.2.4 Cloud Firestore

Na aplicação desenvolvida, o uso do Cloud Firestore é utilizado para o usuário se cadastrar na aplicação e consequentemente realizar um login com seus dados cadastrados, na tela de cadastro é solicitado o nome, e-mail e senha para poder acessar o app.

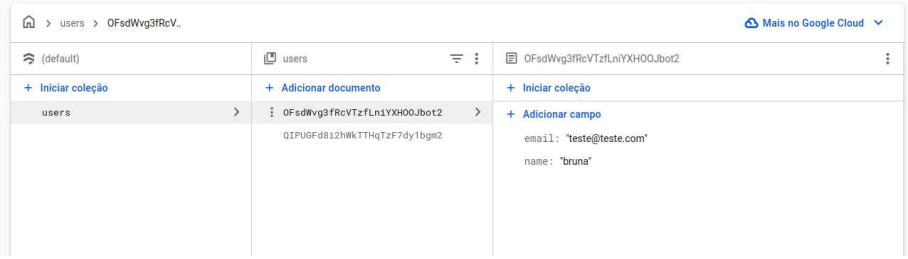
Figura 17 – Tela de cadastro do usuário

Fonte: Autoria Própria

Após o usuário realizar o cadastro, os dados são salvos no documento com um ID

gerado para coleção users criada para salvar os usuários cadastrados.

Figura 18 – Dados do usuário cadastrado



Fonte: Autoria Própria

Com os dados salvos, o usuário poderá realizar o login com seu e-mail e senha cadastrados, a tela de login.

Figura 19 – Tela de login

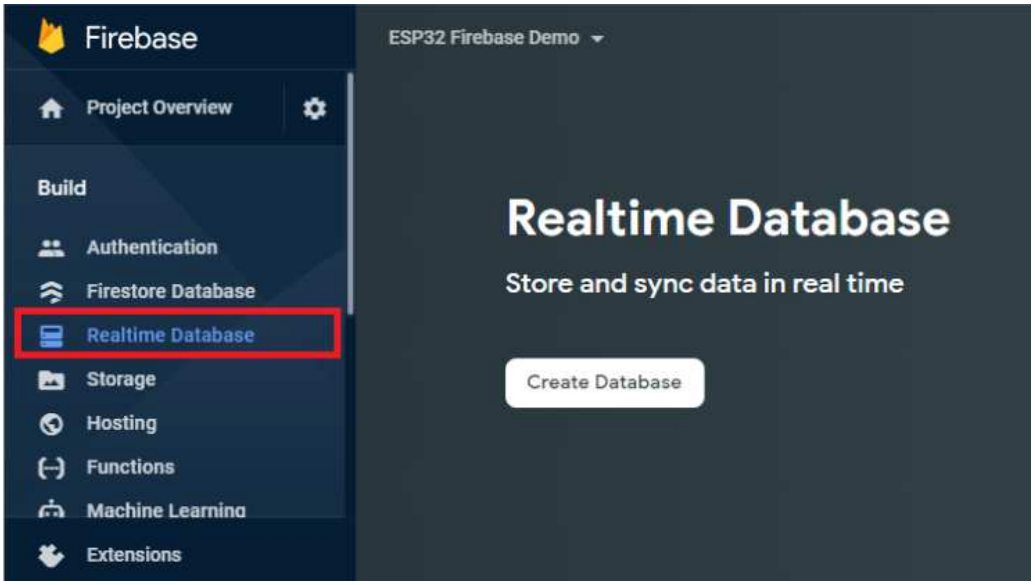
Fonte: Autoria Própria

3.2.5 Realtime Database

Na aplicação desenvolvida, o Realtime Database foi utilizado para salvar os dados de temperatura ambiente enviados pelo sensor de temperatura DHT11,assim como para salvar os dados do infra-vermelho para podermos controlar o ar-condicionado.

- Criação e configuração:

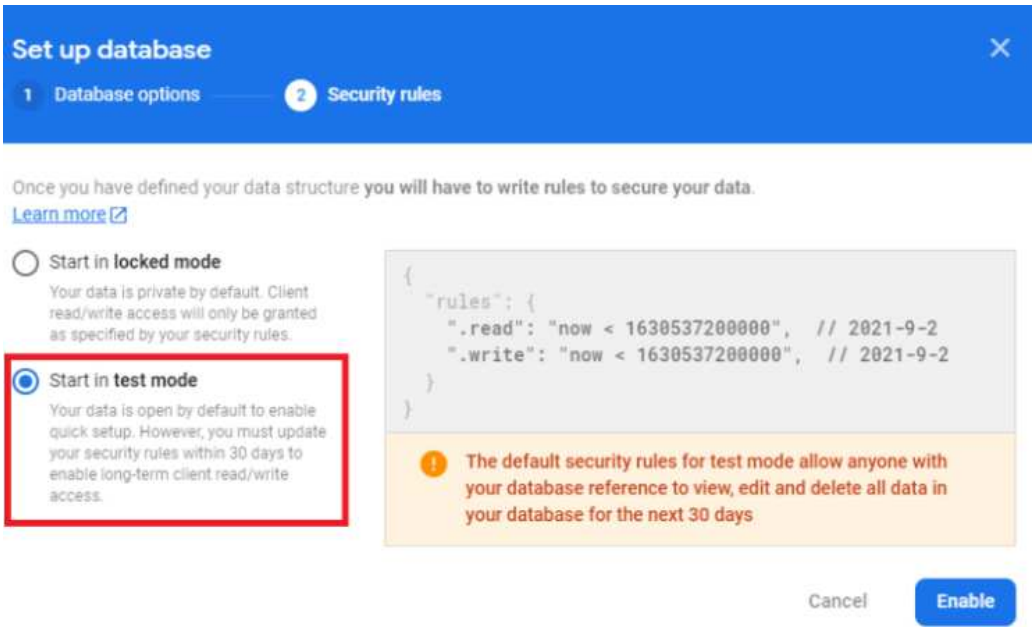
Figura 20 – Criação do banco de dados em tempo real



Fonte: Autoria Própria

Configuração de regras de segurança para seu banco de dados. Para fins de teste, selecione Iniciar no modo de teste. Em tutoriais posteriores você aprenderá como proteger seu banco de dados usando regras de banco de dados.

Figura 21 – Criação



Fonte: Autoria Própria

Seu banco de dados agora está criado. Você precisa copiar e salvar o URL do banco de dados – destacado na imagem a seguir – porque você precisará dele posteriormente

em seu código ESP32.

Figura 22 – Criação

Fonte: Autoria Própria

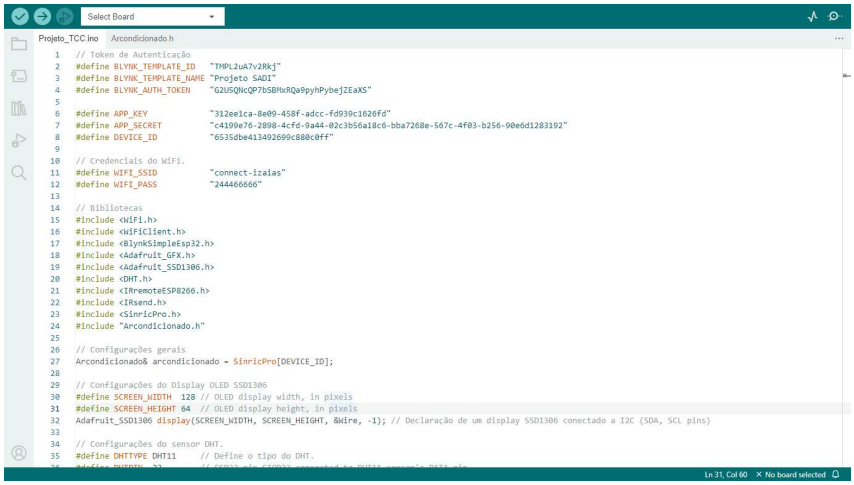
Os dados são salvos como uma árvore JSON, no primeiro nó da árvore geofence, tem o UID para a identificação de para qual usuário o aparelho foi cadastrado.

3.2.6 Arduino IDE

O Arduino utiliza uma interface Integrated Development Environment (IDE) de forma intuitiva e simplificada para melhorar a experiência do usuário. A placa ESP32 pode ser programada em várias frameworks diferentes, mais comum sendo a ESP-IDF, framework oficial da *Espressif*, e Arduino IDE, sendo este último em que o microcontrolador foi programado. A interface do Arduino IDE é mostrada na Figura 23.

Além disso, a IDE do Arduino oferece uma vasta biblioteca para diversos tipos de dispositivos, sensores e serviços, incluindo a integração com o firebase, facilitando e simplificando a integração de diferentes sistemas e dispositivos, e assim acelerando o processo de criação e prototipagem de aplicativos IoT.

Figura 23 – Interface do Arduino IDE.

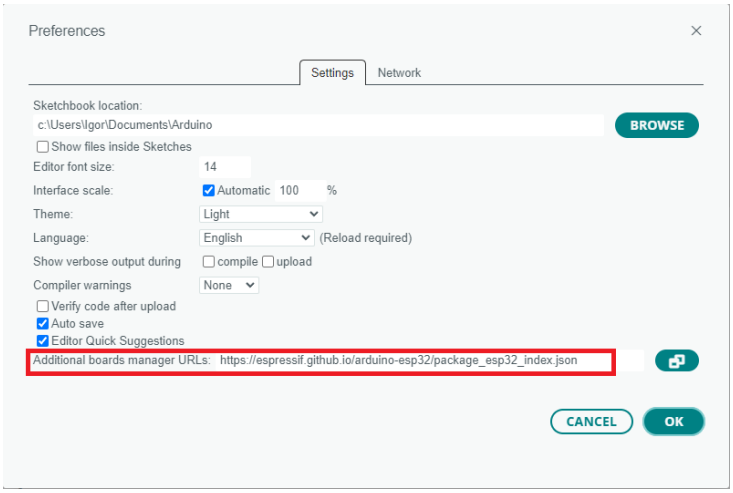


Fonte: Autoria Própria

3.2.7 Implementação na placa de desenvolvimento ESP32

Inicialmente, para implementação na placa de desenvolvimento ESP32, foi realizada a instalação do Arduino IDE, framework da Arduino na qual tem suporte para o desenvolvimento de aplicações nas placas ESP32. Para utilizar dispositivos ESP32 no arduino é necessário primeiro adicionar as placas ao IDE. Para isso clicamos em Arquivos -> Preferências, e na aba Preferências aberta, escrevemos em "URLs adicionais para Gerenciadores de Placas" a URL *"https://espressif.github.io/arduino-esp32/package_esp32_index.json"*.

Figura 24 – Configuração para instalar placas ESP32 na IDE do Arduino.

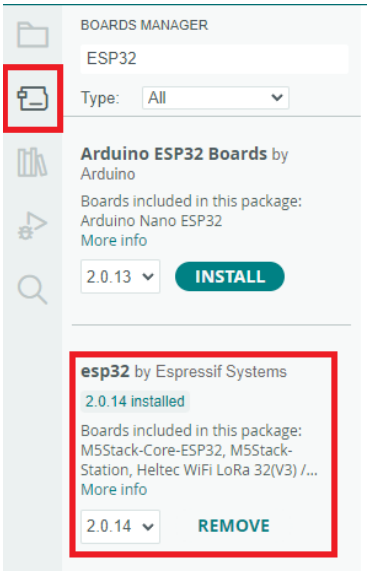


Fonte: Autoria Própria

Após isso vamos em Ferramentas > Placa > Gerenciador de Placas procuramos "ESP32", e a opção de instalar placas de desenvolvimento ESP32 estará disponível. A placa

selecionada deverá ser a mesma utilizada para não acorrer problemas de compatibilidade nas GPIOs.

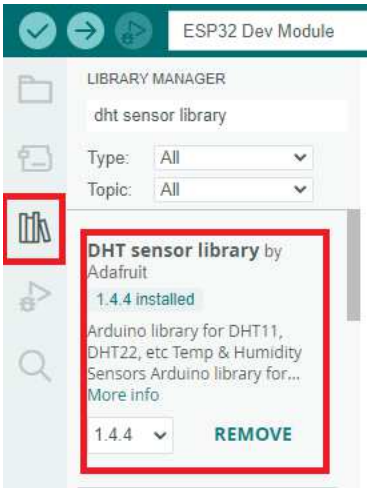
Figura 25 – Configuração para instalar placas ESP32 na IDE do Arduino.



Fonte: Autoria Própria

Com a plataforma instalada, foi iniciado os testes de undiade, onde foram testados todos os componentes individualmente, começando com o ESP32 em conjunto com o sensor de temperatura DHT11. Para isso é necessário a adição das bibliotecas do sensor para que aconteça a comunicação entre os dispositivos. Para adicionar bibliotecas no Arduino IDE, é necessário clicar em "Gerenciador de bibliotecas"e procurar pela biblioteca necessária.

Figura 26 – Configuração para adicionar bibliotecas no Arduino IDE.

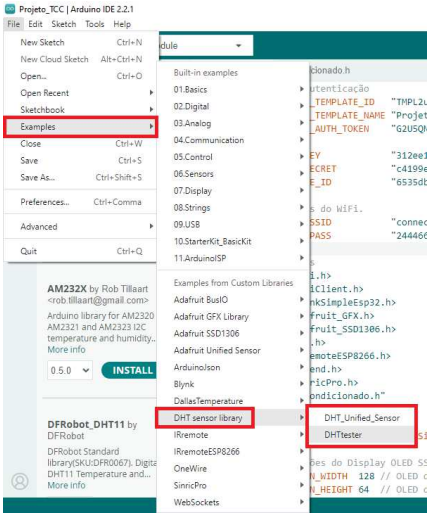


Fonte: Autoria Própria

As bibliotecas instaladas são criadas por usuários, as vezes até por empresas, e

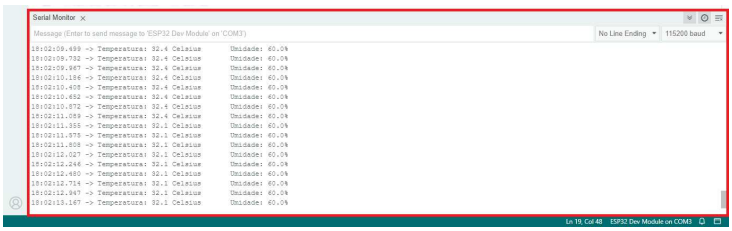
são geralmente mantidas atualizadas em relação as mudanças na IDE do Arduino, ou atualizações nas placas de desenvolvimento. As bibliotecas também oferecem códigos exemplos para as mais diversas funcionalidades. Para o teste do sensor de temperatura foi utilizado o exemplo disponibilizado para testes, como mostrado na figura 27, e os resultados dos testes exibidos no monitor serial são mostrados na figura 28.

Figura 27 – Códigos exemplos disponibilizados pela biblioteca.



Fonte: Autoria Própria

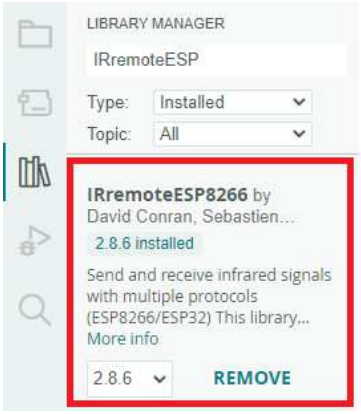
Figura 28 – Resultado do teste de unidade da biblioteca do sensor DHT11 exibidos no monitor serial.



Fonte: Autoria Própria

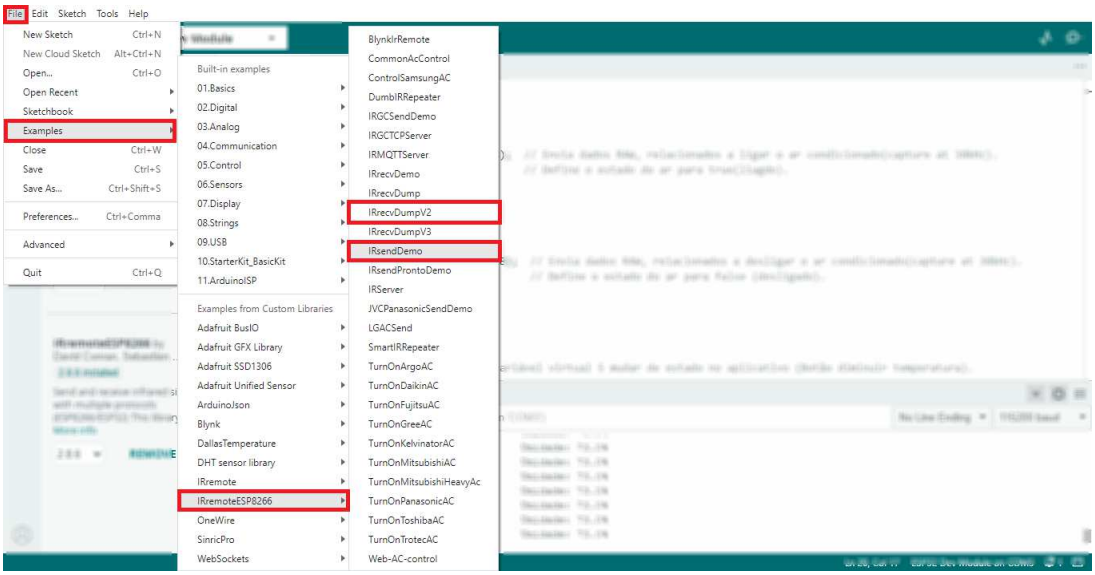
O segundo teste de unidade realizado foi o do módulo receptor infravermelho 1838T e do LED emissor de infravermelho. Para o módulo receptor foi primeiro instalado a biblioteca *IRremoteESP8266*, que fornece suporte para placas de desenvolvimento ESP32. Esta biblioteca permite enviar e receber sinais infravermelhos em um ESP8266 ou ESP32 usando a *framework* do Arduino, com LEDs IR comuns de 940 nm e módulos receptores IR como o 1838T. A biblioteca também conta com exemplos disponíveis com códigos prontos que foram utilizados para os testes.

Figura 29 – Configuração para adicionar a biblioteca *IRremoteESP8266*.



Fonte: Autoria Própria

Figura 30 – Configuração para abrir códigos exemplos da biblioteca.



Fonte: Autoria Própria

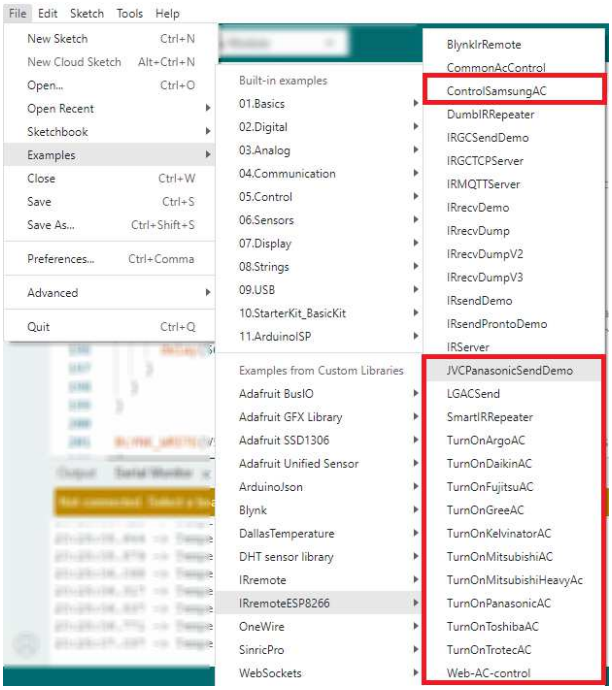
Com a biblioteca instalada, foi possível fazer a captura dos comandos do controle do ar-condicionado,o teste foi feito com o ar do meu quarto e os comandos relativos as temperaturas de 16°C até 32°C, todos em formato NEC. Com os códigos em formato NEC capturados, foi possível realizar o teste do emissor de infravermelho. A biblioteca *IRremoteESP8266* oferece um exemplo de código para envio de dados IR, como mostra a figura 33. Com ela foi possível fazer a clonagem dos sinais do controle do ar-condicionado, fazendo com que o ar-condicionado identificasse o comando emitido.

Essa biblioteca também conta com um acervo de códigos para controle de de ar-condicionados de diversas marcas como Samsung, Toshiba, Panasonic, Sony e outras,

Figura 32 – Enter Caption

onde os códigos e protocolos para cada fabricante já estão configurados para o uso em cada aparelho que trabalhe nessas configurações. A biblioteca é mantida e atualizada por usuários de forma voluntária, e a cada atualização nova, novos códigos são adicionados, aumentando ainda mais a quantidade de fabricantes disponíveis.

Figura 31 – Códigos prontos para controle de Ar-condicionados de várias fabricantes.



Fonte: Autoria Própria

3.2.8 Integração NodeMCU (ESP32) + Firebase Realtime Database

Será necessario adicionar a biblioteca cliente Firebase ESP para este projeto. Esta biblioteca é compatível com placas de desenvolvimento ESP32 e ESP8266. Usaremos o Library Manager no Arduino IDE para instalá-lo com sucesso.

Abra o Arduino IDE e clique em Sketch > Biblioteca > Gerenciar Bibliotecas. Digite 'Firebase ESP Client' na guia de pesquisa e instale a versão mais recente da Firebase Arduino Client Library para ESP8266 e ESP32 mostrada abaixo.

Figura 33 – Biblioteca Google Firebase ESP Client

Fonte: Autoria Própria

Agora a placa ESP32 já esta pronta para interagir com o banco de dados. Sendo necessário apenas inserir suas credenciais de rede, chave API do projeto, URL do banco de dados e o e-mail e senha do usuário autorizado.

-Incluir bibliotecas

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

- Credenciais de Rede

```
#define WIFI_SSID "REPLACE_WITH_YOUR_SSID"
#define WIFI_PASSWORD "REPLACE_WITH_YOUR_PASSWORD"
```

- Chave de API e URL do banco de dados.

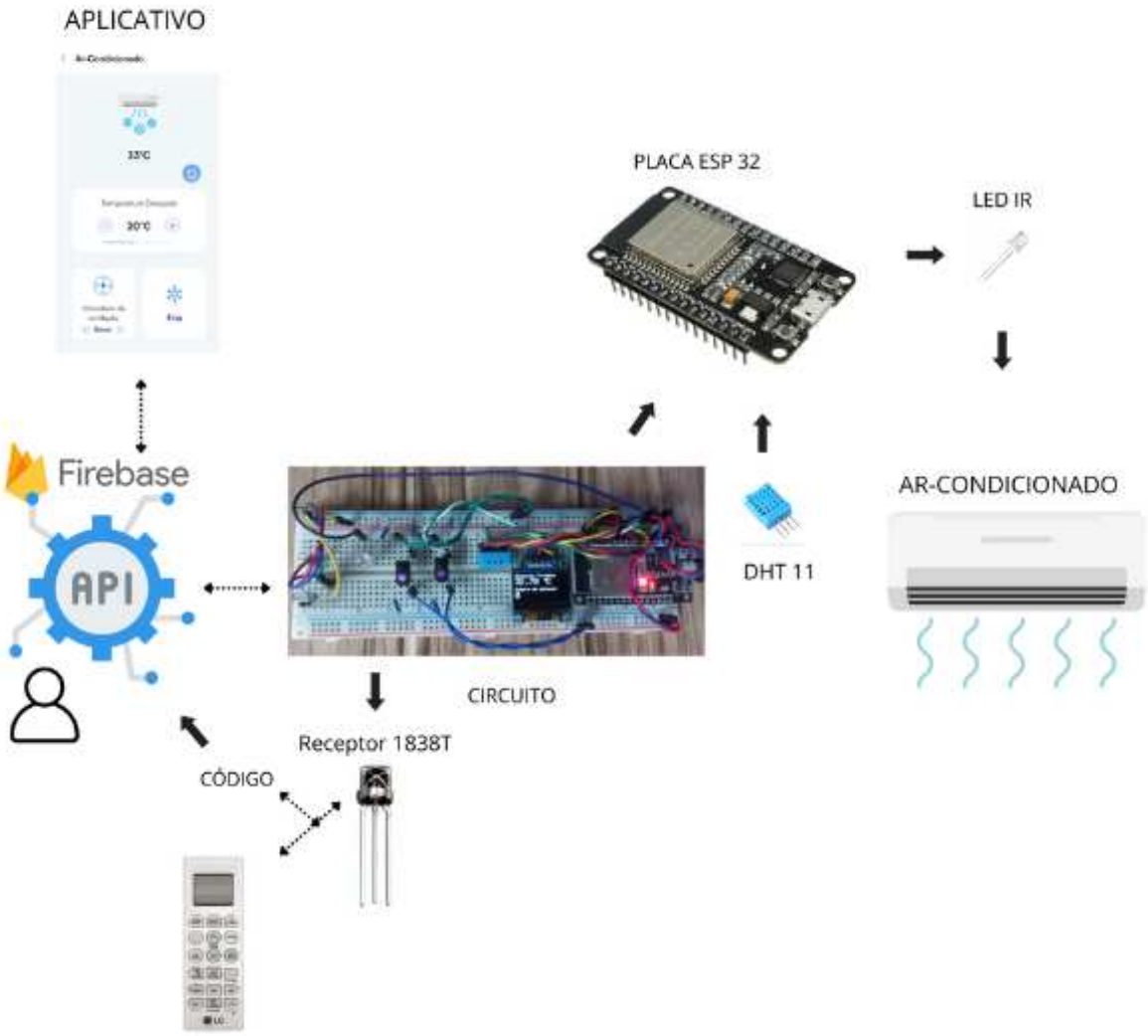
```
#define API_KEY "REPLACE_WITH_YOUR_PROJECT_API_KEY"
#define DATABASE_URL "REPLACE_WITH_YOUR_DATABASE_URL"
```

Dados esses que são adquiridos ao iniciar o projeto no firebase,como mostrado na figura 14.

4 Resultados e discussões

Neste capítulo são apresentados os resultados obtidos no decorrer do trabalho.Os testes realizados de integração hardware + software fora realizados com sucesso. Uma ilustração do projeto desenvolvido é mostrado na Figura 34.

Figura 34 – Ilustração do projeto realizado.



Fonte: Autoria Própria

Como mostrado na Figura 34, a comunicação entre o microcontrolador ESP32 e o ar-condicionado é estabelecida por meio de um módulo receptor infravermelho (IR) 1838T, que recebe os sinais de controle, como ajustes de temperatura e modos de operação, enviados por um LED infravermelho. Esses sinais correspondem aos comandos de um controle remoto tradicional do ar-condicionado.

Cada comando enviado pelo usuário é armazenado em tempo real no banco de dados Firebase. O ESP32 acessa essas informações e utiliza os dados armazenados para controlar o ar-condicionado remotamente. O aplicativo, desenvolvido em React Native, oferece ao usuário uma interface intuitiva, onde é possível visualizar e ajustar as configurações do ar-condicionado de maneira prática, com controle via internet.

4.0.1 Aplicativo EconoWise

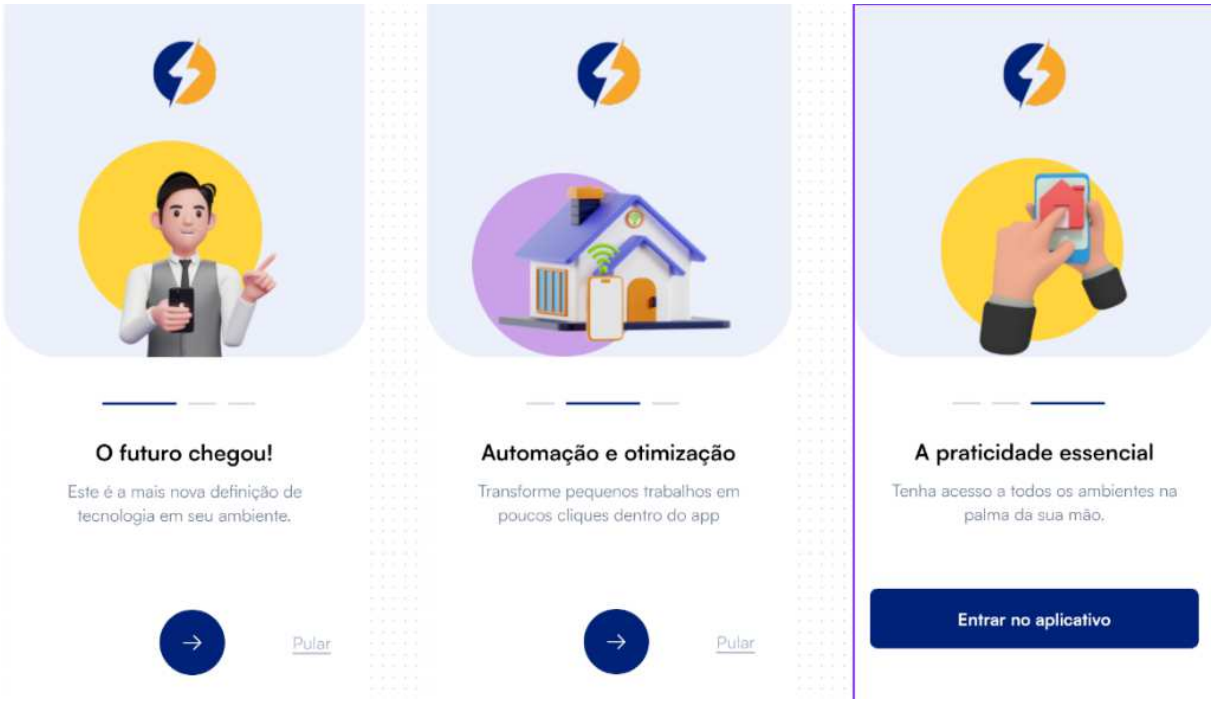
- A seguir estão listados os requisitos funcionais do aplicativo,seguido pelas telas realizadas no figma:
  - **SplashScreen:**
    - Pagina inicial de carregamento

Figura 35 – SplashScreen.

Fonte: Autoria Própria

  - **Página de Onboarding:**
    - Série inicial de telas exibidas quando um usuário inicia um aplicativo ou visita um site pela primeira vez;

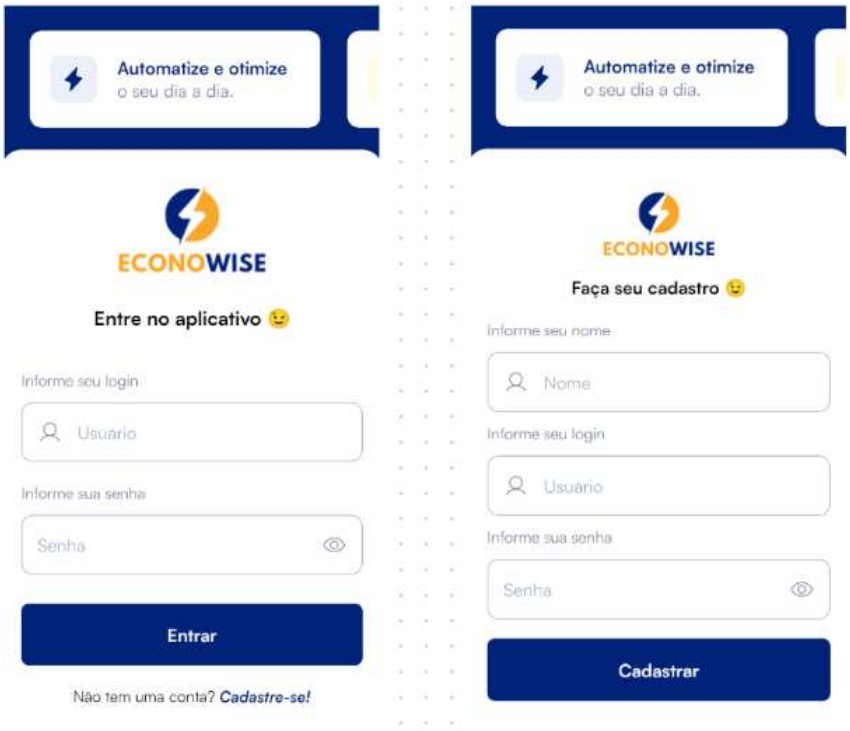
Figura 36 – Telas de Onboarding.



Fonte: Autoria Própria

- **Página de Login:**
  - Login com Email e Senha;
- **Página de Cadastro:**
  - Cadastro com Nome,Email e Senha;

Figura 37 – Telas de Autenticação.

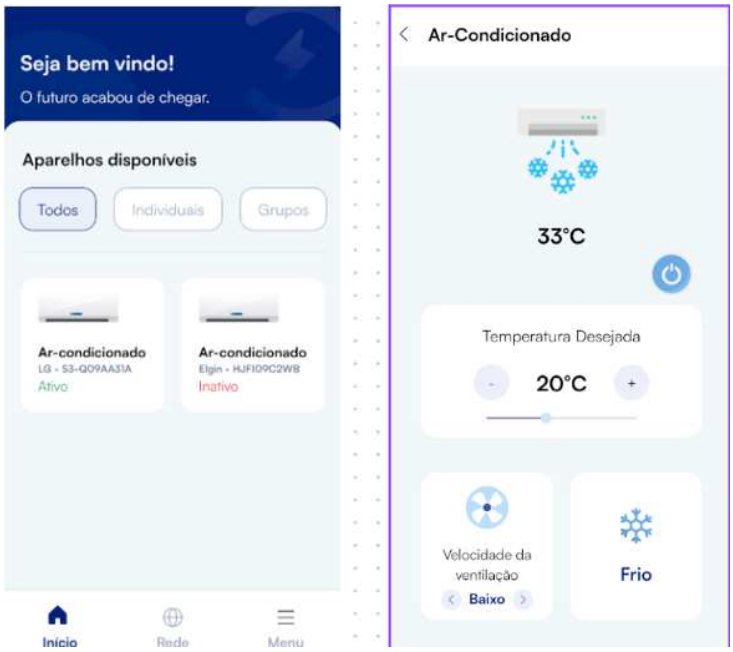


Fonte: Autoria Própria

– **Página Home:**

  * Acesso à página de perfil, configurações e logout;
  * Acesso à página dos Dispositivos;
  * Listagem de dispositivos;

Figura 38 – o protótipo desenvolvido com ESP32.

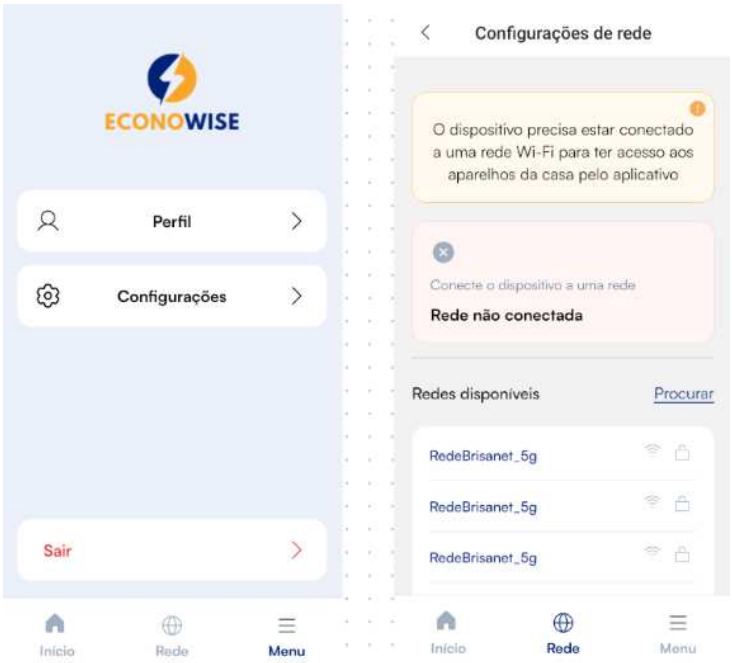


Fonte: Autoria Própria

– **Página de Perfil, Configurações e Logout:**

  * Perfil do usuário;
  * Configurações de rede;
  * Logout do aplicativo;

Figura 39 – Tela Principal.



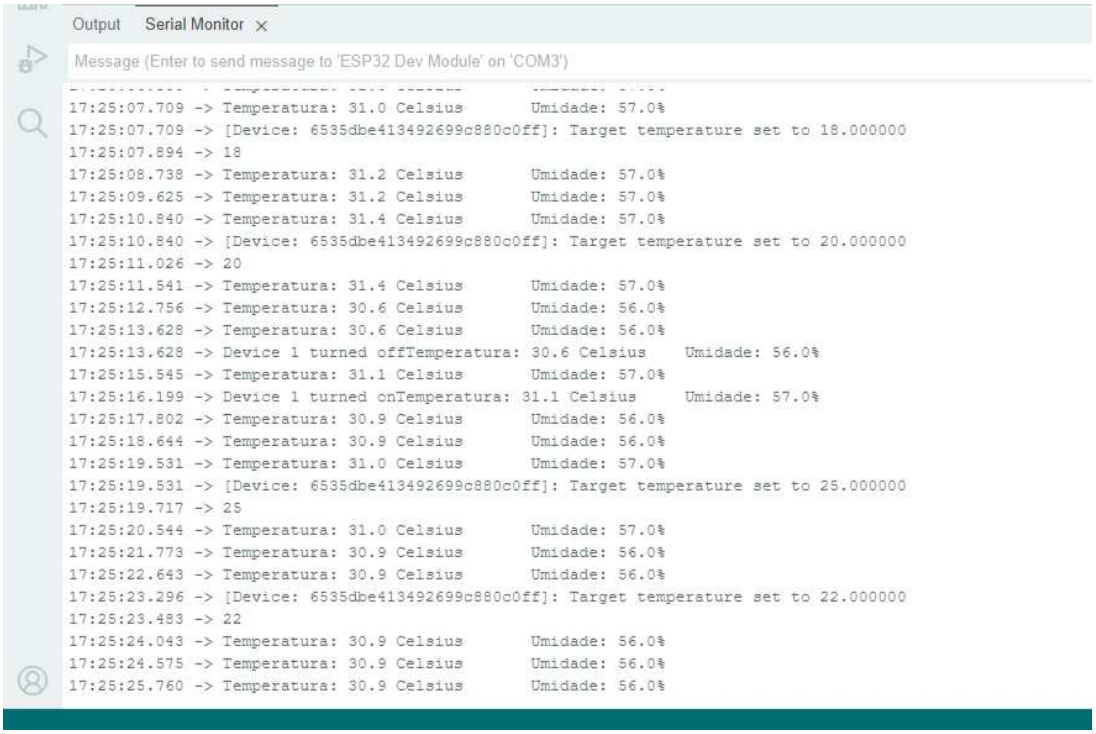
Fonte: Autoria Própria

– O código-fonte do projeto desenvolvido pode ser encontrado:

  * EconoWise - Disponível em `https://github.com/BrunaEduarda03/SmartAir`.

Também pode-se acompanhar as ações realizadas no Monitor Serial do Arduino IDE, que exibem os valores de temperatura e umidade atuais.

Figura 40 – Monitor serial exibindo todas as ações feitas no ESP32.



Fonte: Autoria Própria

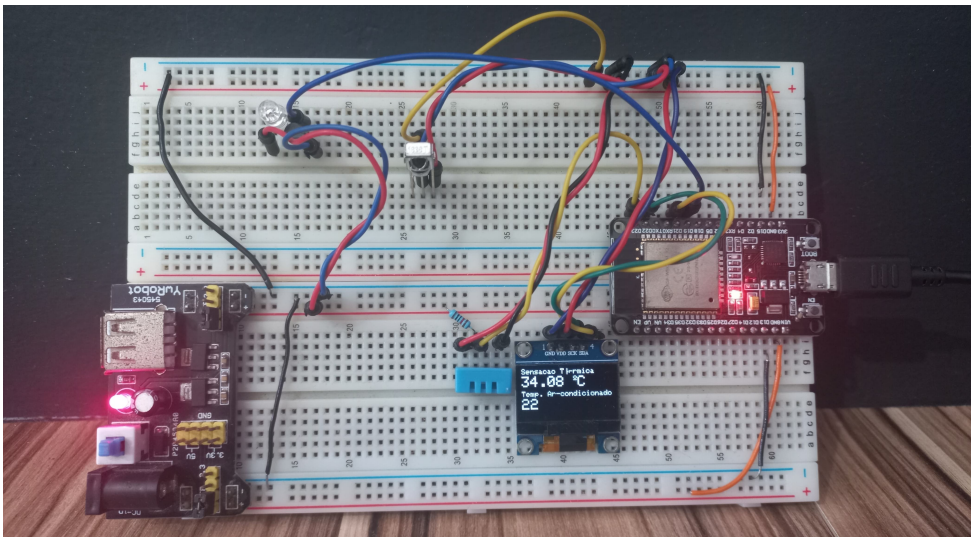
No momento, foram capturados os códigos de controle para o teste do ar-condicionado da marca *LG*. Para ar-condicionados de diferentes marcas e modelos, é necessária adicionar ao banco através da captura dos sinais utilizando a biblioteca *IRremoteESP8266*, assim como mostrado na subseção 3.2.7.

4.0.2 Lista de Materiais

- Microcontrolador ESP32
- Sensor de temperatura DHT11
- Display OLED SSD1306 (Testes Iniciais)
- Módulo Receptor Infravermelho 1838T
- LED emissor infravermelho
- Módulo de Fonte YwRobot MB102
- Resistores
- Jumpers
- Protoboard

4.0.3 Protótipo de Automação

Figura 41 – o protótipo desenvolvido com ESP32.



Fonte: Autoria Própria

4.0.4 Funcionamento do Circuito

O circuito do sistema é composto por vários componentes que trabalham juntos para controlar remotamente o ar-condicionado. O microcontrolador ESP32 é responsável por processar os dados dos sensores e controlar os dispositivos, sendo o "cérebro"do sistema. O sensor DHT11 mede a temperatura e a umidade do ambiente, com os resultados exibidos no display OLED SSD1306, oferecendo feedback visual ao usuário.

O módulo receptor infravermelho 1838T recebe os sinais do controle remoto do ar-condicionado, permitindo que o ESP32 interprete esses sinais e, quando necessário, os armazene para futuras ações. O LED infravermelho, amplificado pelo transistor BC337, transmite os sinais para o ar-condicionado conforme as configurações definidas no aplicativo.

A alimentação do sistema é fornecida pelo módulo de fonte YwRobot MB102, que oferece 3.3V ou 5V, garantindo a estabilidade e flexibilidade necessária para o funcionamento do ESP32 e dos outros componentes.

Esse conjunto de módulos e componentes permite ao usuário controlar o ar-condicionado de maneira eficiente e automatizada, com feedback visual e controle remoto via aplicativo.

O funcionamento do projeto pode ser visto atraves do github em:

https://github.com/BrunaEduarda03/SmartAir.

5 Considerações Finais

Neste capítulo, apresentamos as considerações finais sobre o desenvolvimento deste trabalho de conclusão de curso.

5.0.1 Conclusões

O objetivo deste trabalho foi desenvolver um aplicativo para o monitoramento e controle de sistemas de ar-condicionado. Ao longo do projeto, conseguimos criar um dispositivo de fácil utilização, utilizando o microcontrolador ESP32, que permite a automação do controle via internet, proporcionando eficiência energética e maior conforto aos usuários.

Com este sistema, diversos usuários podem cadastrar e controlar diferentes modelos de ar-condicionados, assim como outros aparelhos que operam por meio de sinais infravermelhos. Para isso, é necessário um circuito equipado com um módulo receptor de infravermelho adequado, capaz de captar e decodificar os respectivos códigos e protocolos.

O projeto também propiciou um estudo prático sobre o React Native, que se destacou como uma excelente ferramenta para desenvolvimento mobile devido à sua natureza multiplataforma. No entanto, o uso do Cloud Firestore levantou a necessidade de uma pesquisa mais aprofundada sobre o custo-benefício desta solução.

5.0.2 Perspectivas Futuras

Para a continuidade deste projeto, sugerimos a implementação de novos tipos de dispositivos, além da adição de outros equipamentos no banco de dados para mais testes. Com as ferramentas disponíveis, também é possível gerar relatórios de gerenciamento de dispositivos e monitoramento da utilização pelos usuários.

Outra proposta interessante é a criação de uma tela de perfil do usuário, onde possam ser exibidas suas informações, juntamente com uma interface de configuração básica para o aplicativo e garantir que o aplicativo seja capaz de detectar automaticamente dispositivos compatíveis na rede.

Por fim, um passo importante seria a disponibilização do aplicativo nas lojas de aplicativos, permitindo testes em ambiente de produção, o que contribuirá para a melhoria contínua da ferramenta e para a experiência do usuário.

Referências Bibliográficas

ALURA. React native: o que é e tudo sobre o framework. *reactnative*, 2024. Disponível em: <https://www.alura.com.br/artigos/react-native?srsltid=AfmBOorpFUN6jig42GgY5GNSYaN-NG_BSYzEvpaXOXelYJo3bfAStYzW, note="Acessoem:05deout.de2024">. Citado na página 6.

ARCHTOOLBOX. How air conditioners work. 2021. Acesso em: 03 de out. de 2024. Disponível em: <https://www.archtoolbox.com/how-air-conditioners-work/>. Citado na página 4.

CODE visual studio. visual studio code. Acesso em: 09 de out. de 2024. 2024. Disponível em: <https://code.visualstudio.com/docs>. Citado 2 vezes nas páginas 13 e 14.

DECHALERT, A. We need to talk about firebase. Acesso em: 8 de out. de 2024. 2019. Disponível em: <https://medium.com/madhash/weneed-to-talk-about-firebase-1ffea1180d75>. Citado na página 8.

EPE. Empresa de pesquisa energética. *Relatório Síntese do Balanço Energético Nacional*, 2024. Acesso em: 04 de ago. de 2024. Disponível em: <https://www.epe.gov.br/pt/publicacoes-dados-abertos/publicacoes/balanco-energetico-nacional-2024>. Citado na página 1.

ESPRESSIF. Esp32 series. *Espressif Systems*, 2012. Acesso em: 05 de ago. de 2024. Disponível em: <https://www.espressif.com/sites/default/files/documentation/esp32_datasheet_en.pdf>. Citado na página 10.

FIREBASE. Firebase realtime database. Acesso em: 8 de out. de 2024. 2020. Disponível em: <:https://firebase.google.com/docs/database/?hl=pt-br>. Citado 3 vezes nas páginas 8, 9 e 15.

IEA. *The Future of Cooling*. 2018. Disponível em: <https://www.iea.org/reports/the-future-of-cooling>. Citado na página 1.

KITABAYASHI, H. et al. Development of high power infrared led. *SEI Technical Review*, p. 71–74, Abril 2010. Citado na página 6.

LAYTON, J. Como tudo funciona. Acesso em: 8 de out. de 2024. 2021. Disponível em: <https://www.crcontroleremoto.com.br/como-funciona-um-controle-remoto.html>. Citado na página 5.

LV, J. et al. A new usb home appliances based on pc and infrared remote control protocol. In: *2010 International Conference on Computer and Communication Technologies in Agriculture Engineering*. [S.l.: s.n.], 2010. v. 3, p. 572–575. Citado na página 6.

MAKERHERO. Módulo wifi esp32 bluetooth. Acesso em: 05 de out. de 2024. 2017. Disponível em: <https://www.makerhero.com/produto/modulo-wifi-esp32-bluetooth/>. Citado 2 vezes nas páginas 10 e 11.

MAKERHERO. Led emissor infravermelho ir 5mm. Acesso em: 05 de out. de 2024. 2024. Disponível em: <https://www.makerhero.com/produto/led-emissor-infravermelho-ir-5mm/>. Citado na página 12.

RYTRONICS. Tsop1838 ir receiver. Acesso em: 05 de out. de 2024. 2024. Disponível em: <https://www.rytronics.in/product/tsop1838-ir-receiver/>. Citado na página 12.

SMACNA. Sistemas de ar condicionado. 2024. Acesso em: 04 de out. de 2024. Disponível em: <https://smacna.org.br/artigos-tecnicos/sistemas-de-ar-condicionado-principio-basico-de-refrigeracao/>. Citado na página 4.