

**Centro Universitário Estácio de Sá
Unidade Barreiros - São José - SC**

**Fiscalizador Eletrônico para Transportes PÚblicos
Aplicação de Cloud, IOT e Indústria 4.0 em Python**

**Igor Gomes Vieira, Joseph Patrick R. A. Trupel, Lucas Rachadel
Vagner Cordeiro**

**Novembro de 2024
São José - SC**

Sumário

1. DIAGNÓSTICO E TEORIZAÇÃO	4
1.1. Identificação das partes interessadas e parceiros	4
Passageiros dos transportes públicos	4
Empresas de Transporte Público	4
Prefeitura e Órgãos públicos	4
Estudantes	4
1.2. Problemática e/ou problemas identificados	4
Fonte de Dados	4
Principais problemas identificados	6
1.3. Justificativa	6
1.4. Objetivos/resultados/efeitos a serem alcançados (em relação ao problema identificado e sob a perspectiva dos públicos envolvidos)	7
1.5. Referencial teórico	8
Wagner Cordeiro	8
Flávio Guimarães	8
Alexander Lichter	8
Documentação do Wokwi	9
2. PLANEJAMENTO E DESENVOLVIMENTO DO PROJETO	9
2.1. Plano de trabalho	9
2.2. Envolvimento do público na formulação do projeto	10
2.3. Grupo de trabalho	11
Igor Gomes	11
Lucas Rachadel	11
Jhon	11
2.4. Metas, critérios ou indicadores de avaliação do projeto	11
2.5. Recursos previstos	12
2.6. Detalhamento técnico do projeto	13
Dispositivo IOT - Circuitos e Componentes	13
Dispositivo IOT - Integração com Micropython	14
Integração com Beebotte	27
Aplicativo para o usuário final	30
3. ENCERRAMENTO DO PROJETO	43
3.1. Relato Coletivo:	43
3.2. Relato de Experiência Individual	43
Igor Gomes	43
Joseph Patrick	43

1. DIAGNÓSTICO E TEORIZAÇÃO

1.1. Identificação das partes interessadas e parceiros

Passageiros dos transportes públicos

Os principais beneficiários. Geralmente são pessoas de diferentes idades, gêneros, escolaridades e perfis socioeconômicos variados, desde trabalhadores e estudantes até idosos e pessoas com necessidades especiais.

- Faixa Etária: Variada, entre 10 e 80 anos.
- Gênero: Todos os gêneros.
- Perfil Socioeconômico: A maioria possui renda média ou baixa, considerando que o transporte público é mais utilizado por aqueles que dependem de uma opção econômica e acessível.
- Escolaridade: Diversificada, de fundamental incompleto a ensino superior completo.

Empresas de Transporte Público

Operador de transporte público na região. Interessa-se pelo projeto para melhorar a qualidade do serviço oferecido, aumentar a satisfação dos usuários e otimizar a operação, especialmente no controle e manutenção de equipamentos de climatização, gestão de passageiros e controle de rotas.

Prefeitura e Órgãos públicos

São responsáveis pela regulação e fiscalização da qualidade do transporte público, incluindo aspectos de segurança e conforto. É também responsabilidade da prefeitura, de acordo com a legislação, de prover itinerários que atendem toda a população

Estudantes

Estudantes universitários com conhecimentos em tecnologia e desenvolvimento de software. Aplicam o conhecimento teórico-prático em uma solução que gere impacto social positivo, proporcionando uma melhor experiência no transporte público.

1.2. Problemática e/ou problemas identificados

Foram identificados problemas com base em comentários nas redes sociais, incluídos no anexo de apresentação. Além disso, utilizou-se como referência notícias sobre o transporte público nas cidades de São José e Florianópolis.

Fonte de Dados

Monitoramos páginas de notícias locais no Instagram para nos manter atualizados sobre o transporte público e entender a percepção da população sobre a situação atual. Selecionei os comentários mais relevantes, destacando as questões e desafios que podem ser abordados com soluções de IoT (Internet das Coisas), buscando insights que contribuam para melhorias efetivas e práticas na mobilidade urbana.

karolgodin 1 d
@terezinhaferndo peguei Jotur essa semana, sai sem coluna 😢 imagina quem usa esses ônibus todo dia

Responder Ver tradução

gisa_jambo 3 d
@nessaulian 99,99% porque ônibus em Floripa é um achado.

Responder Ver tradução

dionathanallan7 3 d
Queremos que diminua o valor da passagem isso sim.

Responder Ver tradução

silvanalucyfonseca 3 d
Pagando passagens caras já demoram horas. Imaginam sem pagar? Nem irão aparecerem nos pontos de parada.

Responder Ver tradução

vieiradesousafachini 2 d
Voltar os cobradores pros ônibus nem pensar né? O motorista fica sobrecarregado as X perdido nas multifunções, dar troco? Liberar catraca, descer pra ajudar cadeirante, as X passando o ponto, as fechando a porta antes da hora, um caos total, poderiam i vest

Responder Ver tradução

pbvilsondasilva 3 d
Tem que acabar com os velhos ônibus banco duro sem ar condicionado

Responder Ver tradução

galves84 3 d
O problema que o povo vai reclamar do ar condicionado... Antes tinha alguns ônibus com ar condicionado, tiraram... o povo reclamava direto que tava frio se foi esse motivo que tiraram nunca saberemos)... Mas que venha pra fica o ar...

mari_pioe 3 d
Ixpia só só Jotur que tal assim nox não ficaria Max na entrada parado por o ônibus se arrebatou tudim!

Responder Ver tradução

ckarol.sc 3 d
Pronto para pegar fogo 😊

ursulademyra 3 d
Gente e jotur pq não segue esse exemplo 😊😊

ghnqz77 3 d
Esse povo iludido acreditando em ônibus com ar 😊😊

edsonmanoela65 2 d
Eu saio mas de bicicleta é mas bom que ônibus

edsonmanoela65 2 d
Em florianópolis os ônibus bem ruuuuiii e os valor

pbvilsondasilva 3 d
então fica com a carroça e os acentos duro e sem ar condicionado

Responder Ver tradução

nessaulian 3 d
Ótimo, agora melhora os outros 99% dos ônibus velhos caíndo os pedacos, sem ar e com motoristas mal educados que agem como se estivessem fazendo um favor para as pessoas, transporte público de Florianópolis tem muito o que melhorar, precisa aprender com outros capitais e países que tem ónibus novos. Não adianta gastar dinheiro fazendo um ônibus top e moderno para ficar bonito para turista enquanto não melhora o básico da frota!

gisa_jambo 3 d
Mas os arcaicos terminais estão lá, deixando a vida do povo bem mais difícil do que já é.

Responder Ver tradução

maristela_boufeur 3 d
Me lembro dirigindo quando vim morar aqui em Floripa 2003, os ônibus tinham quase todos ar condicionados. Depois de algum tempo tiraram os ares e ninguém mais pediu nada. Retrocesso, é isso ai. Vamos ver se agora melhoram o transporte público e se ver se o povo de anima mais pra usar.💡

Responder Ver tradução

"Os horários das linhas são muito ruins, insuficientes. Às vezes, passam dois ônibus quase em seguida e aí temos um espaço de tempo de 40 minutos durante a semana, em horário comercial, para um outro ônibus que faz a linha, passar pela rota", comenta.

Alex Souza
Busão da Jotur é igual coração de crush: escuro e causador de insegurança.

Mile Paes
Outro dia não consegui entrar no ônibus que viria do porto da lagoa. O motorista parou. Olhou pra mim eu falei estou usando rexona mais não vou. O tava cheio até a boca

6 ano Curtir Responder

"No meu caso, duas linhas passam próximo a minha casa. Sempre vem um ônibus na cola do outro, passam juntos praticamente, e aí fica um longo período sem passar outro. Não entendo o porquê de os responsáveis não distribuem melhor o horário a fim de ofertar mais opções pra gente", destaca a carioca que há 22 anos trocou o Rio de Janeiro pela capital catarinense.

gutenberg__limaa 3 d
@thubernert kkkkk os CLT passando sufoco na volta do trabalho cansado, e ainda ter que suportar o mal cheiro de CC, chulé, de muita gente. kkkkkk

Responder Ver tradução

waltingho1923 3 d
@thubernert melhor 50 em pé no calor de 30 graus com ar condicionado do que sem ar né?

Responder Ver tradução

thubernert 3 d
Melhor é um ônibus com bastante lugar e ar condicionado. Porque o povo sempre tem que escolher o menos pior? O valor de imposto e da passagem que nós pagamos dava tranquilamente pra isso!!

Responder Ver tradução

anacristina_holistica 3 d
@joaogabrielgbatista fizeram pior... demitiram os cobradores e hoje motorista acunula duas funções estressantes... TUDO SÓ PIORA.

Responder Ver tradução

joaogabrielgbatista 3 d
Sempre achei um absurdo Floripa ter tido ônibus com ar nos anos 2000 e depois terem tirado e deixado só no amarelinho..

carvalhocad 3 d
@si_shanti essa linha precisa de mais horários com urgência.

Responder Ver tradução

yng.nyk 3 d
@morandjuliana real, o direto para na beira mar sempre de manhã não entendo isso, e ainda por cima o direto só tem o tamano menor e eles sabem que sempre vai lotado e não colocam um ônibus duplo como direto

morandjuliana 3 d
@si_shanti claro que não! Ainda fizeram aquele direto que para até no inferno!!! As vezes a inteligência dos governantes me assusta

rejozanotta 5 d
O uso do cinto não é obrigatório?????

Responder Ver tradução

michel.ktotz 5 d
Só param após perderem uns dentes 😊

Responder Ver tradução

bruno_leao82 5 d
Esporte mais radical de floripa, é pegar madrugadão, tilag - tiro, via canto da lagoa hahahahahaha. Need for speed não chega perto!

costa2390 2 d
Se fosse jotur estaria tudo normal 😊

gabrielceliosilva 3 d
@joaogabrielgbatista tem que agradecer o @dariobergersc . A retirada do ar-condicionado foi autorizada no mandato dele, se não fosse retirado na época, hoje em dia teríamos 100% da frota com ar-condicionado.

macedo.taninha 3 d
Muito legal que também façam isso com as outras rotas de ônibus porque ninguém merece esses ônibus com essas janelas minúsculas o povo passando calor vê se tenham um pouco de bom senso com nós usuários de ônibus né!!!! 😊😊😊

artur_eger 3 d
@joaogabrielgbatista eu vim morar agora aqui eu achei ridículo isso, ônibus tem que ser gratuito, com ar e wifi, esse é o mínimo, em Balneário Camboriú é assim o os ônibus não ficam lotados iguais em floripa

gabrielceliosilva 3 d
@joaogabrielgbatista Concordo contigo, porém é um vulgo teste e não é garantia wie vai ter ar-condicionado nas futuras aquisições. Espero me surpreender.

Eduardo Kiresmerio
Planejamento? NÃO TEMOS

Transporte Pùblico Deficiente: O transporte público em Florianópolis ainda é considerado inadequado em termos de cobertura, frequência e qualidade dos serviços oferecidos. Muitos moradores enfrentam dificuldades para acessar os pontos de ônibus e sofrem com a falta de linhas diretas e horários irregulares.

28 de maio de 2024

REFERENCIAL TEÓRICO

- EM UMA NOTÍCIA POSTADA NO SITE DO G1 EM 2023, FOI MOSTRADO UMA PESQUISA EM RELAÇÃO AOS VÁRIOS PROBLEMAS COM TRANSPORTES PÚBLICOS, ENTRE OS PROBLEMAS RELATADOS, TEMPERATURA FOI UM DOS MAIS MENCIONADOS PELOS ENTREVISTADOS. DURANTE PERÍODOS DE EXTREMO CALOR E A GRANDE QUANTIDADE DE PASSAGEIROS DURANTE HORÁRIOS DE PICO, MUITAS PESSOAS MOSTRARON GRANDE INSATISFAÇÃO COM A FALTA DE UM SISTEMA DE RESFRIAMENTO NOS TRANSPORTES PÚBLICOS.



Principais problemas identificados

Sistema de Refrigeração Precário

Durante períodos de extremo calor, a falta de um sistema de climatização eficaz causa grande desconforto para passageiros que dependem diariamente do transporte público.

Valor das Passagens Acima do Normal

Na região da Grande Florianópolis, as tarifas de transporte público estão entre as mais altas do Brasil, o que impacta o custo de vida e torna o serviço menos acessível.

Falta de Planejamento e Lotação

Má distribuição do itinerário, com linhas de ônibus frequentemente superlotadas enquanto outras permanecem vazias. Isso indica uma necessidade de melhor planejamento e otimização das rotas.

Desrespeito às Leis de Trânsito

Existem muitas reclamações sobre motoristas que trafegam acima da velocidade permitida, resultando em situações de risco e ferimentos para passageiros dentro do transporte.

1.3. Justificativa

A problemática identificada – as condições precárias no transporte público, especificamente em questões de climatização, superlotação e falta de planejamento – é pertinente academicamente pois reflete um cenário real que impacta a qualidade de vida e segurança dos usuários. Em um contexto de Aprendizagem Baseada em Projetos, a proposta permite que os estudantes integrem conhecimentos técnicos de Internet das Coisas (IoT) para desenvolver soluções que respondam a demandas concretas.

A relação com o curso “Aplicação de Cloud, IoT e Indústria 4.0 em Python” é direta, pois o projeto visa aplicar conceitos teóricos em um contexto prático, ampliando a capacidade dos alunos de

desenvolver e implementar soluções tecnológicas inovadoras. A motivação do grupo de trabalho é proporcionar melhorias efetivas no transporte público, ao mesmo tempo em que expandem suas habilidades em programação, sensores, análise de dados e integração de sistemas. Essa abordagem não apenas aprofunda o entendimento dos estudantes sobre IoT, mas também os prepara para resolver problemas complexos e dinâmicos, como os enfrentados por grandes centros urbanos no setor de mobilidade.

1.4. Objetivos/resultados/efeitos a serem alcançados (em relação ao problema identificado e sob a perspectiva dos públicos envolvidos)

- Identificar padrões e comportamentos críticos no transporte público relacionados à superlotação e desconforto térmico, gerando dados para ajustes e melhorias que aumentem a segurança e o bem-estar dos usuários.
- Capacitar os estudantes no desenvolvimento de uma solução IoT prática e inovadora que oferece uma visão integrada do funcionamento do transporte público, alinhada com as metas do curso e aplicável a outros contextos.
- Incentivar a colaboração entre usuários e gestores de transporte por meio de uma plataforma de monitoramento, promovendo uma experiência de transporte mais segura, confiável e eficiente para a população.
- O sistema de monitoramento em tempo real para o transporte público trará melhorias significativas no conforto, segurança e conveniência dos passageiros. Ao captar dados precisos sobre temperatura, lotação e localização dos veículos, essa solução permitirá que as empresas de transporte público ajustem de forma proativa a climatização e a distribuição de veículos, proporcionando um ambiente mais agradável e seguro para todos, especialmente em dias de clima extremo.
- Com a disponibilização de um dashboard analítico, operadores e gestores de transporte terão acesso a informações essenciais para decisões mais rápidas e eficientes, ajudando a otimizar rotas, reduzir lotação e promover um uso mais sustentável dos sistemas de climatização. Essa iniciativa também possibilitará uma fiscalização mais efetiva por parte dos órgãos públicos, que terão dados concretos para avaliar e melhorar a qualidade do serviço oferecido.
- Para os passageiros, o sistema oferece uma nova experiência de transporte: por meio de um aplicativo, eles poderão acompanhar em tempo real a temperatura, localização e lotação dos veículos, o que contribui para uma experiência mais prática e confiável, permitindo que planejem melhor suas viagens.
- Além de melhorar o conforto e a segurança dos passageiros, o sistema promoverá uma operação mais sustentável, reduzindo o consumo energético e o desgaste dos equipamentos de climatização com o uso otimizado de ar-condicionado e ventilação.

1.5. Referencial teórico

Wagner Cordeiro

Em suas aulas abordou os princípios fundamentais da Internet das Coisas (IoT) e a relevância do uso de tecnologias de comunicação para sistemas IoT. Durante as aulas, o professor enfatizou o conceito de IoT como uma rede de dispositivos conectados que trocam informações, viabilizando a automação e análise de dados.

O professor Cordeiro também explicou a importância de arquiteturas de comunicação baseadas em cloud (ou nuvem), que atuam como um middleware essencial para o gerenciamento e o processamento dos dados coletados por dispositivos IoT. Ele detalhou o funcionamento de protocolos como o MQTT (Message Queuing Telemetry Transport), que é amplamente utilizado em redes IoT por seu baixo consumo de energia e por oferecer uma conectividade robusta e de baixa latência.

As aulas e materiais complementares de Wagner Cordeiro, servem como referências para nossa proposta, dando suporte ao desenvolvimento de uma solução baseada em IoT com uma comunicação eficaz e estruturada para a coleta e análise de dados em tempo real.

Flávio Guimarães

Outro referencial relevante para nossa proposta é o Flávio Guimarães, criador do canal Brincando com Ideias no YouTube. Em vídeos como "Melhores Fontes - Automação com ESP32 e Programação Arduino #MaratonaMaker", Guimarães explica com clareza e didática os fundamentos da automação e da Internet das Coisas (IoT) utilizando o microcontrolador ESP32.

Em seus conteúdos, ele aborda aspectos técnicos como tensão, voltagem, e a configuração de circuitos eletrônicos, contextualizando esses conceitos para aplicações práticas em projetos de IoT. A linguagem acessível e a abordagem prática adotada por Guimarães são de grande valor para o desenvolvimento deste projeto, especialmente por tornar compreensíveis conceitos técnicos necessários para a implementação de sensores e sistemas IoT.

A experiência compartilhada em seu canal serve como um recurso complementar importante para entender melhor as particularidades do ESP32, um componente amplamente utilizado em redes IoT devido à sua versatilidade e capacidade de comunicação em redes Wi-Fi.

Alexander Licher

Um dos referenciais importantes para nossa proposta é o desenvolvedor Alexander Licher, conhecido por seus ensinamentos e conteúdos práticos sobre Vue e Nuxt 3. Licher tem sido fundamental para o aprendizado sobre o desenvolvimento de aplicações web, oferecendo uma base sólida para construir interfaces modernas e responsivas que integram os conceitos de IoT em uma plataforma acessível ao usuário final.

Suas aulas e tutoriais abordam desde os conceitos básicos até as práticas avançadas de Vue.js e Nuxt 3, fornecendo uma compreensão detalhada sobre como criar aplicações escaláveis e eficientes. Esses conhecimentos foram essenciais para estruturar uma interface web capaz de exibir os dados capturados pelo sistema de IoT, facilitando o acesso de usuários e operadores a informações em tempo real.

Documentação do Wokwi

Não menos importante, a documentação do Wokwi foi essencial para o desenvolvimento do nosso projeto, especialmente por abordar uma ampla variedade de sensores e fornecer instruções detalhadas sobre como montar placas com o ESP32. A plataforma Wokwi permite simular circuitos e dispositivos IoT, facilitando o aprendizado prático e o teste de configurações sem a necessidade de hardware físico.

Além disso, a documentação inclui exemplos e tutoriais de programação em MicroPython, uma linguagem muito utilizada para dispositivos de IoT devido à sua leveza e facilidade de integração com sensores. Esses recursos permitiram à equipe experimentar diferentes combinações de sensores e componentes, entender como conectá-los de maneira eficaz e otimizar o código para uma operação mais eficiente. A documentação do Wokwi, com seus tutoriais claros e ilustrativos, complementa nosso referencial teórico e fornece uma base sólida para a aplicação prática dos conceitos de IoT no projeto.

2. PLANEJAMENTO E DESENVOLVIMENTO DO PROJETO

2.1. Plano de trabalho

O plano de trabalho do projeto foi desenvolvido com um cronograma que define etapas claras, prazos, responsáveis e recursos necessários. As reuniões e atividades de desenvolvimento poderão ocorrer de forma individual ou grupal, conforme a necessidade do grupo.

Objetivo	Atividade	Prazo	Responsáveis	Recursos
Discutir ideias iniciais e definir a problemática a ser abordada com IoT.	Brainstorming de problemas e soluções potenciais; elaboração dos primeiros rascunhos de objetivos e resultados esperados.	1ª Semana	Todos os membros do grupo.	Anotações Digitais
Integração inicial com a plataforma Wokwi para simulação do sistema IoT.	Familiarização com o Wokwi e configuração básica dos sensores,	2ª Semana	Todos os membros do grupo.	Wokwi, ESP32, documentação de sensores, Thinkercad.

Criar um slide de apresentação da ideia do projeto para o docente.	Desenvolvimento de uma apresentação inicial que descreva a proposta, objetivos e benefícios do projeto.	3ª Semana	Todos os membros do grupo.	Rascunhos, Canva, Discord
Iniciar o desenvolvimento do sistema IoT.	Configuração e programação dos sensores para monitoramento de temperatura e lotação.	4ª Semana	Todos os membros do grupo.	Wokwi, editor de código, Micropython
Comunicação do dispositivo IOT	Comunicação MQTT no projeto do Wokwi, utilizando o Beebotte	5ª Semana	Joseph	Wokwi, Beebotte, Micropython, MQTT
Dashboard em tempo real Beebotte	Implementação do sistema de dashboard, integração dos dados e ajustes de configuração.	6ª Semana	Joseph	Beebotte
Interface para usuário final	Implementar uma tela para que usuários de transporte público possam acessar dados dos mesmos.	7ª Semana	Joseph	Vuejs, MQTT, Javascript, Beebotle API, Google Maps API
Criação do Vídeo de Apresentação	Criar um vídeo em formato de pitch, explicando o conceito do nosso projeto e colocando em demonstração	8ª e 9 Semana	Igor, Joseph	Canva, Discord, Editor de Vídeo, Gravações

2.2. Envolvimento do público na formulação do projeto

O envolvimento do público participante no projeto ocorreu de forma informal, por meio de conversas diretas com usuários do transporte público. Embora não tenhamos realizado reuniões

estruturadas, essas interações proporcionam insights valiosos sobre as condições e desafios enfrentados diariamente, especialmente em relação à superlotação e ao desconforto térmico.

Além disso, a opinião dos usuários nas redes sociais teve um papel importante, pois monitoramos comentários e discussões em plataformas digitais para entender as principais reclamações e sugestões sobre o transporte público. Essas observações complementam nossas conversas diretas, garantindo que a formulação do projeto estivesse alinhada com as demandas reais da comunidade e refletisse as expectativas dos próprios usuários.

2.3. Grupo de trabalho

Igor Gomes

Responsabilidade: Organização do andamento do projeto, desenvolvimento do roteiro de extensão e contribuição na preparação dos slides de apresentação e gravação de vídeos.

Lucas Rachadel

Responsabilidade: Contribuição e otimização dos slides de apresentação, coleta de dados do público alvo

Jhon

Responsabilidade: Construção eletrônica, e codificação, comunicação com a plataforma e coleta de dados e contribuição na preparação da gravação de vídeos.

2.4. Metas, critérios ou indicadores de avaliação do projeto

Etapa	Ação	Critério de Avaliação	Indicador de Avaliação
Pesquisa e Análise Inicial de Dados	Realizar pesquisa em redes sociais e páginas de notícias locais, analisando comentários sobre transporte público.	Compilar pelo menos 50 comentários relevantes para embasar o projeto.	Documentação dos problemas levantados pelos usuários para uso na introdução e embasamento teórico do projeto.
Testes Práticos em Sala de Aula	Realizar testes com dispositivos IoT simulados no Wokwi e pesquisas para aprimorar o projeto.	Finalização bem-sucedida de testes de conexão e coleta de dados com sensores simulados.	Registro dos testes realizados, indicando limitações e resoluções encontradas.
Simulação do Dispositivo IoT no Wokwi	Configurar o ESP32 com programação em MicroPython e integrar sensores como movimentação, temperatura, umidade e GPS.	Configuração completa e funcional do ESP32 com sensores.	Comprovação de dados de sensores enviados para sistema central, como temperatura e localização.

Conexão à Internet e Envio de Dados via MQTT	Estabelecer conexão WiFi para o ESP32 e configurar envio de dados via protocolo MQTT.	Conexão estável e envio de dados sem interrupções para o servidor MQTT.	Logs que comprovem a transmissão e recepção de dados por meio do protocolo MQTT.
Apresentação dos Dados Coletados	Integrar display, LEDs e buzzer para exibir temperatura, lotação e localização em tempo real no dispositivo e em uma aplicação web.	Demonstração visual do protótipo com feedback em tempo real sobre a situação do ônibus.	Evidências visuais do sistema em operação, com exibição dos dados coletados em tempo real.
Desenvolvimento da Aplicação Web e Dashboard Beebotte	Desenvolver uma aplicação para o usuário final e configurar um dashboard analítico para operadores no Beebotte.	Aplicação web e dashboard acessíveis, com dados atualizados em tempo real.	Testes de usabilidade e precisão de informações exibidas no aplicativo e dashboard.
Apresentação do Projeto em Formato Pitch	Criar vídeo de apresentação estilo pitch, destacando benefícios, viabilidade e impacto do sistema.	Clareza e objetividade do pitch, com apresentação impactante dos benefícios da solução.	Feedback de professores e colegas sobre a clareza e viabilidade do projeto.
Orçamento e Análise de Viabilidade	Desenvolver um orçamento cobrindo custos do projeto e projeção de economia com a implementação.	Análise de viabilidade financeira que demonstre um retorno de investimento positivo.	Relatório de custos e benefícios, com estimativas de impacto financeiro da solução no longo prazo.

2.5. Recursos previstos

Por ser tudo simulado, não houve necessidade de gastos financeiros para o desenvolvimento do projeto. No entanto, foi elaborado um orçamento estimado para a criação da placa, considerando os componentes necessários para uma implementação real. Esse orçamento é baseado em preços de componentes disponíveis no mercado dos Estados Unidos, com valores totais em dólares e convertidos para reais.

Em relação à hospedagem, utilizamos exclusivamente serviços gratuitos, assegurando que o projeto pudesse ser desenvolvido sem custos adicionais e mantendo o foco em estratégias acessíveis e sustentáveis para a execução do trabalho.

Item	Descrição	Qtd.	Preço Unitário	Total
LED RGB	LED RGB (SparkFun, modelo COM-12062)	1	\$1.00	\$1.00
Buzzer	Mini Buzzer (Adafruit, modelo 1264)	1	\$1.00	\$1.00
Sensor de Movimento	PIR Motion Sensor (Adafruit, modelo PIR-AM312)	2	\$2.00	\$4.00
Módulo GPS	U-blox NEO-6M GPS Module	1	\$8.00	\$8.00
ESP32 com WiFi	ESP32 WiFi & Bluetooth (D1 Mini)	1	\$3.50	\$3.50

	ESP32			
Resistor	Resistor 220 ohm (SparkFun, modelo COM-09120)	3	\$0.10	\$0.30
Fios de Conexão	Fios de Conexão (SparkFun, modelo PRT-09140)	1 pacote (10)	\$2.00	\$2.00

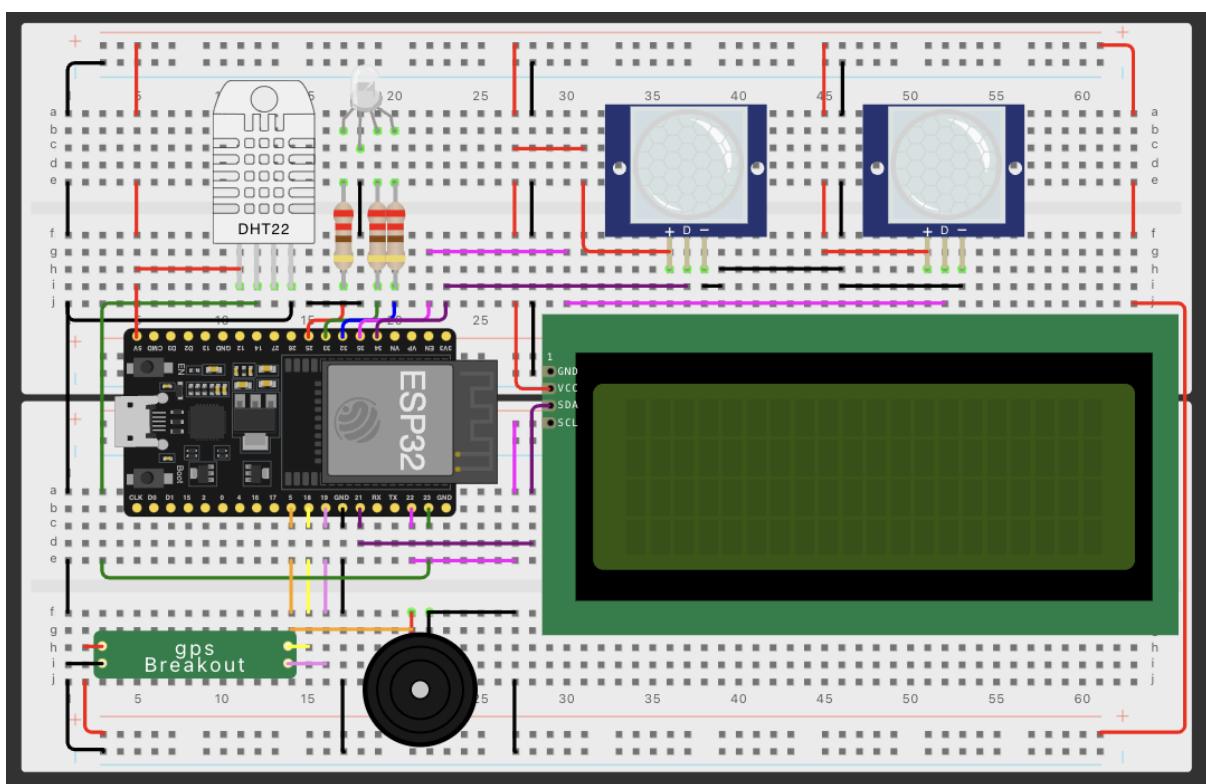
Total em Dólares (USD): \$19.80

Cotação em Reais (BRL) (\$1 USD = R\$5.00): R\$99.00

2.6. Detalhamento técnico do projeto

Dispositivo IOT - Circuitos e Componentes

O esquema abaixo mostra a organização dos circuitos e componentes conectados em uma protoboard para facilitar a montagem e testes do dispositivo IoT destinado ao monitoramento de transporte público. O uso da protoboard permite realizar conexões temporárias e ajustáveis entre os componentes e a placa ESP32, possibilitando uma configuração experimental prática e segura antes de uma montagem definitiva.



Componentes utilizados

Componente	Descrição	Pinos ESP32	Conexões e Observações
ESP32	Placa principal para o controle de todos os	-	Placa central onde todos os sensores e atuadores estão

	componentes e comunicação IoT		conectados.
Sensor de Temperatura e Umidade (DHT22)	Sensor para monitorar temperatura e umidade.	Pino 23	Conectado ao pino 23 do ESP32 para leitura de temperatura e umidade.
GPS (U-blox NEO-6M)	Módulo GPS para obter localização do veículo.	Pino 5 (OUT), Pino 18 (IN)	Usa dois pinos: GPS_OUT_PIN no pino 5 para saída de dados e GPS_IN_PIN no pino 18 para entrada.
Sensor de Movimento (PIR)	Detecta movimento para monitorar entrada e saída de passageiros.	Pino 34 (Esquerda), Pino 35 (Direita)	Dois sensores conectados: um para monitorar a entrada de passageiros e outro para a saída
LED RGB	LED RGB para indicar status de forma visual com luzes coloridas.	Pino 25 (R), Pino 33 (G), Pino 32 (B)	Conectado ao ESP32 nos pinos: LED_R_PIN (Vermelho) no pino 25, LED_G_PIN (Verde) no pino 33 e LED_B_PIN (Azul) no pino 32.
Buzzer	Dispositivo sonoro para alertas e notificações audíveis.	Pino 5	Conectado ao BUZZER_PIN no pino 5 do ESP32 para emitir sons de alerta.
LCD Display (I2C)	Display para exibir informações como temperatura, lotação e localização.	SDA (Pino 21), SCL (Pino 22)	Conectado ao ESP32 usando comunicação I2C. Parâmetros I2C: Endereço 0x27, linhas I2C_NUM_ROWS = 4, colunas I2C_NUM_COLS = 20. SDA no pino 21 e SCL no pino 22.

Observações para o LCD Display com I2C

O LCD é configurado com o endereço I2C 0x27, facilitando a conexão com o ESP32 sem a necessidade de múltiplos fios para cada pino do display.

Com 4 linhas e 20 colunas, o display oferece espaço suficiente para exibir informações em tempo real sobre temperatura, lotação e localização.

A comunicação I2C usa o SDA_PIN (pino 21) e o SCL_PIN (pino 22) para transmissão de dados, permitindo uma conexão eficiente e economizando pinos GPIO do ESP32.

Dispositivo IOT - Integração com Micropython

Na produção deste projeto, foi desenvolvido um código em Python, disponível para acesso por meio do seguinte link: <https://wokwi.com/projects/413996323601280001>. Informações adicionais estão apresentadas nos anexos fornecidos abaixo.

Código Principal (main.py)

```
1 #####  
2 # Dependências #  
3 #####  
4 import time # Usado para pausar a aplicacao  
5 import dht # Usado para o sensor de temperatura e umidade  
6 from machine import Pin, SoftI2C, reset # Uso do ESP32  
7 from umqtt.simple import MQTTClient # Enviar dados para o servidor  
8 from machine_i2c_lcd import I2cLcd # Interagir com o Display  
9 from utilsdefs import (  
10     play_note, log_display,  
11     send_data, connect_wifi,  
12     show_info_on_display, show_time_on_display  
13 ) # Funções Separadas para melhor organização do código
```



```
1 #####  
2 # Parâmetros #  
3 #####  
4  
5 # Parâmetros da conexão do WI-FI  
6 WIFI_LOGIN = "Wokwi-GUEST"  
7 WIFI_PASSWORD = ""  
8  
9 # Parâmetros da conexão do MQTT  
10 MQTT_CLIENT_ID = "ofdrsXZPBnW6xF7acIfmlmQq4sGNKwNC"  
11 MQTT_BROKER = "mqtt.beebotte.com"  
12 MQTT_USER = "ofdrsXZPBnW6xF7acIfmlmQq4sGNKwNC"  
13 MQTT_PASSWORD = "ofdrsXZPBnW6xF7acIfmlmQq4sGNKwNC"  
14 MQTT_BASE_TOPIC = "trip13601"  
15 MQTT_PORT = 8883  
16  
17 # Parâmetros do I2C (usado para simplificar a estrutura) e o LED LCD  
18 I2C_ADDR = 0x27  
19 I2C_NUM_ROWS = 4  
20 I2C_NUM_COLS = 20  
21 SDA_PIN = Pin(21)  
22 SCL_PIN = Pin(22)  
23  
24 # Parâmetros do Sensor de Movimentacao  
25 MOTIONSENSOR1_PIN = Pin(34, Pin.IN) # Sensor da Esquerda (Entrada de Passageiros)  
26 MOTIONSENSOR2_PIN = Pin(35, Pin.IN) # Sensor da Direita (Saida de Passageiros)  
27  
28 # Parâmetros do Sensor de Temperatura e Umidade  
29 DHT22_PIN = Pin(23)  
30  
31 # Parâmetros do Sensor de GPS  
32 GPS_OUT_PIN = Pin(5)  
33 GPS_IN_PIN = Pin(18)  
34  
35 # Parâmetros do led RGB  
36 LED_R_PIN = Pin(25, Pin.OUT) # Vermelho  
37 LED_G_PIN = Pin(33, Pin.OUT) # Verde  
38 LED_B_PIN = Pin(32, Pin.OUT) # Azul  
39  
40 # Parâmetros do buzzer  
41 BUZEER_PIN = 5
```



```
1 #####  
2 # Inicialização dos Componentes e Variaáveis #  
3 #####  
4  
5 # Inicializar LCD  
6 i2c = SoftI2C(sda=SDA_PIN, scl=SCL_PIN, freq=400000)  
7 lcd = I2cLcd(i2c, I2C_ADDR, I2C_NUM_ROWS, I2C_NUM_COLS)  
8  
9 # Inicializar GPS  
10 altitude = -27.631182  
11 longitude = -48.641013  
12  
13 # Inicializar contagem de passageiros  
14 motion_1_status = False  
15 motion_2_status = False  
16 persons_in = 0  
17 persons_out = 0  
18 persons_total = 0  
19  
20 # Guardar registro da última medição, para não enviar dados duplicados  
21 prev_temperature = -1  
22 prev_humidity = -1  
23 prev_persons_in = -1  
24 prev_persons_out = -1  
25 prev_persons_total = -1
```

```
1 ######
2 # Código de Execução Única #
3 #####
4
5 # Acionar catch em caso de erros
6 try:
7     # Inicializar LED
8     # Pinos configurados em LOW são acessos. Acender LED branco (Todas as cores)
9     LED_R_PIN.off()
10    LED_G_PIN.off()
11    LED_B_PIN.off()
12
13    # Inicializar Sensor de Temperatura
14    sensor = dht.DHT22(DHT22_PIN)
15
16    # Inicializar Wi-Fi
17    log_display(lcd, "Iniciar WI-FI...")
18    play_note(BUZEER_PIN, lcd)
19    connect_wifi(WIFI_LOGIN, WIFI_PASSWORD)
20    log_display(lcd, "Wifi Conectado!")
21    play_note(BUZEER_PIN, lcd)
22
23    # Inicializar conexão MQTT
24    log_display(lcd, "Conectar MQTT...")
25    play_note(BUZEER_PIN, lcd)
26    client = MQTTClient(MQTT_CLIENT_ID, MQTT_BROKER, user=MQTT_USER, password=MQTT_PASSWORD, port=MQTT_PORT, ssl=True)
27    client.connect()
28    log_display(lcd, "MQTT conectado!")
29    play_note(BUZEER_PIN, lcd)
```



```
1 #####  
2 # Início do Loop Principal #  
3 #####  
4 while True:  
5     changed = False # Variavel para verificar se algo mudou no display  
6     sensor.measure() # Efetuar a medição  
7     temperature = sensor.temperature() # Temperatura  
8     humidity = sensor.humidity() # Umidade  
9     # Entrada de Passageiros  
10    readMotion1 = MOTIONSENSOR1_PIN.value()  
11    if(readMotion1 == 1):  
12        if not motion_1_status:  
13            changed = True  
14            persons_in += 1  
15            motion_1_status = True  
16        else:  
17            if motion_1_status:  
18                motion_1_status = False  
19    # Saída de Passageiros  
20    readMotion2 = MOTIONSENSOR2_PIN.value()  
21    if(readMotion2 == 1):  
22        if not motion_2_status:  
23            changed = True  
24            persons_out += 1  
25            motion_2_status = True  
26        else:  
27            if motion_2_status:  
28                motion_2_status = False  
29    #  
30    persons_total = max(persons_in - persons_out, 0) # Total de Passageiros
```

```
 1 ######
 2 # Enviar dados da temperatura #
 3 #####
 4 if temperature != prev_temperature:
 5     # Enviar novos dados ao servidor MQTT
 6     changed = True
 7     prev_temperature = temperature
 8     log_display(lcd, "Enviando dados de temperatura...")
 9     play_note(BUZEER_PIN, lcd)
10     send_data(client, MQTT_BASE_TOPIC, "temperature", temperature)
11
12 # Acrender led vermelho em caso de calor, azul em caso de frio e verde temperatura normal
13 if temperature < 15:
14     LED_R_PIN.on()
15     LED_G_PIN.on()
16     LED_B_PIN.off()
17 elif temperature > 26:
18     LED_R_PIN.off()
19     LED_G_PIN.on()
20     LED_B_PIN.on()
21 else:
22     LED_R_PIN.on()
23     LED_G_PIN.off()
24     LED_B_PIN.on()
25 #
```



```
1 #####  
2 # Enviar dados da Umidade e Quantidade de Passageiros #  
3 #####  
4 if prev_humidity != humidity:  
5     # Enviar novos dados ao servidor MQTT  
6     changed = True  
7     prev_humidity = humidity  
8     log_display(lcd, "Enviando dados de umidade...")  
9     play_note(BUZEER_PIN, lcd)  
10    send_data(client, MQTT_BASE_TOPIC, "humidity", humidity)  
11    #  
12  
13 if prev_persons_in != persons_in:  
14     # Enviar novos dados ao servidor MQTT  
15     changed = True  
16     prev_persons_in = persons_in  
17     log_display(lcd, "Enviando dados de Passageiros...")  
18     play_note(BUZEER_PIN, lcd)  
19     send_data(client, MQTT_BASE_TOPIC, "persons_in", persons_in)  
20    #  
21  
22 if prev_persons_out != persons_out:  
23     # Enviar novos dados ao servidor MQTT  
24     changed = True  
25     prev_persons_out = persons_out  
26     log_display(lcd, "Enviando dados de Passageiros")  
27     play_note(BUZEER_PIN, lcd)  
28     send_data(client, MQTT_BASE_TOPIC, "persons_out", persons_out)  
29    #  
30  
31 if prev_persons_total != persons_total:  
32     # Enviar novos dados ao servidor MQTT  
33     changed = True  
34     prev_persons_total = persons_total  
35     log_display(lcd, "Enviando dados de Passageiros")  
36     play_note(BUZEER_PIN, lcd)  
37     send_data(client, MQTT_BASE_TOPIC, "persons_total", persons_total)  
38    #
```

```
● ● ●  
1 #####  
2 # Atualização do Display, e intervalo antes de reiniciar o loop #  
3 #####  
4  
5 # Verifica se houve alterações antes de atualizar o display  
6 if changed:  
7     show_info_on_display(lcd, temperature, humidity, altitude, longitude, persons_in, persons_out)  
8 # Atualizar hora no display  
9 show_time_on_display(lcd)  
10 # Aguardar 1 segundo antes de reiniciar loop  
11 time.sleep(1)
```

```
● ● ●  
1 #####  
2 # Tratamento de Erros #  
3 #####  
4  
5 # Em caso de erros (como de conexão), reiniciar o sistema  
6 except OSError as e:  
7     text = str(e)  
8     print("ERRO:")  
9     print(text)  
10    lcd.clear()  
11    log_display(lcd, "Erro interno, reiniciando sistema...")  
12    repetitions = 6  
13    for _ in range(repetitions):  
14        play_note(BUZEER_PIN, lcd)  
15    time.sleep(5)  
16    reset()
```

Funções Úteis (utilsdefs.py)



```
1 import network # Usado para conectar o WI-FI
2 from machine import PWM, Pin # Funções do ESP32
3 from time import sleep, time # Pausar aplicação e pegar hora atual
4 import ujson # Usado para enviar dados em JSON no servidor
5 import utime # Usado para converter datas e horas
```



```
1 # Método para acionar o buzzer
2 def play_note(pin, lcdForBlink = None, freq=440, sleepduration=0.1):
3     if lcdForBlink is not None:
4         lcdForBlink.backlight_off()
5         audio = PWM(Pin(pin), 1)
6         audio.freq(freq)
7         audio.duty(512)
8         sleep(sleepduration)
9         audio.deinit()
10    if lcdForBlink is not None:
11        lcdForBlink.backlight_on()
12    #
```



```
1 # Método para formatar hora
2 def format_timestamp(timestamp):
3     # Convert the timestamp to a local time tuple
4     time_tuple = utime.localtime(timestamp)
5     # Format time in HH:MM:SS format
6     formatted_time = "{:02}:{:02}:{:02}".format(time_tuple[3], time_tuple[4], time_tuple[5])
7     return formatted_time
8 #
```

```
● ● ●  
1 # Método para formatar coordenadas  
2 def format_coordinate(coord_str):  
3     # Verificar se já contém ponto decimal  
4     if '.' not in coord_str:  
5         coord_str += '.'  
6     # Completar com zeros à direita até atingir 20 caracteres  
7     while len(coord_str) < 20:  
8         coord_str += '0'  
9     # Substituir o ponto por uma vírgula  
10    coord_str = coord_str.replace('.', ',')  
11    return coord_str  
12 #
```

```
● ● ●  
1 # Método para formatar números com 3 dígitos  
2 def format_persons(persons):  
3     persons_str = str(persons)  
4     # Adicionar zeros à esquerda até o comprimento ser 3  
5     while len(persons_str) < 3:  
6         persons_str = '0' + persons_str  
7     return persons_str  
8 #
```

```
● ● ●  
1 # Método para demonstrar informações no Display  
2 def show_info_on_display(lcd, temperature, humidity, lat, lng, persons_in, persons_off):  
3     lcd.clear()  
4     lcd.move_to(0,0)  
5     lcd.putstr(str(temperature) + "C")  
6     lcd.move_to(6,0)  
7     lcd.putstr(str(humidity) + "%")  
8     lcd.move_to(12,0)  
9     lcd.putstr(format_timestamp(time()))  
10    lcd.move_to(0,1)  
11    latstr = format_coordinate(str(lat))  
12    lngstr = format_coordinate(str(lng))  
13    lcd.putstr("Lat: " + latstr)  
14    lcd.move_to(0,2)  
15    lcd.putstr("Lng: " + lngstr)  
16    lcd.move_to(0,3)  
17    lcd.putstr("Entrd. " + format_persons(persons_in) + " Said. " + format_persons(persons_off))  
18 #
```

```
● ● ●  
1 # Método para atualizar somente a hora no display  
2 def show_time_on_display(lcd):  
3     lcd.move_to(12,0)  
4     lcd.putstr(format_timestamp(time()))
```



```
1 # Método para simplificar a mostragem de uma mensagem
2 def log_display(lcd, message):
3     lcd.clear()
4     lcd.move_to(1,1)
5     lcd.putstr(message)
6 #
```



```
1 # Método para simplificar o envio de dados para o MQTT
2 def send_data(client, base_topic, topic, data):
3     client.publish(base_topic + "/" + topic, ujson.dumps({
4         "data": data,
5         "write": True,
6     }))
7 #
```



```
1 # Método para conectar no WI-FI
2 def connect_wifi(login, password):
3     sta_if = network.WLAN(network.STA_IF)
4     sta_if.active(True)
5     sta_if.connect(login, password)
6     while not sta_if.isconnected():
7         sleep(0.1)
8 #
```

API DE LCD (lcd_api.py)

Fornece uma API para comunicação com displays LCD de caracteres compatíveis com HD44780.

https://github.com/dhylands/python_lcd/tree/master/lcd

Implementação LCD (machine_i2c_lcd.py)

Implementa um display LCD de caracteres HD44780 conectado via PCF8574 usando I2C.

https://github.com/dhylands/python_lcd/blob/master/lcd/machine_i2c_lcd.py

Integração com Beebotte

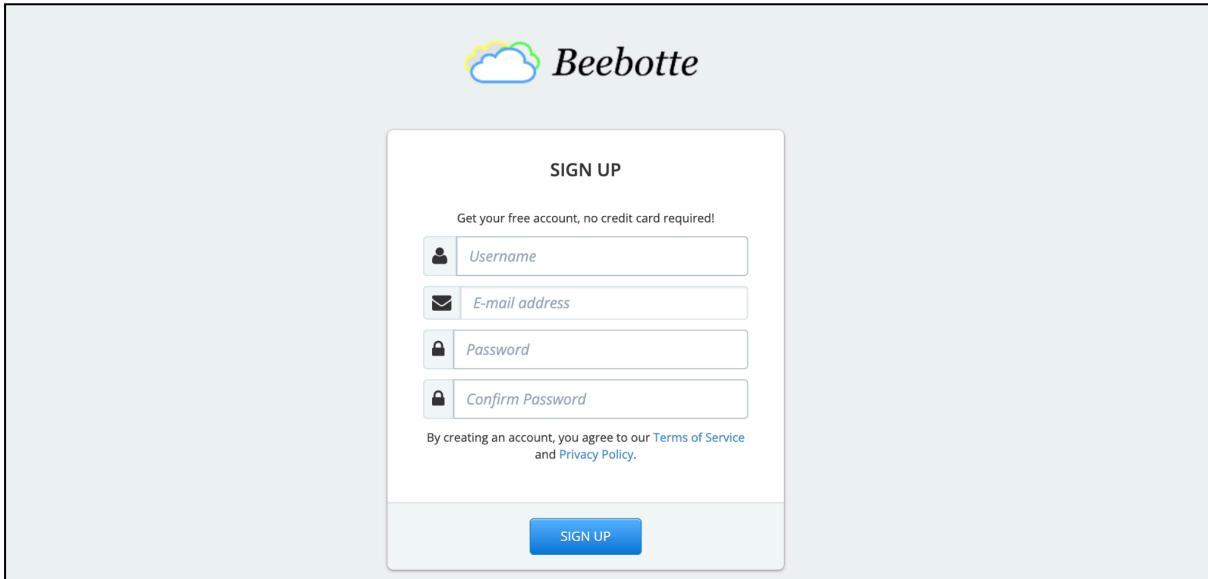
A plataforma Beebotte foi utilizada neste projeto para facilitar a coleta, monitoramento e visualização de dados em tempo real, oferecendo integração eficiente com sensores e dispositivos IoT. Com ela, foi possível configurar canais específicos para registrar informações como temperatura, umidade, localização e movimentação, além de criar um dashboard analítico dinâmico, acessível através do link

<https://beebotte.com/dash/08b66b10-9e36-11ef-9187-737958943ad4>. Este painel permite visualizar os dados coletados de forma organizada e interativa, com atualizações em tempo real, otimizando o acompanhamento e a análise dos transportes monitorados.

1. Criação da Conta

- Acesse o site [Beebotte](#) e realize o cadastro utilizando um e-mail válido e uma senha segura.
- Após confirmar o cadastro, inicie a configuração do projeto criando um canal. Dê um nome significativo ao canal e adicione uma breve descrição que explique sua finalidade.

- Em seguida, configure os recursos associados ao canal. Cada recurso deve corresponder a um tipo de dado que será monitorado, como Temperatura e Umidade. Para que esses dados sejam acessíveis, configure o canal como público, se necessário.



2. Criação e Configuração de Canais

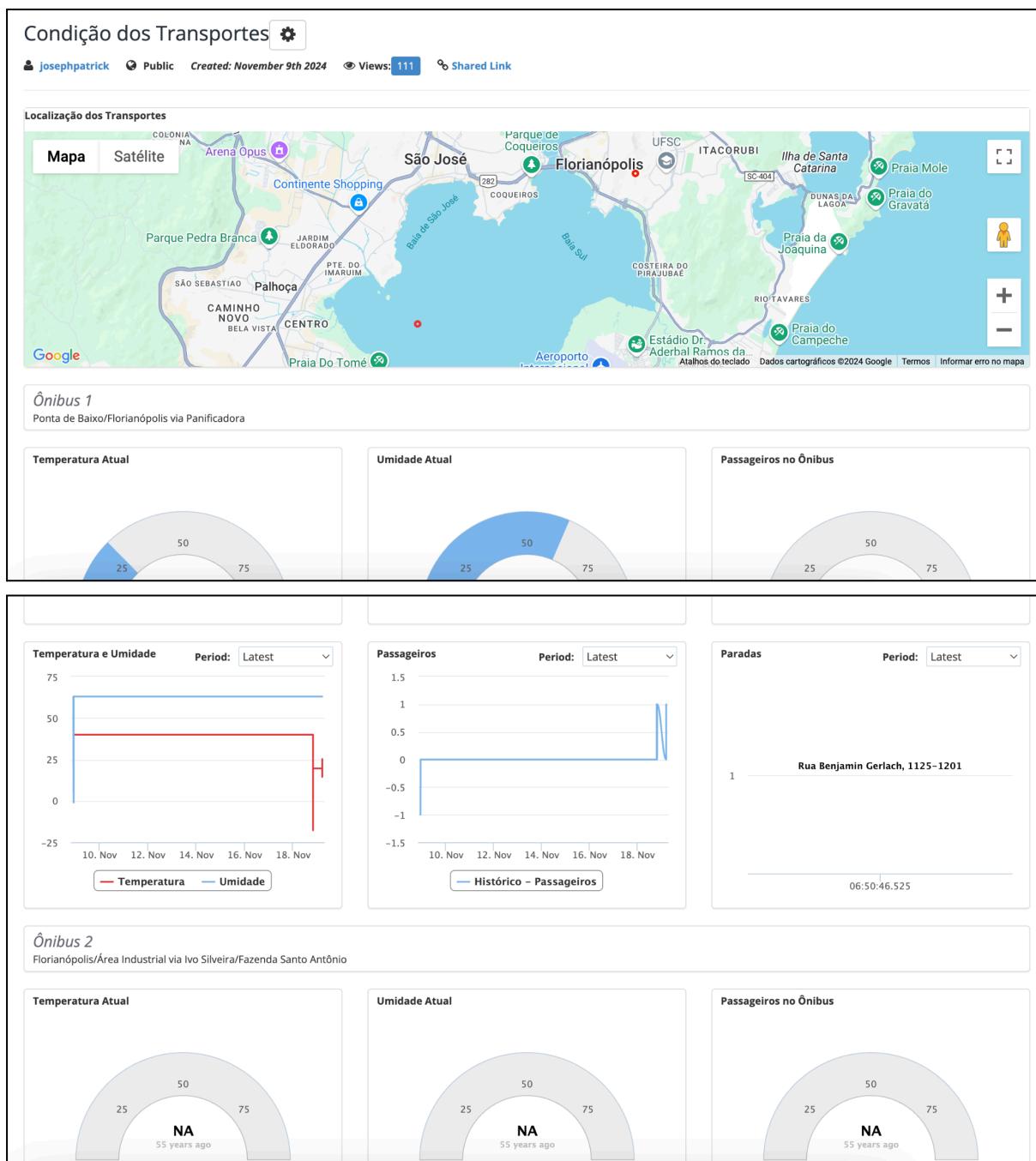
- Os canais no Beebotte funcionam como contêineres para dados específicos. No contexto deste projeto, foi criado um canal para cada transporte ou dispositivo monitorado.

3. Parâmetros dos Recursos

- Cada recurso foi configurado com parâmetros específicos, como:
 - Tipo de dado (numérico, string, booleano, etc.).

4. Criação de Dashboards em Tempo Real

- O Beebotte oferece uma funcionalidade robusta para criar dashboards personalizados.
- No projeto, foi configurado um dashboard analítico que apresenta as informações coletadas de maneira gráfica e interativa.
- Os principais destaques do dashboard incluem:
 - Visualização em tempo real dos dados coletados.
 - Gráficos e métricas organizados por canal, permitindo o monitoramento individual de cada transporte/dispositivo.
 - Possibilidade de filtrar e detalhar os dados por período, facilitando a análise histórica e preditiva.

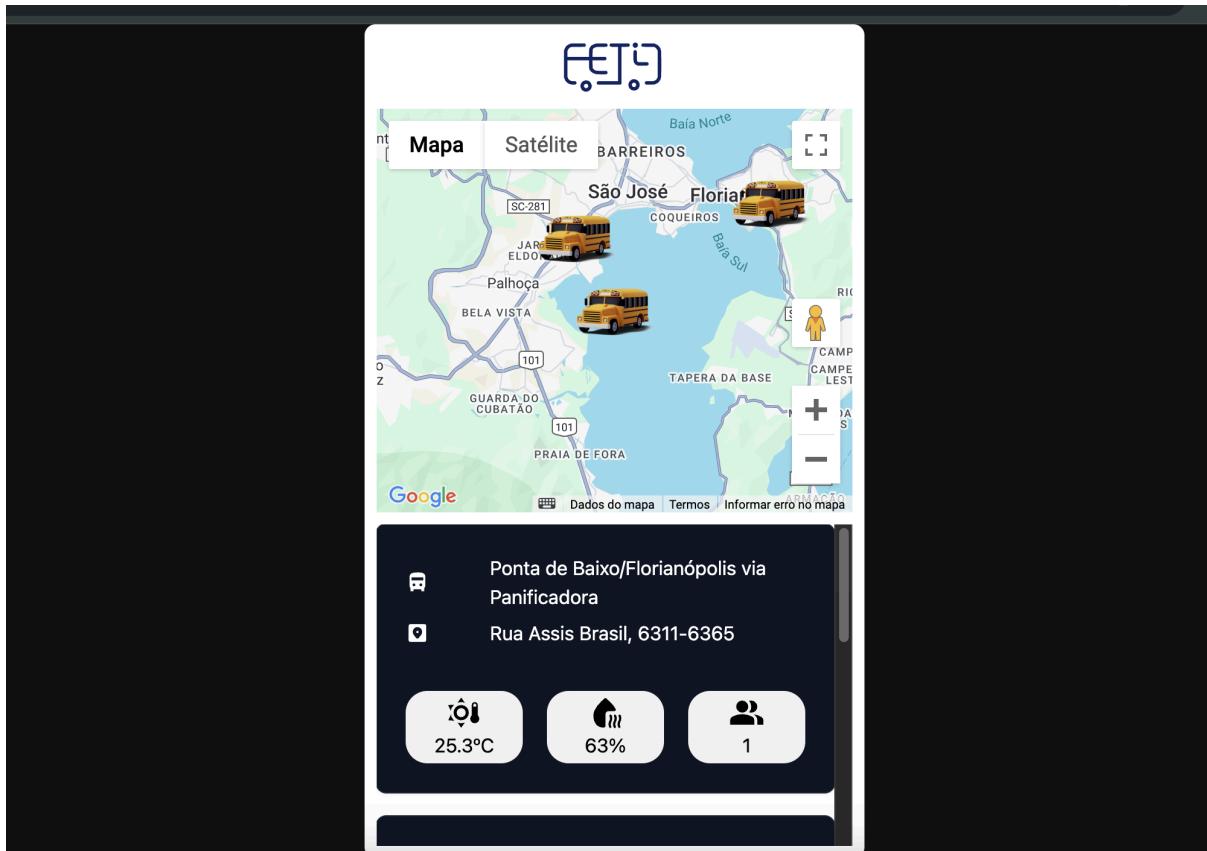


Essa integração proporciona uma solução eficiente e acessível para monitoramento contínuo, tornando o Beebotte uma escolha ideal para projetos que exigem visualização e análise de dados em tempo real.

Aplicativo para o usuário final

Na etapa final do projeto, foi desenvolvido um aplicativo web (roda através do navegador) com o objetivo de fornecer informações em tempo real sobre o transporte público. A aplicação permite que os usuários acessem dados como localização, temperatura e lotação dos veículos, oferecendo uma experiência prática e informativa.

Visual do Aplicativo



Linguagem e tecnologias utilizadas

O aplicativo em questão foi desenvolvido utilizando o Nuxt, um framework baseado em Vue.js que facilita a criação de aplicações web modernas e escaláveis. Além da estrutura base do Nuxt, o aplicativo integra diversas tecnologias para ampliar suas funcionalidades e interatividade. Entre elas, destaca-se a customização do Google Maps, que permite uma visualização personalizada e adaptada às necessidades dos usuários.

Além disso, o aplicativo realiza uma conexão com o Beebotte através de API Web, proporcionando comunicação para o monitoramento dos dados coletados. Essa integração permite que o sistema atualize automaticamente informações críticas, como localização, temperatura e lotação, melhorando a experiência dos usuários.

Simulação de viagens do GPS (Backend)

Devido à incapacidade técnica do Wokwi de realizar simulações de GPS com coordenadas reais, utilizamos dados fornecidos pela Jotur para simular as rotas de viagem. Para reproduzir o deslocamento dos ônibus de maneira realista, foram implementados scripts que calculam a localização dos veículos com base em parâmetros de tempo e nas informações das paradas ao longo do trajeto.

Esses scripts permitiram simular o movimento dos ônibus entre as paradas, oferecendo uma representação precisa do deslocamento. Assim, foi possível realizar uma simulação mais próxima da realidade, que integra o monitoramento de localização em tempo real, mesmo com as limitações da plataforma Wokwi.

getCurrentStop.ts

Código utilizado para determinar a parada, através dos dados de paradas e o tempo decorrido.

```
● ● ●

1 import { getTripData } from "./getTripData";
2 export function getCurrentStop(tripData: ReturnType<typeof getTripData>[0], elapsedTime: number) {
3     let totalTimePassed = 0;
4     for (let i = 0; i < tripData.stops.length; i++) {
5         // Adiciona o tempo da parada atual ao tempo total acumulado
6         totalTimePassed += tripData.stops[i].time;
7
8         // Se o tempo passado já ultrapassa ou iguala o elapsedTime
9         if (totalTimePassed >= elapsedTime) {
10             // Verifica se o ônibus está exatamente em uma parada ou entre duas paradas
11             if (totalTimePassed === elapsedTime) {
12                 return {
13                     status: "On stop",
14                     stop: tripData.stops[i]
15                 };
16             } else {
17                 return {
18                     status: "Between stops",
19                     from: tripData.stops[i - 1],
20                     to: tripData.stops[i]
21                 };
22             }
23         }
24     }
25
26     // Caso o tempo ultrapasse o último ponto da viagem, considera a viagem concluída
27     return {
28         status: "Trip completed",
29         stop: tripData.stops[tripData.stops.length - 1]
30     }
31 }
```

getInterpolatedCoordinates.ts

Retornar latitude e longitude da parada de ônibus. Em caso de o transporte não estiver em uma determinada parada, realiza uma interpolação linear, sugerindo um local entre uma parada e outra

```
 1 import { getCurrentStop } from "./getCurrentStop";
2 import type { getTripData } from "./getTripData";
3
4 export const getInterpolatedCoordinates = (tripData: ReturnType<typeof getTripData>[0], elapsedTime: number) => {
5   const stop = getCurrentStop(tripData, elapsedTime);
6   if(stop.status === "On stop" || stop.status === "Trip completed") {
7     return {
8       lat: stop?.stop?.lat,
9       lng: stop?.stop?.lon,
10      done: stop.status === "Trip completed",
11      stopName: stop?.stop?.stopName
12    };
13  }
14  else {
15    const stopRecalculate = stop as { from: typeof tripData.stops[0], to: typeof tripData.stops[0] };
16    // interpolação linear
17    const timeDiff = elapsedTime - (stopRecalculate.from.time + stopRecalculate.to.time);
18    const timeTotal = stopRecalculate.to.time - stopRecalculate.from.time;
19    const lat = stopRecalculate.from.lat + (stopRecalculate.to.lat - stopRecalculate.from.lat) * (timeDiff / timeTotal);
20    const lon = stopRecalculate.from.lon + (stopRecalculate.to.lon - stopRecalculate.from.lon) * (timeDiff / timeTotal);
21    return {
22      lat: lat,
23      lng: lon,
24      done: false,
25      stopName: stop?.stop?.stopName
26    };
27  }
28 }
```

getTripData.ts

Código com dados fornecidos pela Jotur contendo informações sobre paradas de determinados ônibus

```
1  export const getTripData = () => [
2    {
3      tripId: 13601,
4      tripName: "Ponta de Baixo/Florianópolis via Panificadora",
5      totalTime: 2471,
6      channelToken: "token_N973lxxoX1QXEYnk",
7      stops: [
8        {
9          reference: 0,
10         stopId: 21346,
11         dist: 0,
12         lon: -48.63789995999998600324033759534358978271484375,
13         time: 0,
14         lat: -27.629029549999984940586728043854236602783203125,
15         stopName: "Ponto inicial/final na Curva do Cego",
16       },
17       {
18         reference: 0,
19         stopId: 21347,
20         dist: 0.204999999999999877875467291232780553400516510009765625,
21         lon: -48.63826482999997118648025207221508026123046875,
22         time: 35,
23         lat: -27.63062973999999827659848961047828197479248046875,
24         stopName: "Rua Frederico Afonso, 1142-1158",
25       },
26     // Continua
```

getMqttClient.ts

Obter o cliente MQTT, com objetivo de atualizar a localização do transporte dentro do canal MQTT.

```
1 import mqtt from "mqtt";
2 export const getMqttClient = () => {
3     return new Promise<mqtt.MqttClient>((resolve, reject) => {
4         const MQTT_CLIENT_ID = "ofdrsXZPBNw6xF7acIfmlmQq4sGNKwNC";
5         const MQTT_BROKER = "mqtts://mqtt.beebotte.com";
6         const MQTT_USER = "ofdrsXZPBNw6xF7acIfmlmQq4sGNKwNC";
7         const MQTT_PASSWORD = "ofdrsXZPBNw6xF7acIfmlmQq4sGNKwNC";
8         const MQTT_PORT = 8883;
9         const client = mqtt.connect(MQTT_BROKER, {
10             clientId: MQTT_CLIENT_ID,
11             username: MQTT_USER,
12             password: MQTT_PASSWORD,
13             port: MQTT_PORT,
14         });
15         client.on("connect", () => {
16             console.log("Connected to MQTT Broker");
17             resolve(client);
18         });
19         client.on("error", (error) => {
20             console.log("Error:", error);
21             reject(error);
22         });
23     });
24 };
25 export default getMqttClient;
26
```

api/trip/start/[id].ts

API Endpoint para iniciar a simulação de uma corrida de ônibus e atualizar dados de gps dentro do Beebotte

```
● ● ●

1  export default defineEventHandler(async (event) => {
2    const tripId = getRouterParam(event, "id");
3    if (!tripId) {
4      throw new Error("ID não encontrado");
5    }
6    const trip = getTripData().find((trip) => {
7      return String(trip.tripId) === String(tripId);
8    });
9    if (!trip) {
10      throw new Error("Trip não foi encontrada");
11    }
12
13    const channelId = `trip${tripId}`;
14    const MqttTopicGps = `${channelId}/gps`;
15    const MqttTopicStopName = `${channelId}/stop_name`;
16
17    const mqttClient = await getMqttClient();
18    let totalTimePassed = 0;
19    setInterval(() => {
20      const coordinates = getInterpolatedCoordinates(trip, totalTimePassed);
21      totalTimePassed += 1;
22      const dataToSend = {
23        latitude: coordinates.lat,
24        longitude: coordinates.lng,
25      };
26      mqttClient.publish(
27        MqttTopicGps,
28        JSON.stringify({
29          data: dataToSend,
30          write: true,
31        })
32      );
33      mqttClient.publish(
34        MqttTopicStopName,
35        JSON.stringify({
36          data: coordinates.stopName,
37          write: true,
38        })
39      );
40    }, 1000);
41    return getInterpolatedCoordinates(trip, totalTimePassed);
42  });
43
```

trips.ts

API Endpoint para listar os transportes públicos disponíveis

```
1 import getField from "~/server/utils/getField";
2
3 export default defineEventHandler(async (event) => {
4     let tripsData = getTripData();
5     let trips: Array<{
6         tripId: number;
7         tripName: string;
8         totalTime: number;
9         channelToken: string;
10        temperature: number,
11        humidity: number,
12        gps: {
13            latitude: number;
14            longitude: number;
15        };
16        personsIn: number;
17        personsOut: number;
18        personsTotal: number;
19        stopName: string;
20        stops: typeof tripsData[0]["stops"]
21    }> = [];
22    for (let trip of tripsData) {
23        const { tripId, tripName, totalTime, channelToken, stops } = trip;
24        trips.push({
25            tripId,
26            tripName,
27            totalTime,
28            channelToken,
29            temperature: await getField(tripId, "temperature"),
30            humidity: await getField(tripId, "humidity"),
31            gps: await getField(tripId, "gps"),
32            personsIn: await getField(tripId, "persons_in"),
33            personsOut: await getField(tripId, "persons_out"),
34            personsTotal: await getField(tripId, "persons_total"),
35            stopName: await getField(tripId, "stop_name"),
36            stops
37        });
38    }
39    return trips;
40 });
41
```

Código de Apresentação do APP (Frontend)

stores/trip.ts

Consumir dados de transportes

```
● ● ●

1  export const useTripsStore = defineStore("trips-store", () => {
2    const {
3      data: tripsData,
4      status: tripsStatus,
5      refresh: tripsRefresh,
6    } = useFetch("/api/trips");
7    return {
8      tripsData,
9      tripsStatus,
10     tripsRefresh,
11   };
12 });
13
```

components/map.vue

Exibir um Mapa

```
1 <script setup lang="ts">
2 import { GoogleMap, CustomMarker } from "vue3-google-map";
3 const tripsStore = useTripsStore();
4 const {tripsRefresh} = tripsStore;
5 const { tripsData } = storeToRefs(tripsStore);
6 const center = computed(() => {
7   return {
8     lat: tripsData.value?.[0].gps.latitude,
9     lng: tripsData.value?.[0].gps.longitude,
10    };
11  });
12 let interval: string | number | NodeJS.Timeout | undefined;
13 onMounted(() => {
14   console.log("refreshing...")
15   interval = setInterval(() => {
16     tripsRefresh();
17   }, 8000);
18 });
19 onUnmounted(() => {
20   if(interval) {
21     clearInterval(interval);
22   }
23 })
24 </script>
25 <template>
26   <GoogleMap
27     api-key="AIzaSyAxZG8HgntTJzBsSXF3oW_sThwP-fyL9P4"
28     style="width: 100%; height: 100%"
29     :center="center"
30     :zoom="11"
31   >
32     <CustomMarker
33       v-for="trip of tripsData"
34       :key="trip.tripId"
35       :options="{
36         position: {
37           lat: trip.gps.latitude,
38           lng: trip.gps.longitude,
39         },
40       }"
41     >
42       
43     </CustomMarker>
44   </GoogleMap>
45 </template>
46
```

Página Principal (Script)

Consumir dados de transportes

```
1 <script setup lang="ts">
2 const tripStore = useTripsStore();
3 const {tripsRefresh} = tripStore;
4 const { tripsData, tripsStatus } = storeToRefs(tripStore);
5 let interval: string | number | NodeJS.Timeout | undefined;
6 onMounted(() => {
7   interval = setInterval(() => {
8     if(tripsStatus.value == 'success' || tripsStatus.value == 'error') {
9       tripsRefresh();
10    }
11  }, 1000)
12 })
13 onUnmounted(() => {
14   clearInterval(interval)
15 })
16 </script>
17
```

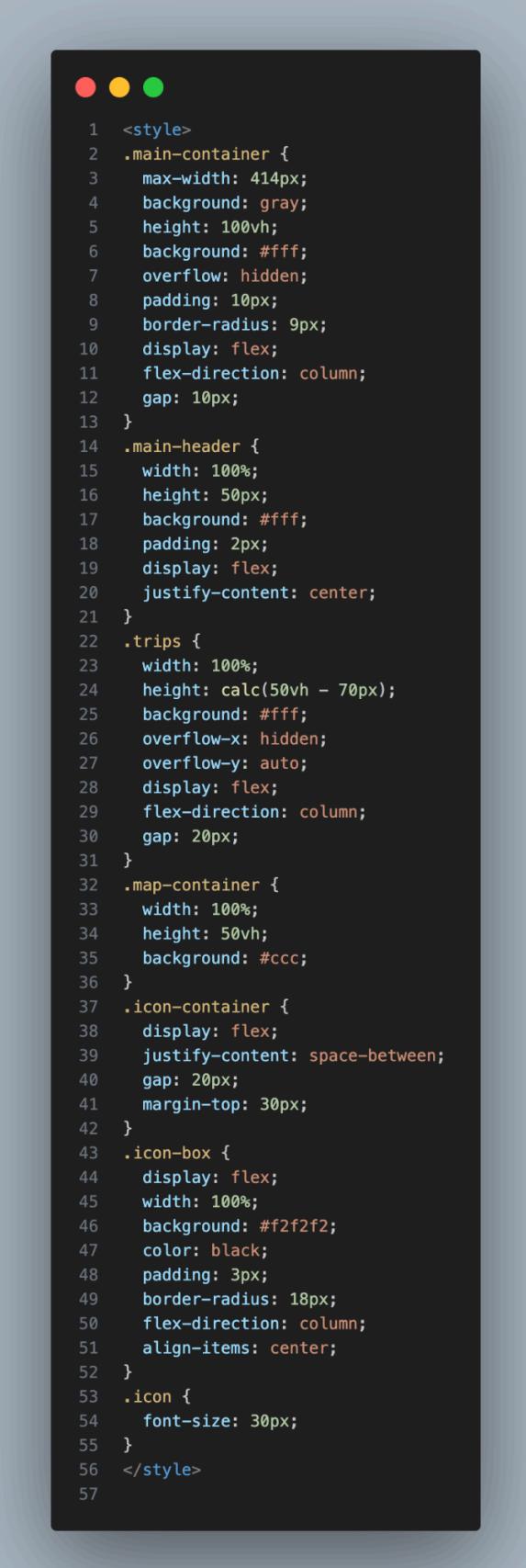
Página Principal (Template)

Consumir dados de transportes

```
1  <template>
2    <UContainer class="main-container" v-if="tripsData?.length && tripsData.length > 0">
3      <div class="main-header">
4        
5      </div>
6      <div class="map-container">
7        <ClientOnly>
8          <Map></Map>
9        </ClientOnly>
10      </div>
11      <div class="trips">
12        <UCard v-for="trip of tripsData">
13          <div style="display: flex; align-items: center; gap: 50px">
14            <UIcon
15              name="i-material-symbols:directions-bus-sharp"
16              class="w-9 h-9"
17              style="width: 20px"
18            />
19            <div style="width: 80%">
20              <h3>{{ trip.tripName }}</h3>
21            </div>
22          </div>
23
24          <div style="display: flex; align-items: center; gap: 50px">
25            <UIcon
26              name="i-material-symbols:file-map-rounded"
27              class="w-9 h-9"
28              style="width: 20px"
29            />
30            <div style="width: 80%">
31              <h3>{{ trip.stopName }}</h3>
32            </div>
33          </div>
34
35          <div class="icon-container">
36            <div class="icon-box">
37              <UIcon name="mdi:sun-thermometer-outline" class="icon" />
38              {{trip.temperature}}°C
39            </div>
40            <div class="icon-box">
41              <UIcon name="material-symbols:water-voc-rounded" class="icon" />
42              {{trip.humidity}}%
43            </div>
44            <div class="icon-box">
45              <UIcon name="material-symbols:group" class="icon" />
46              {{trip.personsTotal}}
47            </div>
48          </div>
49        </UCard>
50      </div>
51    </UContainer>
52    <div v-else>
53      Carregando...
54    </div>
55  </template>
```

Página Principal (Estilos)

Consumir dados de transportes



The screenshot shows a mobile application interface. At the top, there are three circular icons: red, yellow, and green. Below them is a dark header bar. The main content area has a light gray background. It displays flight trip information: "Flight 123456789", "From: São Paulo (GRU) To: Rio de Janeiro (GIG)", "Arrival: 2023-09-15 10:00", and "Departure: 2023-09-15 12:00". Below this, there's a section titled "Flight Details" with items like "Flight Number: 123456789", "Airline: TAM", "Status: On Time", and "Flight Type: Economy". A large blue button at the bottom says "Check-in". The bottom of the screen features a navigation bar with icons for home, search, and notifications.

```
1 <style>
2 .main-container {
3   max-width: 414px;
4   background: gray;
5   height: 100vh;
6   background: #fff;
7   overflow: hidden;
8   padding: 10px;
9   border-radius: 9px;
10  display: flex;
11  flex-direction: column;
12  gap: 10px;
13 }
14 .main-header {
15   width: 100%;
16   height: 50px;
17   background: #fff;
18   padding: 2px;
19   display: flex;
20   justify-content: center;
21 }
22 .trips {
23   width: 100%;
24   height: calc(50vh - 70px);
25   background: #fff;
26   overflow-x: hidden;
27   overflow-y: auto;
28   display: flex;
29   flex-direction: column;
30   gap: 20px;
31 }
32 .map-container {
33   width: 100%;
34   height: 50vh;
35   background: #ccc;
36 }
37 .icon-container {
38   display: flex;
39   justify-content: space-between;
40   gap: 20px;
41   margin-top: 30px;
42 }
43 .icon-box {
44   display: flex;
45   width: 100%;
46   background: #f2f2f2;
47   color: black;
48   padding: 3px;
49   border-radius: 18px;
50   flex-direction: column;
51   align-items: center;
52 }
53 .icon {
54   font-size: 30px;
55 }
56 </style>
57
```

3. ENCERRAMENTO DO PROJETO

3.1. Relato Coletivo:

O desenvolvimento do projeto proporcionou ao grupo uma experiência enriquecedora, tanto do ponto de vista técnico quanto humano. Conseguimos alcançar os objetivos sociocomunitários estabelecidos, especialmente no que se refere à integração de tecnologias IoT para monitoramento e controle em tempo real, atendendo às demandas de inovação e funcionalidade.

Apesar dos desafios técnicos e logísticos, a colaboração foi essencial para superar obstáculos e alinhar expectativas, garantindo a entrega de um sistema que reflete os valores de conectividade e eficiência propostos inicialmente.

3.2. Relato de Experiência Individual

Igor Gomes

Contextualização

Durante o desenvolvimento deste projeto, adquiri conhecimentos fundamentais sobre a aplicação de sensores IoT, especificamente para monitoramento de temperatura, passando por desafios técnicos que me ensinaram sobre eletrônica prática e programação. Além disso, a experiência de trabalhar em equipe reforçou minha habilidade de comunicação e organização, pois tivemos que dividir tarefas e alinhar expectativas para garantir a execução eficiente de cada etapa.

Considerações Finais

Tive uma vivência voltado para a parte de conectividade e integração do sistema com a plataforma Wokwi, o que representou um grande desafio de aprendizado, me agregando com os protocolos de comunicação, especialmente no MQTT, para enviar dados do ESP - 32 para o Arduino de forma eficiente e estável. A minha contribuição foi vital para o sucesso do sistema de monitorar em tempo real, o que lhe trouxe uma sensação de realização ao ver a integração funcionando sem falhas.

Joseph Patrick

Contextualização

Minha experiência no projeto foi enriquecedora e desafiadora, especialmente devido ao trabalho em equipe e à complexidade técnica envolvida. Como participante, eu assumi responsabilidades no desenvolvimento técnico, contribuindo principalmente nas etapas de programação.

Metodologia

O projeto foi desenvolvido ao longo de várias semanas, passando por etapas como pesquisa inicial, configuração de dispositivos IoT, programação em Python e testes práticos. Apesar de imprevistos, conseguimos adaptar-nos e entregar o projeto final. A maior parte do desenvolvimento foi realizada na Estácio, com complemento de tarefas em casa e reuniões via Discord para alinhar e compartilhar os avanços do projeto.

Resultados

Minhas expectativas foram atendidas e, em alguns aspectos, superadas. A experiência prática de desenvolver um dispositivo IoT foi muito satisfatória, embora o processo tenha trazido desafios técnicos, especialmente na etapa final. Uma das maiores dificuldades foi garantir a funcionalidade dos dispositivos, já que tive problemas com algumas plataformas, exigindo adaptações e o uso de alternativas.

Reflexão Aprofundada

Apesar de não conseguir apresentar o projeto no dia final, fiquei extremamente satisfeito com o resultado, dado a nova oportunidade de finalizá-lo. Os imprevistos serviram como aprendizado valioso e a experiência foi de grande valor para meu crescimento, o que me surpreendeu. Além disso, pude colaborar com outras equipes, contribuindo para resolver problemas nos quais eu tinha mais facilidade.

Considerações Finais

Essa experiência abriu portas para aprimorar ainda mais o projeto e expandi-lo além do ambiente acadêmico, com potencial até para transformá-lo em um produto real. Essa vivência despertou em mim um interesse maior por IoT e motivou-me a aprofundar meus conhecimentos nessa área, buscando também futuras oportunidades profissionais e projetos no setor.