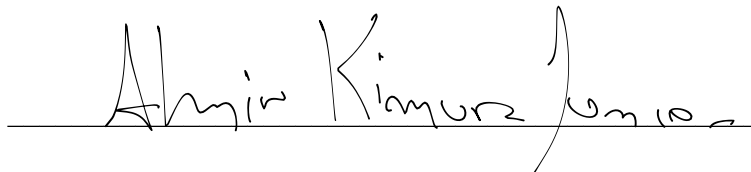


FELIPE CORREA DA SILVA

**SISTEMA INTELIGENTE INTEGRADO DE PROTEÇÃO E DETECÇÃO DE
ACIDENTES DE MOTOCICLISTAS E SEUS VEÍCULOS.**

Orientador: Prof. Dr. Almir Kimura Junior

A handwritten signature in black ink, reading "Almir Kimura Junior", is written over a horizontal line. The signature is fluid and cursive, with the last name "Junior" written in a smaller, more compact script.

Manaus

2023

Universidade do Estado do Amazonas – UEA
Escola Superior de Tecnologia – EST

Reitor:

André Luiz Nunes Zogahib

Vice-Reitor:

Kátia do Nascimento Couceiro

Diretora da Escola Superior de Tecnologia:

Ingrid Sammyne Gadelha Figueiredo

Coordenador do Curso de Engenharia Elétrica:

Israel Gondres Torné

Banca Avaliadora composta por: Data da defesa: 29/03/2023.

Prof. Almir Kimura Junior, Dr (Orientador)

Prof. Fábio de Sousa Cardoso, Dr

Prof. Israel Gondres Torné, Dr

CIP – Catalogação na Publicação

Da Silva, Felipe Correa

Sistema inteligente integrado de proteção e detecção de acidentes de motociclistas e seus veículos; [orientado por] Almir Kimura Junior – Manaus: 2023.

117 p.: il.

Trabalho de Conclusão de Curso (Graduação em Engenharia Elétrica).
Universidade do Estado do Amazonas, 2023.

1. Detecção de acidentes. 2. GPS. 3. HTTP. 4. Leitor biométrico. 5. ESP32. 6. Aplicações. I. Junior, Almir Kimura.

FELIPE CORREA DA SILVA

**SISTEMA INTELIGENTE INTEGRADO DE PROTEÇÃO E DETECÇÃO DE
ACIDENTES DE MOTOCICLISTAS E SEUS VEÍCULOS.**

Projeto de pesquisa desenvolvido durante a disciplina de Trabalho de Conclusão de Curso I e apresentado à banca avaliadora do Curso de Engenharia Elétrica da Escola Superior de Tecnologia da Universidade do Estado do Amazonas, como pré-requisito para a obtenção do título de Engenheiro Eletricista.

Nota obtida: 10 (Dez)

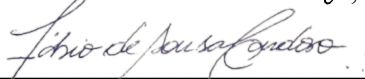
Aprovada em 29/03/23.

Área de concentração: Programação de microcontroladores.

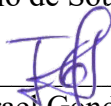
BANCA EXAMINADORA



Orientador: Almir Kimura Junior, Dr.



Avaliador: Fábio de Sousa Cardoso, Dr.



Avaliador: Israel Gondres Torné, Dr.

Manaus 2023

AGRADECIMENTOS

Dedico esse trabalho a Deus primeiramente, por toda a força e saúde que ele me deu. Aos meus pais Francisco Fernando e Maria de Jesus, pelo amor e apoio em todos os momentos da minha vida, sendo eles difíceis ou não, sempre priorizando a minha saúde e felicidade. Ao meu amor Ana Beatriz por toda a ajuda, incentivo e compreensão. Aos meus amigos de faculdade que foram incríveis, especialmente ao meu amigo Carlos Emanuel pela grande ajuda com os projetos que tive a oportunidade de trabalhar em equipe. Ao meu amigo Vitor Ugo que sempre me ajudou em todas as situações ocorridas durante nossa amizade. Aos meus amigos do OCEAN, HUB e STEM, que me auxiliaram bastante nesse projeto, além de deixarem o ambiente de trabalho mais alegre. Aos professores que fizeram parte do meu crescimento acadêmico e profissional, especialmente o meu coordenador Israel Gondres e o professor Fábio Cardoso. Ao meu orientador, professor Almir Kimura Junior, pela atenção, supervisão e acompanhamento em busca do melhor trabalho possível, prezando ainda mais minha responsabilidade com a excelência e evoluindo meu caráter profissional, característica que vou levar para o resto da vida. A UEA, que foi uma instituição que possibilitou grandes aprendizados e muitas experiências. A minha família de Manaus e Santarém que sempre torceram pelo meu sucesso e minha saúde para conquistar meus objetivos de vida.

O verdadeiro TCC são os amigos que fizemos na jornada...

RESUMO

Este projeto foi desenvolvido como um sistema que garante a segurança de motociclistas e de seus veículos por meio de tecnologias de sistemas embarcados e internet das coisas. O sistema foi composto por duas vertentes: sistema antifurto, para segurança do veículo, que libera a ignição somente com a digital biométrica de pessoas autorizadas; e sistema de análise de sinais vitais, no qual, por meio da conectividade de um aparelho *smartwatch* ao sistema acoplado, tem acesso as aferições fisiológicas e níveis de consciência do condutor em caso de acidentes, acrescentados a outras informações do agravamento do acidente, em que esses dados serão encaminhados para entidades de socorro imediato ou algum contato de emergência, em caso de acidentes para uma interface *web* ou *desktop* desses usuários. Ambos os sistemas foram conectados a um aplicativo de gerenciamento de pessoas cadastradas, onde o proprietário pode adicionar ou deletar cadastros no sistema, além verificações das funcionalidades do equipamento instalado no veículo. O sistema também conta com protocolos de bloqueios funcionais da motocicleta, caso esta tenha sido furtada ou roubada, tanto via aplicativo quanto outra digital biométrica cadastrada para esse fim, na qual aciona de forma temporária o veículo para uma distância segura entre o criminoso e a vítima. Além de possuir uma verificação de localização remota acoplada ao sistema para futura recuperação do veículo que também pode ser vista pelo aplicativo web ou mobile que o proprietário tenha acesso. O Sistema foi aplicado e instalado em um veículo real de duas rodas para os devidos testes para obtenção de dados e validação das funcionalidades do projeto. Assim sendo, foi obtido êxito no funcionamento do projeto desenvolvido.

Palavras-chave: Anti-roubo, Detecção de acidente, GPS, HTTP, Leitor biométrico, ESP32, Aplicações.

ABSTRACT

This project was developed as a system to ensure the safety of motorcyclists and their vehicles through embedded systems technologies and the Internet of Things. The system was composed of two components: Anti-theft system, for vehicle security, which releases the ignition only with the biometric fingerprint of authorized persons; and Vital signs analysis system, in which, through the connectivity of a smartwatch device to the coupled system, has access to physiological measurements and levels of consciousness of the driver in case of accidents, added to other information of the worsening of the accident, in which these data will be forwarded to immediate rescue entities or some emergency contact, in case of accidents to a web or desktop interface of these users. Both systems were connected to a management application of registered people, where the owner can add or delete registrations in the system, besides verifying the functionalities of the equipment installed in the vehicle. The system also has protocols for the functional blocking of the motorcycle, in case it has been stolen, both via the application and another biometric fingerprint registered for this purpose, which temporarily drives the vehicle to a safe distance between the criminal and the victim. Besides having a remote location verification system attached to the system for future recovery of the vehicle that can also be seen by the web or mobile application that the owner has access. The system was applied and installed in a real two-wheeled vehicle for the necessary tests to obtain data and validate the project's functionalities. Therefore, it was successful in the operation of the developed project.

Keywords: Anti-theft, Accident detection, GPS, HTTP, Biometric reader, ESP32, Applications.

LISTA DE ABREVIATURAS E SIGLAS

GPS – Global Positioning System

IOT – Internet of Things(Internet das coisas)

I/O – Input/Output

PCA – Principal Components Analysis

RAM – Random Access Memory

SVM – Support Vector Machine

LAN – Local Area Network

API – Application Programming Interface

PCB – Printed Circuit Board

URL – Uniform Resource Locator

JSON– Javascript object notation

CNC - Controle Numérico Computadorizado

UART - transmissor/receptor assíncrono universal

LISTA DE FIGURAS

Figura 1: Porcentagem de indenizações por tipos de veículos.	18
Figura 2: Número de roubos e furtos de motos em São Paulo.	19
Figura 3: Esp 32.	22
Figura 4: Kodular.	23
Figura 5: Módulo relé.	25
Figura 6: Módulo GPS.	28
Figura 7: Diagrama de funcionamento do projeto.	35
Figura 8: Funcionamento da detecção de acidente inteligente.	36
Figura 9: Funcionamento do sistema de autenticação por leitura biométrica. ...	37
Figura 10: Fluxo de páginas do aplicativo WEB.	38
Figura 11: Esboço da página de Login.	38
Figura 12: Esboço da página de chamados.	39
Figura 13: Interface inicial do Bubble.	39
Figura 14: Identificação do projeto web.	40
Figura 15: Ícone da aplicação do projeto.	40
Figura 16: Alteração do ícone padrão do aplicativo.	41
Figura 17: Upload do ícone para o aplicativo.	41
Figura 18: Definição de todas as fontes do aplicativo.	42
Figura 19: Página de adição de extensões.	42
Figura 20: Interface de desenvolvimento do bubble.	43
Figura 21: Escolha do formato de cor de fundo aplicativo.	43
Figura 22: Plataforma do uigradients.	44
Figura 23: Configuração das cores gradientes do site.	44
Figura 24: Botão de pré-visualização.	45
Figura 25: Emulação do aplicativo em tempo real.	45
Figura 26: Dimensionamento da tela do aplicativo.	46
Figura 27: Tela de login e cadastro do aplicativo.	46
Figura 28: Minitela de cadastro.	47
Figura 29: Minitela de login.	47
Figura 30: Campo de usuário criado.	48
Figura 31: Teste de cadastro.	48
Figura 32: Validação de cadastro de usuários.	49

Figura 33: Configuração de extensões do bubble.....	49
Figura 34: Esboço das telas do aplicativo mobile.	50
Figura 35: Interface da plataforma Kodular.	51
Figura 36: Páginas de blocos de programação.	51
Figura 37: Tela de login do aplicativo mobile.....	52
Figura 38: Programação da página de login.	52
Figura 39: Programação dos comandos de envios dos usuários.....	53
Figura 40: Programação do sistema de notificação por voz.	54
Figura 41: Iniciar uma nova planilha google.....	54
Figura 42: Criação de uma planilha no google sheets.	55
Figura 43: Criação de uma nova aba da planilha.....	55
Figura 44: Identificação da coluna de informações manipuladas.....	56
Figura 45: Identificação da aba correspondentes aos dados trabalhados.	56
Figura 46: Configuração de segurança para a planilha.....	57
Figura 47: Configuração da planilha no formato restrito.	57
Figura 48: Ilustração do funcionamento de APIs.	58
Figura 49: Interface da plataforma de banco de dados.....	58
Figura 50: Campo para anexar a URL da planilha que será utilizada.	59
Figura 51: Dashboard da plataforma para configurações de comunicação.	59
Figura 52: Autenticação básica HTTP do sistema.....	61
Figura 53: URLs de requisição do sistema.....	61
Figura 54: Função de GET habilitada.....	62
Figura 55: Identificação padrão da API.....	62
Figura 56: Configuração de autenticação da API do bubble.	63
Figura 57: Configuração de parâmetros de chamadas.....	63
Figura 58: Recebimento em formato JSON dos dados da planilha.	64
Figura 59: Formatação de dados para localização do google.....	64
Figura 60: Texto dinâmico do acidente.	65
Figura 61: Criação de mapa para amostragem do local de chamado.	66
Figura 62: Implantação de métodos de adição de dados.	66
Figura 63: URL de implantação de dados.	67
Figura 64: Configuração da programação do aplicativo para envio de dados....	68
Figura 65: Diagrama de blocos da arquitetura do circuito.	68
Figura 66: Esquema elétrico do projeto.....	69

Figura 67: Designe da placa de circuito impresso contendo as trilhas e furos...	70
Figura 68: Imagem da placa em formato Bitmap.	70
Figura 69: Imagem dos pontos de furos em formato bitmap.	71
Figura 70: Configuração do arquivo gcode no Aspire.	71
Figura 71: Placa de fenolite padrão cobreada.	72
Figura 72: Manufatura da placa de circuito impresso na máquina de CNC.	73
Figura 73: Placa de fenolite depois do processo de manufatura da placa CNC.	73
Figura 74: Teste de condutividade das trilhas da placa.	74
Figura 75: Soldagem dos headers na placa manufaturada.	74
Figura 76: Placa manufaturada sem conectores.	75
Figura 77: Placa manufaturada com conectores.	75
Figura 78: Placa de circuito impresso finalizada.	76
Figura 79: Módulo biométrico R307.	77
Figura 80: Modelagem do suporte para o módulo leitor biométrico.	77
Figura 81: Placa de circuito impresso no Fusion.	78
Figura 82: <i>Case</i> lateral da placa de circuito impresso.	78
Figura 83: <i>Case</i> fechado com tampa.	79
Figura 84: <i>Case</i> principal com placa de alimentação na parte superior.	80
Figura 85: Fixação dos dois <i>cases</i> .	80
Figura 86: Manufatura das peças modeladas na impressora 3D.	81
Figura 87: <i>Cases</i> da placa principal de fonte regulável de tensão.	81
Figura 88: Suporte protetor do leitor biométrico.	82
Figura 89: Acoplamento de peças nos módulos.	82
Figura 90: Corte das bordas da placa manufaturada.	83
Figura 91: Programação da conexão do esp32 com a internet.	83
Figura 92: Função de conexão do esp32 com o sensor de biometria.	84
Figura 93: Comandos de funcionalidades do módulo biométrico.	84
Figura 94: Impressão da execução de funções do esp32.	85
Figura 95: <i>Firmware</i> de validação de usuários.	86
Figura 96: Código de leitura e envio dos dados do sensor giroscópio.	87
Figura 97: Envio das informações de leitura de peso.	88
Figura 98: Calibração das células de carga.	88
Figura 99: <i>Firmware</i> do módulo GPS.	89
Figura 100: Acoplamento dos sistemas em moto real.	90

Figura 101: instalação do sistema de chaveamento no relé de ignição.	90
Figura 102: Cabos de conexão manufaturados.....	91
Figura 103: Teste de conexão do microcontrolador com a internet.	92
Figura 104: teste de acesso negado de usuários não cadastrados.	92
Figura 105: Teste de acesso permitido de usuários.	93
Figura 106: Teste de cadastro de usuários.....	93
Figura 107: Telas do aplicativo desenvolvido.....	94
Figura 108: Teste de uma digital não cadastra no sistema instalado na moto.	95
Figura 109: Teste de uma digital cadastrada no sistema instalado na moto.....	95
Figura 110: Simulação de um acidente com o sistema instalado na moto.	96
Figura 111: Aplicativo mobile detectando um acidente.....	97
Figura 112: Banco de dados recebendo os dados de localização.	98
Figura 113:Aplicativo de amostras de chamados.....	98
Figura 114: Localização do acidente no google maps.....	99
Figura 115: Localização do acidente em imagem de satélite.	99

SUMÁRIO

INTRODUÇÃO	15
1 TEMA	16
2 FORMULAÇÃO DO PROBLEMA	16
3 HIPÓTESE	16
4 OBJETIVOS	17
4.1. OBJETIVO GERAL	17
4.2. OBJETIVOS ESPECÍFICOS	17
5 JUSTIFICATIVA	18
5.1. RELEVÂNCIA SOCIAL	18
5.2. RELEVÂNCIA ACADÊMICA	19
6 REFERENCIAL TEÓRICO	20
6.1. ACIDENTADO DE TRÂNSITO	20
6.2. LEIS DO CÓDIGO DE TRÂNSITO BRASILEIRO QUANTO A DISPOSITIVOS ANTIFURTOS.	20
6.3. BLOQUEIO DE VEÍCULO	21
6.4. ACESSO BIOMÉTRICO	21
6.5. ESP32	22
6.6. BUBBLE	22
6.7. KODULAR	23
6.8. SHEETY.COM	24
6.9. IOT 24	
6.10. MÓDULO RELÉ	24
6.11. SISTEMA DE IGNIÇÃO DA MOTO	25
6.12. MEDIÇÃO DE SINAIS VITAIS	27
6.13. MÓDULO GPS	27
6.14. COMUNICAÇÃO HTTP	29
6.15. CRIPTOGRAFIA RSA	30
6.16. PROTEÇÃO DE MEMÓRIA FLASH	31
6.17. DATA BASE	31
6.18. PROGRAMAÇÃO EM C E C++	32
6.19. EASYEDA	33
6.20. PROTOTIPADORA DE PLACA DE CIRCUITO IMPRESSO	33
7. METODOLOGIA	33
7.1. DIAGRAMA DE FUNCIONAMENTO DO PROJETO	34
8. IMPLEMENTAÇÃO DO PROJETO	37
8.1. <i>SOFTWARE</i> DE ALERTA DE CHAMADOS	37
8.2. DESENVOLVIMENTO APLICATIVO <i>MOBILE</i>	49
8.3. BANCO DE DADOS	54
8.4. INTEGRAÇÃO DO BANCO DE DADOS COM O APLICATIVO WEB	58
8.5. INTEGRAÇÃO DO BANCO DE DADOS COM O APLICATIVO <i>MOBILE</i>	66
8.5.1. Configuração do banco de dados	66
8.5.2. Configuração do aplicativo <i>mobile</i> para comunicação	67

8.6. DESENVOLVIMENTO DA PLACA DE CIRCUITO IMPRESSO.....	68
8.6.1. Desenvolvimento da placa no altium designer	69
8.6.2. Manufatura da placa física.....	72
8.6.3. Acoplagem de componentes na de circuito impresso	76
8.7. DESENVOLVILMENTO DE PEÇAS D3.....	76
8.7.1. Suporte para módulo biométrico	76
8.7.2. <i>Case</i> da placa de circuito impresso do projeto	77
8.7.3. <i>Case</i> da placa de fonte regulável	79
8.7.4. Impressão das peças 3d.....	81
8.7.5. Acoplagem dos módulos nos <i>cases</i> 3d	82
8.8. DESENVOLVIMENTO DE FIRMWARE	83
8.9. INSTALAÇÃO DO SISTEMA EM UMA MOTO REAL.....	90
8.9.1. Acoplagem dos <i>cases</i> e suporte nos periféricos da moto.....	90
8.9.2. Integração do chaveamento automático no relé de ignição	90
9. ANÁLISE DE RESULTADOS.....	91
9.1. TESTE DO SISTEMA DE ANTIFURTO E ROUBO DE MOTOCICLETAS.	91
9.1.1. Validação de funcionalidades de usuários com o módulo biométrico	92
9.1.2. Validação do aplicativo <i>mobile</i>	94
9.1.3. Validação em campo do sistema de antifurto e roubo de motocicletas	94
9.2. VALIDAÇÃO DO SISTEMA DE DETECÇÃO DE ACIDENTE	96
CONCLUSÃO	100
TRABALHOS FUTUROS.....	100
REFERÊNCIAS BIBLIOGRÁFICAS	102
FONTES CONSULTADAS.....	104
APÊNDICE A – CÓDIGO FONTE DO MICROCONTROLADOR DO SISTEMA	106

INTRODUÇÃO

Com o aumento da mobilidade urbana no Brasil, a motocicleta vem protagonizando esse cenário com um número crescente de veículos por habitantes. Nos últimos dez anos, a frota de motocicletas no País praticamente dobrou. Cresceu 91,8%, passando de cerca de 15 milhões em 2009 para 28 milhões em 2019, ou seja, uma moto para cada oito habitantes, de acordo com a Associação Brasileira dos Fabricantes de Motocicletas, Ciclomotores, Motonetas, Bicicletas e Similares (Abraciclo).

O furto de motocicletas em Manaus registrou aumento de 73% no primeiro quadrimestre deste ano, em relação ao mesmo período de 2021. Os dados são da Secretaria de Segurança Pública do Amazonas (SSP-AM).

Segundo a SSP, de janeiro a abril, foram furtadas 502 motocicletas, contra 289 nos quatro primeiros meses do ano passado. Por outro lado, o número de roubos caiu de 400 para 324 (SSP-AM).

Ademais, o número crescente de motos reflete diretamente índice de acidentes de trânsito. Em dez anos, no Brasil, o total de acidentados com motos e ciclomotores (veículos de até 50 cilindradas) cresceu 72%. É mais do que o dobro do avanço registrado em acidentes com outros meios de transporte, que subiu 28% no mesmo período. O salto foi ainda maior, de 142%, no caso de invalidez permanente. Em 2019 mais de 11 mil motociclistas morreram em um ano, segundo balanço do Ministério da Saúde. É inevitável, que diante desses dados seria se suma importância a checagem dos sinais vitais e agravamento do acidente para que o socorro seja feito da melhor forma possível, assim, reduzindo as taxas de mortalidade no trânsito (SSP-AM).

Diante do exposto, é imprescindível destacar duas situações: primeiro, a necessidade de bons sistemas antifurtos, para garantir a segurança das motocicletas, e eventualmente, a baixa no número de furtos no Brasil; e segundo, a importância de sistemas de aferições de sinais vitais, que revelará aos socorristas a gravidade da situação.

PROJETO DE PESQUISA

1 TEMA

Sistema inteligente integrado de proteção e detecção de acidentes de motociclistas e seus veículos.

2 FORMULAÇÃO DO PROBLEMA

Existem muitos roubos de motocicletas, por se tratar de veículos abertos e de fácil locomoção, além de possuírem poucas formas de prevenção contra esses crimes os quais não englobam inúmeras situações de risco. Além de não existirem formas de bloqueios comuns para motocicletas.

Além de problemas relacionados a crimes de roubos e furtos envolvendo esse tipo de veículo, também são bem comuns em cidades de grande tráfego; acidentes de trânsito, onde os motociclistas são as vítimas mais graves dessas tragédias, que, em determinadas situações é muito importante um chamado de socorro eficiente com sua localização específica e informações relacionadas ao acidentado estarem de posse dos paramédicos que estão a caminho de socorrer essas pessoas.

3 HIPÓTESE

É possível elaborar sistemas inteligentes que previnem crimes relacionados ao roubo e furto de veículos, em que de forma integrada por meio da validação por identidade biométrica através de sensores de leitura de digital, seja feita a verificação dos usuários da motocicleta em questão, dando acesso aos cadastrados e impedindo pessoas com algum objetivo criminoso de utilizar o veículo. Além do proprietário administrar os usuários cadastrados utilizando um aplicativo incorporado ao sistema via *internet of things* (IOT), onde irá adicionar ou deletar novas pessoas.

4 OBJETIVOS

4.1. OBJETIVO GERAL

Projetar um sistema que garanta a segurança de motociclistas e de seus veículos por meio de tecnologias de sistemas embarcados e internet das coisas. O sistema será composto por duas vertentes: Sistema antifurto, para segurança do veículo, que liberará sua ignição somente com a digital biométrica de pessoas autorizadas; e Sistema de análise de sinais vitais, no qual, por meio da conectividade de um aparelho smartwatch ao sistema acoplado, teremos acesso as aferições fisiológicas e níveis de consciência do condutor em caso de acidentes, acrescentados a outras informações do agravamento do acidente, em que esses dados serão encaminhados para entidades de socorro imediato ou algum contato de emergência, em caso de acidentes para uma interface web ou *desktop* desses usuários. Ambos os sistemas serão conectados a um aplicativo de gerenciamento de pessoas cadastradas, onde o proprietário poderá adicionar ou deletar cadastros no sistema, além verificações das funcionalidades do equipamento instalado no veículo.

O sistema também contará com protocolos de bloqueios funcionais da motocicleta caso esta tenha sido furtada ou roubada, tanto via aplicativo quanto outra digital biométrica cadastrada para esse fim, na qual acionará de forma temporária o veículo para uma distância segura entre o criminoso e a vítima. Além de possuir uma verificação de localização remoto acoplada ao sistema para futura recuperação do veículo que também pode ser vista pelo aplicativo web ou *mobile* que o proprietário tenha acesso.

4.2. OBJETIVOS ESPECÍFICOS

- Desenvolver um sistema de autenticação de usuário pela biometria.
- Criação um sistema eletrônico de bloqueio do veículo.
- Prototipar um sistema de localização do veículo.
- Produzir uma placa de circuito impresso para o projeto.
- Programar um *firmware* para cada módulo acoplado no projeto.
- Desenvolver uma Aplicação de notificação de roubo.
- Criação de um aplicativo WEB de amostragem de localização do veículo.
- Fazer um aplicativo para celular que interaja com o sistema físico.
- Administrar funcionalidades de banco de dados
- Desenvolver sistema eletrônico de detecção de acidentes.
- Modelar e manufaturar peças e *cases* 3D

- Instalar o sistema em uma moto real.
- Validar e testar o sistema contra furtos e roubos de motocicletas.
- Validar e testar o sistema de detecção automática de acidentes.

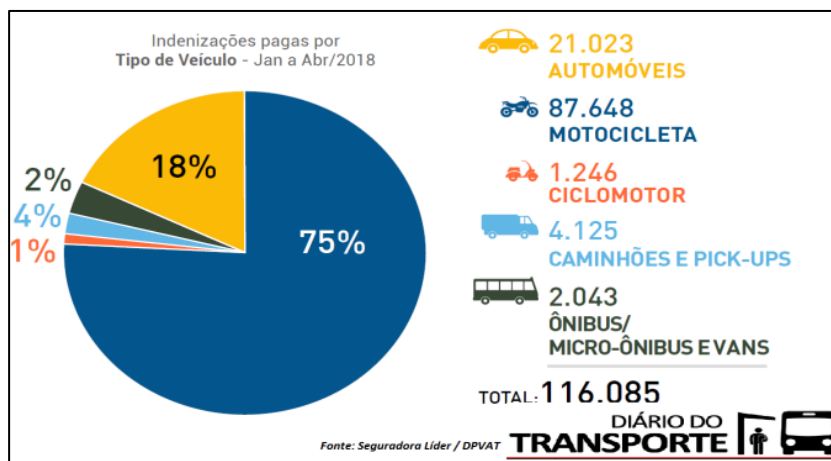
5 JUSTIFICATIVA

5.1. RELEVÂNCIA SOCIAL

Em relação a problemas no tratamento prévio, foi verificado que o agravamento no estado médico do paciente pode se alarmar por demora no socorro, além de informações prévias de triagem que podem ajudar socorristas no tratamento do acidentado.

Foi relatado que houve predomínio de acidentes motociclísticos com 42,2% dos traumas. As regiões corpóreas mais acometidas foram os membros inferiores/cintura pélvica (32,2%). Os ferimentos superficiais acometeram 88% das vítimas. Na Figura 1 podemos verificar a predominância da indenização pagas para tipo de veículos do tipo motocicletas retirada do Diário do Transporte da seguradora líder:

Figura 1: Porcentagem de Indenizações por tipos de veículos.



Fonte: Diário do transporte, 2022.

Com base nos dados acima, o sistema pré planejado irá auxiliar nos aspectos velocidade de atendimento, além de melhorar o pronto atendimento com as informações de aferições lidas pelo sistema, logo, auxiliando com relação a que medidas são mais bem recomendadas para o paciente em questão.

Em se tratando da análise de roubos e furtos, verificando a cidade de São Paulo, vemos na imagem abaixo alto número de roubos e furtos nesta capital, totalizando 33.394 de veículos levados como mostrado na Figura 2 .

Figura 2:Número de roubos e furtos de motos em são Paulo.

ANO	FURTOS	ROUBOS	TOTAL
2018	19.754	16.861	36.615
2019	20.439	14.788	35.227
2020	16.178	13.734	29.912
2021	20.532	12.862	33.394

Fonte: Mecânica Online, 2022.

Em relação a segurança de patrimônio em se tratando dos veículos motorizados de duas rodas, o sistema de autenticação por leitura biométrica irá ajudar a prevenir furtos com base em que somente o proprietário terá sua digital cadastrada, assim como outras pessoas que o usuário queira adicionar ao seu sistema de segurança.

O sistema de bloqueio da moto e localização, ajudará da recuperação desses veículos apropriados por elementos criminosos, visto que esse sistema deixará a motocicleta inutilizável pelo meliante, sendo assim, aumentam as chances de antes de chegar em seu destino, o elemento abandonar o patrimônio roubado ou furtado, e com o sistema de localização é possível recuperar o bem levado.

5.2. RELEVÂNCIA ACADÊMICA

O aprofundamento no aprendizado de microcontroladores em relação ao seu funcionamento utilizando periféricos de leitura e saída, programação nativa e redes de comunicação é de total pertinência na especialização de conhecimentos específicos da

engenharia elétrica com ênfase na eletrônica e de telecomunicação, pois no projeto será abordado a construção de *hardware* e de *firmware* de sistemas embarcados para IOT.

A área de Internet das coisas, engloba uma série de sistemas de telecomunicação que dependente do projeto de desenvolvimento, poderá ser usado diferentes protocolos de comunicação, assim como criptografia de pacotes de informações e interfaces de conexão entre microcontroladores e banco de dados online.

6 REFERENCIAL TEÓRICO

6.1. ACIDENTADO DE TRÂNSITO

Pode-se definir o acidente de trânsito como um incidente involuntário do qual participam, pelo menos, um veículo em movimento, pedestres e obstáculos fixos, isolado ou conjuntamente, ocorrido numa via terrestre, resultando danos ao patrimônio, lesões físicas ou morte. (PERITO DE TRÂNSITO, 2017).

6.2. LEIS DO CÓDIGO DE TRÂNSITO BRASILEIRO QUANTO A DISPOSITIVOS ANTIFURTOS.

O CONSELHO NACIONAL DE TRÂNSITO - CONTRAN, no uso das atribuições que lhe são conferidas pelo art. 12, da Lei nº 9.503, de 23 de setembro de 1997, que instituiu o Código de Trânsito Brasileiro - CTB, e conforme o disposto no Decreto nº 4.711, de 29 de maio de 2003, que trata da coordenação do Sistema Nacional de Trânsito - SNT;

Considerando as atribuições conferidas ao CONTRAN pela Lei Complementar nº 121, de 9 de fevereiro de 2006, que cria o Sistema Nacional de Prevenção, Fiscalização e Repressão ao Furto e Roubo de Veículos e Cargas e dá outras providências e o disposto no caput do art. 105, da Lei nº 9.503, de 23 de setembro de 1997, a fim de estabelecer a obrigatoriedade de equipamento antifurto nos veículos novos saídos de fábrica, produzidos no País ou no exterior; Considerando o que consta do Processo nº 80001.003014/2007-99, resolve:

Art. 1º - Todos os veículos novos, saídos de fábrica, produzidos no País ou importados a partir de 24 (vinte e quatro) meses da data da publicação desta Resolução somente poderão ser comercializados quando equipados com dispositivo antifurto.

§ 1º O equipamento antifurto deverá ser dotado de sistema que possibilite o bloqueio autônomo (local) e bloqueio remoto. (NR dada pela Resolução CONTRAN nº 329 de 2009).

6.3. BLOQUEIO DE VEÍCULO

O bloqueador de veicular é um dispositivo instalado no veículo ou caminhão que tem como função o bloqueio da ignição ou o corte da bomba de combustível do veículo. O bloqueador consiste em um conjunto eletrônico formado principalmente por um relé e os cabos de conexão, que se conectam ao circuito do veículo. (INFLEETR, 2011).

Basicamente, o sistema imobilizador de motocicletas é um recurso de segurança contra furtos. Ele impede o funcionamento da motocicleta, a menos que seja utilizada a chave correta. Dessa forma, os artifícios mais comuns empregados nesse tipo de crime, como chave micha ou jumper na ignição, não são suficientes para dar a partida em um veículo que possui um sistema imobilizador. (INFLEETR, 2011).

Essa tecnologia, inclusive, é muito superior ao tradicional alarme, que costuma cortar o funcionamento apenas após alguns metros rodados. Outra vantagem é sua maior discricção, já que não é preciso carregar nenhum dispositivo adicional, apenas a própria chave da motocicleta (INFLEET, 2011).

6.4. ACESSO BIOMÉTRICO

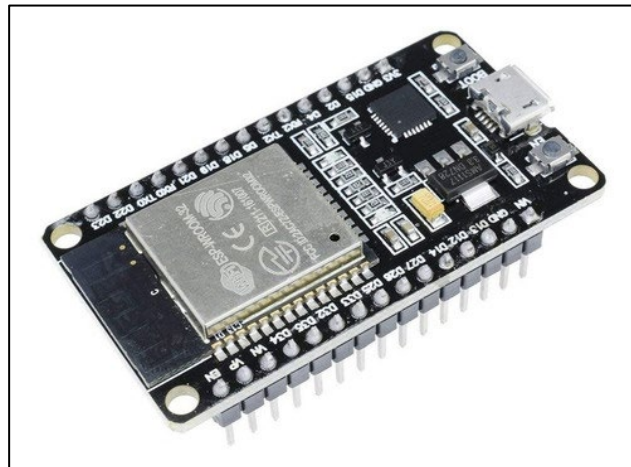
Controle de acesso por biometria consiste em um processo de identificação da impressão digital de um usuário e, caso este esteja autorizado no sistema de monitoramento, sua entrada é liberada no ambiente em questão. (CONTROLID, 2020)

A biometria é uma tecnologia em grande expansão, já usada por clientes e empresas em diversas aplicações. Com o reconhecimento facial, por meio de uma selfie e comparação com a foto usada no documento, é possível garantir que a pessoa é ela mesma. Ou seja, é mais uma medida de segurança nos processos anti-fraude. (CONTROLID, 2020)

6.5. ESP32

É uma série de microcontroladores de baixo custo e baixo consumo de energia. também é um sistema-em-um-chip com microcontrolador integrado, Wi-Fi e Bluetooth. Desenvolvido pela empresa Espressif, o ESP32 como mostrado na Figura 3 apresenta-se como um meio inovador no desenvolvimento de projetos automatizados. Esse pequeno componente demonstra ser mais versátil do que seu antecessor, o ESP8266, pois além do clássico módulo de comunicação Wi-Fi, apresenta um sistema com processador Dual Core, Bluetooth híbrido e múltiplos sensores embutidos, tornando a construção de sistema como internet das coisas (IOT) muito mais simples e compacto (EXPRESSIF SYSTEMS 2016).

Figura 3: Esp 32.



Fonte: Espressif, 2015.

6.6. BUBBLE

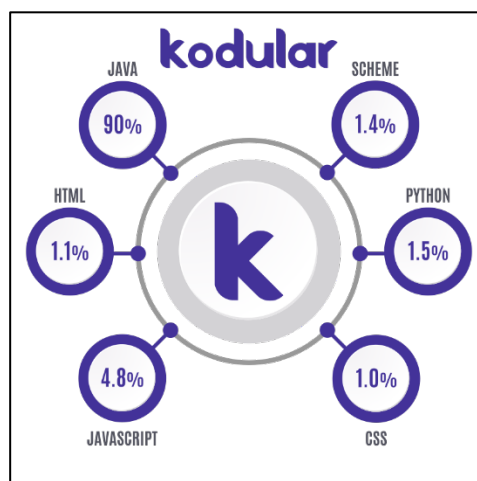
É uma plataforma online de desenvolvimento de aplicações web, integrada com linguagem de programação visual, ambiente de desenvolvimento de aplicação intuitiva de WebApps (sem necessidade de programação por código) e, serviço de hospedagem, desenvolvida pelo Bubble Group, para os usuários leigos em tecnologia desenharem uma interface através do método de clicar e arrastar elementos em uma página (drag and drop), bastando apenas definir fluxos de trabalho para controlar as ações.

O ambiente de desenvolvimento de desenvolvimento visual é executado no navegador (sem necessidade de download ou plugin) e, permite criar sites e aplicativos da web (WebApps) com funcionalidades avançadas,[1] mais completas que os construtores web orientados a modelos, como Wix e Squarespace. . (BUBBLE.IO, 2021).

6.7. KODULAR

Kodular é uma plataforma gratuita que oferece ferramentas e suporte para o usuário com o objetivo de facilitar o processo de criação de aplicativo. A partir de blocos lógicos de programação, essa ferramenta permite a criação de apps sem a necessidade de conhecimentos aprofundados sobre programação, como mostra a Figura 4. Ele conta com um layout simples e intuitivo, sendo necessário poucos cliques ou arrastar alguns ícones e pronto! Sua própria ideia começa tomar a forma de um aplicativo para sistema Android (KODULAR, 2019).

Figura 4: Kodular.



Fonte: Kodular ,2019.

Inspirado na ideia de um sistema de *software* desenvolvido pelo Instituto de Tecnologia de Massachusetts (MIT), o AppInventor, o Kodular oferece ainda uma App Store autodesenvolvida, onde se encontram aplicativos feitos a partir dele, um ID de extensões e um sistema para controle de contas. (PASSARELLI, 2017).

6.8. SHEETY.COM

Sheety transforma sua planilha em uma API JSON Restful, o que significa que você pode obter dados de sua planilha usando solicitações HTTP simples e URLs. Isso também significa que o Sheety pode trabalhar com praticamente qualquer coisa que você quiser: se ele se conectar à Internet, funcionará com o Sheety (SHEETY.CO, 2020).

O Google planilhas é um “Excel online”, que te permite criar tabelas e coisas do tipo, online, sem a necessidade de instalar no seu computador. Podemos compartilhar essa tabela para outras pessoas visualizarem ou editá-las. O Google Planilhas pode **criar um banco de dados online, gratuito e simples**, visto que **não** precisamos ter um servidor dedicado à hospedar o serviço (SHEETY.CO, 2020).

6.9. IOT

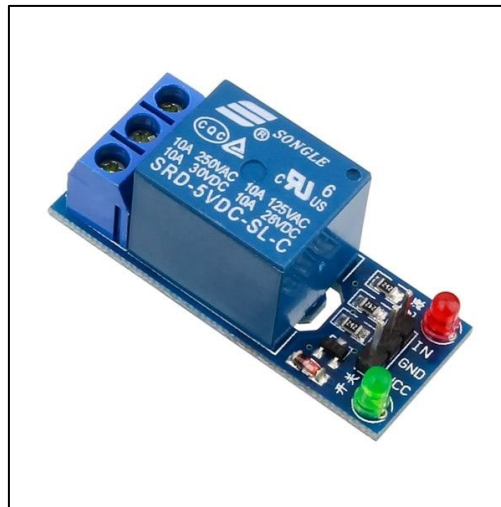
Internet das coisas é um conceito que se refere à interconexão digital de objetos cotidianos com a internet, conexão dos objetos mais do que das pessoas. Em outras palavras, a internet das coisas nada mais é que uma rede de objetos físicos capaz de reunir e de transmitir dados do básico para o complexo, IOT nada mais é que, uma rede de “coisas” interligadas, esse conceito permite o monitoramento e ação sobre dispositivos em qualquer lugar que possam estar independentes da distância que estejam. Sim, essas “coisas” seriam, por exemplo, batedeiras de bolo, lâmpadas, *smart TVs*, *smartphones*, câmeras de monitoramento, máquinas de lavar, semáforos, caminhões, carros, turbinas de avião, mísseis etc (AMAZON AWS, 2020).

Todos possuem sensores que emitem e recebem dados, isso é possível devido a conectividade de rede e eletrônica embarcada nesses dispositivos tornando-os responsivos, ou seja, que respondem, que agem conforme os eventos ocorrem no tempo e um *software* que analisa esses dados aplicando regras que definem o que fazer, ou atribuindo a você a tomada de decisão sobre esses dados. (AMAZON AWS, 2020).

6.10. MÓDULO RELÉ

O Módulo Relé como mostrado na Figura 5, é um módulo utilizado para facilitar o acionamento de cargas através de um microcontrolador, necessitando apenas conectar *Jumpers* no *Relé* e em uma placa Arduino, por exemplo, dispensando a necessidade de montar placas ou circuitos para a sua ligação, tornando o projeto mais prático e organizado. (ELETROGATE, 2017).

Figura 5: Módulo relé.



Fonte: Eletrogate, 2018.

O Módulo Relé Comum funciona como uma espécie chave que trabalha em três posições, Normal Aberto, Normal Fechado e Comum, sendo chaveada a quando recebe uma tensão de 5V através do comando do microcontrolador. Já o Módulo Relé de Estado Sólido tem basicamente a mesma função, tendo como grande diferença o fato de não possuir elementos mecânicos ou qualquer tipo de peça móvel, funcionando a partir de tiristores ao invés de contatos e o fato de trabalhar apenas com as posições Normal Aberto e Normal Fechado. Por não possuir partes mecânica, não sofre desgaste e tem vida útil praticamente ilimitada. (ELETROGATE, 2017).

6.11. SISTEMA DE IGNIÇÃO DA MOTO

O objetivo do sistema da ignição é fornecer uma centelha (faísca gerada entre os pólos da vela no interior da câmara de combustão antes do pistão se aproximar do fim do curso de compressão, a fim de iniciar a queima da mistura ar-combustível. O instante em que ocorre o centelhamento tem importância para a eficiência e desempenho do motor. Como a queima da mistura (ar/combustível) não é instantânea, quanto mais rápida é a velocidade de rotação do motor, mais adiantada deve ser o início da queima. A isso se dá o nome de adiantar a ignição (MOTONLINE, 2007).

Para aproveitar melhor a mistura, é necessário que toda ela termine de queimar pouco depois do pistão passar do PMS, onde ocorrerá a máxima pressão dentro da câmara de combustão. O avanço da ignição é, então, de fundamental importância para o rendimento do motor. Com o desenvolvimento da eletrônica foi possível aprimorar este sistema e atualmente há em todas as motocicletas um sistema eletrônico que é o responsável por fornecer a centelha no instante exato para cada rotação do motor, gerando economia de combustível, redução da emissão de gases tóxicos e diminuição da perda de rendimento do motor. Estamos nos referindo ao módulo de controle da ignição ou simplesmente ICM, podendo ser um CDI (ignição por descarga capacitiva) ou IDI (Ignição por Descarga Indutiva). Todo módulo de ignição moderno possui um pequeno processador de dados, que nada mais é que um processador, parecido com o de um computador, porém de capacidade menor. É na memória do processador está armazenada a curva de avanço do ponto, que basicamente é a relação entre a rotação do motor e o avanço do ponto. A curva de avanço depende de várias características do motor e da moto (MOTONLINE, 2007).

Para que o processador consiga gerar o sinal para a faísca no ponto correto são necessárias duas informações: velocidade de rotação e a posição do pistão. Estes dois sinais são obtidos através de sensores. A configuração mais comum é a de um sensor apenas, mas podem ser mais. O sensor mais comum é a bobina de pulso, também chamada de “pickup”. Pela bobina passam ressaltos metálicos, que normalmente estão no volante do magneto, mas também podem estar em um disco dentado. Na passagem de cada ressalto dois sinais elétricos são gerados, um pulso positivo e um negativo, ou invertido, negativo e depois positivo. Os sinais da bobina de ignição chegam ao módulo de ignição, na etapa chamada “Condicionador de Sinal”, que transforma estes sinais em sinais elétricos que podem ser “lidos” pelo processador. O processador interpreta estes sinais e extrai as duas informações que necessita: posição e velocidade de rotação. Com estas informações o processador obtém o avanço e no momento correto, conforme a posição do motor, gera o sinal para a etapa de potência. Na unidade de potência o sinal gerado pelo processador é usado para disparar um pulso de média tensão (na faixa de 100 a 900 volts) sobre a Bobina de Ignição que trabalha similar a um transformador, elevando a tensão. Este pulso no enrolamento primário da bobina de ignição faz “aparecer” a alta tensão em seu secundário (similar a um transformador), que ligado na vela de ignição gera a faísca para iniciar a queima da mistura de ar e combustível que se encontra dentro da câmara de combustão (MOTONLINE, 2007).

Nos IDIs a energia para o pulso de média tensão é armazenada na própria bobina de ignição, em forma de campo magnético. Para fazer isso a bobina de ignição usada é ligada pelo módulo de ignição em 12 volts (da bateria). Enquanto ligada aos 12 volts circulará pelo primário da bobina uma corrente que irá gerar o campo magnético. A média tensão no primário é gerada ao desligar a bobina dos 12 volts. Em função do comportamento indutivo da bobina (daí o nome descarga indutiva), no momento em que ela é desligada irá surgir no primário da bobina um pulso de tensão na faixa entre 300 e 900 volts. Os IDIs são eletronicamente mais simples, porém sua bobina de ignição é normalmente maior e mais complexa. Finalmente, há ainda os sinais de bloqueio, usados para impedir que a moto ligue em determinadas situações. Os mais comuns são o do descanso lateral e do neutro (ponto morto). Estes sinais evitam, então, que o motor ligue em uma situação que poderia derrubar o motociclista (MOTONLINE, 2007).

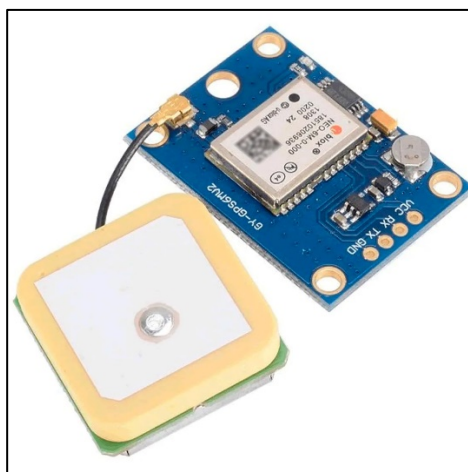
6.12. MEDIÇÃO DE SINAIS VITAIS

A avaliação de sinais vitais faz parte da rotina de muitos serviços de saúde. E não é para menos, dada a importância do pulso, frequência respiratória, pressão arterial e temperatura. Esses 4 itens determinam a condição de saúde do paciente, demonstrando o desempenho em atividades corporais indispensáveis à sobrevivência. Daí a necessidade de saber como verificar os sinais vitais, quais os valores normais e que condutas adotar diante de alterações. (TELEMEDICINA, 2022)

6.13. MÓDULO GPS

O Módulo GPS GY-NEO6MV2 como mostrado na Figura 6, é um dispositivo que tem como finalidade definir a geolocalização e fornecer os dados para uma plataforma microcontrolada. Este Módulo conta com uma antena externa para melhorar a recepção de sinal e a comunicação com a plataforma microcontrolada é feita via serial (RX / TX) (MASTERWALKER, 2016).

Figura 6: Módulo GPS.



Fonte: Masterwalker, 2016.

O GPS ou Global Positioning System (Sistema de Posicionamento Global) é uma rede de satélites que permite a obtenção de um posicionamento geográfico de um determinado dispositivo ou aparelho que tenha um módulo GPS. A tecnologia foi desenvolvida e é controlada pelos Estados Unidos. O GPS utiliza uma rede de pelo menos 24 satélites que orbitam a Terra e enviam sinais que permitem aos dispositivos determinar, com ótima precisão, a posição em que eles se encontram na superfície da Terra. (INFOESCOLA, 2015)

Inicialmente o GPS foi desenvolvido pensando em aplicações militares. Por isso, apesar da tecnologia ter se popularizado para múltiplas aplicações civis, as principais potências do mundo desenvolveram suas próprias redes de satélite e que podem também ser usadas para aplicações civis. A Europa desenvolveu um sistema chamado Galileu, a China o Beidou e a Rússia o Glonass (MAKEHERO).

A tecnologia GSM utiliza uma ampla rede de torres de transmissão e recepção para viabilizar a comunicação entre dispositivos através de longas distâncias. Geralmente, uma torre estabelece uma conexão com um dispositivo, como um aparelho celular, um Arduino ou Raspberry Pi, com o qual troca informações. A torre, em geral, estabelece conexão com outras torres através de um sistema de antes de longa distância, permitindo que o sinal captado pelo dispositivo possa ser transmitido para as torres subsequentes até alcançar uma estação base conectada por cabo com servidores mais robustos. (ELETROGATE, 2022)

A rede GSM é formada por milhares de torres de transmissão e recepção espalhadas por todo o mundo, mas concentrada em regiões urbanas, o que faz com que a qualidade e intensidade do sinal sejam melhores nas cidades. A tecnologia GSM é mais conhecida como redes 2G, 3G e 4G que nos permitem utilizar nossos celulares para ligar e enviar mensagens de texto, imagens e outros (INFOWESTER, 2013).

Para fazer uso dessa rede, o dispositivo precisa de um chip celular e o operador deve ter uma conta de telefonia com algumas das operadoras disponíveis na região. Também podem ser utilizados planos pré-pagos. (MASTER WALKER, 2016)

6.14. COMUNICAÇÃO HTTP

HTTP é um protocolo que permite a obtenção de recursos, como documentos HTML. É a base de qualquer troca de dados na Web e um protocolo cliente-servidor, o que significa que as requisições são iniciadas pelo destinatário, geralmente um navegador da Web. Um documento completo é reconstruído a partir dos diferentes sub -documentos obtidos, como por exemplo texto, descrição do layout, imagens, vídeos, scripts e muito mais (ROCKCONTET, 2019).

Clientes e servidores se comunicam trocando mensagens individuais (ao contrário de um fluxo de dados). As mensagens enviadas pelo cliente, geralmente um navegador da Web, são chamadas de **solicitações** (*requests*), ou também **requisições**, e as mensagens enviadas pelo servidor como resposta são chamadas de **respostas** (*responses*) (DEVELOPER MOZILA, 2022).

Projetado no início da década de 1990, o protocolo HTTP é extensível e evoluiu ao longo do tempo. Atua na camada de aplicação e é enviado sobre o protocolo TCP, ou em uma conexão TCP criptografada com TLS, embora qualquer protocolo de transporte confiável possa, teoricamente, ser usado. Devido à sua extensibilidade, ele é usado não só para buscar documentos de hipertexto, mas também imagens e vídeos ou publicar conteúdo em servidores, como nos resultados de formulário HTML (veja os elementos <html> e <form>). O HTTP também pode ser usado para buscar partes de documentos para atualizar páginas da Web sob demanda. (DEVELOPER MOZILLA, 2022)

6.15. CRIPITOGRAFIA RSA

RSA (Rivest-Shamir-Adleman) é um dos primeiros sistemas de criptografia de chave pública e é amplamente utilizado para transmissão segura de dados. Neste sistema de criptografia, a chave de encriptação é pública e é diferente da chave de decifração que é secreta (privada). No RSA, esta assimetria é baseada na dificuldade prática da fatorização do produto de dois números primos grandes, o "problema de fatoração". O acrônimo RSA é composto das letras iniciais dos sobrenomes de Ron Rivest, Adi Shamir e Leonard Adleman, fundadores da atual empresa RSA Data Security, Inc., os quais foram os primeiros a descrever o algoritmo em 1978. Clifford Cocks, um matemático Inglês que trabalhava para a agência de inteligência britânica Government Communications Headquarters (GCHQ), desenvolveu um sistema equivalente em 1973, mas ele não foi revelado até 1997 (LAMBDA, 2012).

É considerado dos mais seguros, já que mandou por terra todas as tentativas de quebrá-lo. Foi também o primeiro algoritmo a possibilitar criptografia e assinatura digital, e uma das grandes inovações em criptografia de chave pública(LAMBDA, 2012).

Um usuário do RSA cria e publica uma chave (chave pública) baseada em dois números primos grandes, junto com um valor auxiliar. Os números primos devem ser mantidos secretos. Qualquer um pode usar a chave pública para encriptar a mensagem, mas com métodos atualmente publicados, e se a chave pública for muito grande, apenas alguém com o conhecimento dos números primos pode decodificar a mensagem de forma viável. Quebrar a encriptação RSA é conhecido como problema RSA. Se ele for tão difícil quanto o problema de fatoramento, ele permanece como uma questão em aberto (LAMBDA, 2012).

O RSA é um algoritmo relativamente lento e, por isso, é menos usado para criptografar diretamente os dados do usuário. Mais frequentemente, o RSA passa chaves criptografadas compartilhadas para criptografia de chave simétrica que, por sua vez, pode executar operações de criptografia-descriptografia em massa a uma velocidade muito maior. (WIKIPÉDIA, 2022)

6.16. PROTEÇÃO DE MEMÓRIA FLASH

A criptografia da memória flash (AES-256) é uma característica presente no ESP32 que criptografa o conteúdo presente na flash. Quando habilitado, leituras físicas sem a chave não são suficientes para recuperar o conteúdo. Sendo assim, nos protegemos de quem tentar exportá-la para clonagem etc. A chave é gravada em um bloco de eFuse, que pode ser protegido contra leitura e escrita (padrão) e, conhecendo a chave, podemos regravar códigos sem a limitação de quantidade, diferentemente do caso em que o ESP32 gera sua própria chave, onde estamos limitados em até 3 uploads físicos. (EMBARCADOS, 2022)

6.17. DATA BASE

Um banco de dados é uma coleção organizada de informações - ou dados - estruturadas, normalmente armazenadas eletronicamente em um sistema de computador. Um banco de dados é geralmente controlado por um sistema de gerenciamento de banco de dados (DBMS). Juntos, os dados e o DBMS, juntamente com os aplicativos associados a eles, são chamados de sistema de banco de dados, geralmente abreviados para apenas banco de dados. (ORACLE, 2017)

Os dados nos tipos mais comuns de bancos de dados em operação atualmente são modelados em linhas e colunas em uma série de tabelas para tornar o processamento e a consulta de dados eficientes. Os dados podem ser facilmente acessados, gerenciados, modificados, atualizados, controlados e organizados. A maioria dos bancos de dados usa a linguagem de consulta estruturada (SQL) para escrever e consultar dados (ORACLE, 2017).

O *software* de banco de dados é usado para criar, editar e manter arquivos e registros de banco de dados, facilitando a criação de arquivos e registros, entrada de dados, edição, atualização e relatórios de dados. O *software* também processa armazenamento de dados, backup e relatórios, controle multiacesso e segurança. A segurança forte do banco de dados é especialmente importante hoje, porque o roubo de dados se torna mais frequente. O *software* de banco de dados às vezes também é conhecido como "sistema de gerenciamento de banco de dados" (DBMS, 2022).

O *software* de banco de dados simplifica o gerenciamento de dados, permitindo que os usuários armazenem dados em um formulário estruturado e depois os acessem. Ele normalmente tem uma interface gráfica para ajudar a criar e gerenciar os dados e, em alguns casos, os usuários podem construir os próprios bancos de dados usando o *software* do banco de dados (DEV MEDIA).

Um banco de dados normalmente requer um programa abrangente de banco de dados, conhecido como sistema de gerenciamento de banco de dados (DBMS). Um DBMS serve como uma interface entre o banco de dados e seus usuários finais ou programas, permitindo que os usuários recuperem, atualizem e gerenciem como as informações são organizadas e otimizadas. Um DBMS também facilita a supervisão e o controle de bancos de dados, permitindo uma variedade de operações administrativas, como monitoramento de desempenho, ajuste e backup e recuperação (ORACLE 2021).

6.18. PROGRAMAÇÃO EM C E C++

Criada pelo cientista da computação Dennis Ritchie, em 1972, a **linguagem C** foi derivada de outras duas: a BCPL e a Algol 68. Embora tenha sido pensada com o propósito exclusivo de ser uma linguagem de programação usada no desenvolvimento de uma nova versão do sistema operacional Unix, hoje é aplicada nos mais variados tipos de projeto (LINGUAGEMC, 2000).

A **linguagem C** ainda é uma das mais populares do mercado de programação devido às diversas vantagens que apresenta. Por isso, é quase uma obrigatoriedade no currículo de uma pessoa que trabalha com desenvolvimento (BETRYBE, 2020).

Considerada uma linguagem de alto nível genérica, a C pode ser usada em diversos tipos de projeto, como a criação de aplicativos, sistemas operacionais, drivers, entre outros. Trata-se de uma linguagem estruturada que se tornou muito popular nos anos 80 — tanto que é difícil encontrar arquiteturas para as quais não existam compiladores para a C, o que garante o seu **elevado nível de portabilidade** (BETRYBE, 2020).

Uma das grandes vantagens dessa linguagem é a capacidade de gerar códigos rápidos, ou seja, com um tempo de execução baixo. Além disso, a programação em C é bastante simplificada, pois sua estrutura é simples e flexível. Tendo isso em mente, podemos dizer que as principais características da linguagem C são: portabilidade,

geração de código eficiente, simplicidade, confiabilidade, facilidade de uso, regularidade (TREINAMENTO24, 2020) .

6.19. EASYEDA

EasyEDA é uma plataforma de design e simulação de circuitos eletrônicos online. Ele fornece ferramentas para criar esquemas, simular circuitos, gerar layouts de PCB e realizar testes de prototipagem. EasyEDA é uma opção popular para projetos de eletrônica, pois oferece uma ampla gama de recursos gratuitos e fáceis de usar em uma única plataforma na nuvem. Além disso, ele também oferece recursos avançados para usuários pagantes, incluindo suporte para gerar arquivos Gerber para produção em massa (EMBARCADOS, 2015).

6.20. PROTOTIPADORA DE PLACA DE CIRCUITO IMPRESSO

Uma prototipadora de placa de circuito impresso é uma máquina que é usada para produzir protótipos de placas de circuito impresso. É uma ferramenta valiosa para designers e engenheiros eletrônicos, pois permite que eles testem e validem seus designs antes de produzi-los em massa. As prototipadoras de placa de circuito impresso funcionam cortando ou gravando o desenho do circuito em uma placa de circuito impresso, adicionando componentes e realizando testes. Algumas prototipadoras são manuais, enquanto outras são totalmente automatizadas. Elas variam em precisão, velocidade e complexidade, e algumas também oferecem recursos adicionais, como teste de funcionamento e geração de arquivos para produção em massa (VICTORVISION, 2022).

7. METODOLOGIA

O Trabalho apresentado será uma pesquisa aplicada, e terá como objetivo a realização de pesquisa exploratória sobre o material bibliográfico e de laboratório. Serão utilizados os procedimentos técnicos de pesquisa bibliográfica e experimental. Será utilizado o método de abordagem hipotético-dedutivo e o método de procedimento monográfico em sua elaboração. Para coleta de dados será utilizada documentação indireta e a análise e interpretação de seus dados qualitativos ocorrerão globalmente.

O projeto necessita de estudos profundos de cada tecnologia mostra na primeira etapa do cronograma; os quais se especificam na estrutura de portas do ESP32 e sua linguagem e formas de código, assim como o entendimento da sua conexão bluetooth e WI – FI, adicionando também para entendimento do último, será necessário o estudo da

IDE do Arduino e da ESP IDE, a qual também servirá para programar o Esp. Além desses, vamos compreender as linguagens de criação de Apks para *mobile* junto ao módulo de leitura biométrica. Assim sendo o projeto será dividido em 4 etapas:

A primeira etapa será destinada especificamente ao estudo das tecnologias citadas no referencial bibliográfico e então criar um filtro de validação e de utilidade dos equipamentos que serão usados no projeto e a aquisição do material filtrado necessário para a montagem do circuito físico em se tratando de microcontrolador, sensores periféricos me como modelagem de peças para suportes de equipamentos prototipados.

A segunda etapa será necessária uma pesquisa focada em design para uma melhor interação do usuário com a aplicação *mobile* conectada com o sistema, em que será usado ferramentas do google para experiências virtuais de aplicativos, onde será analisado itens dinâmicos visuais assim como blocos de informações com boas estéticas visuais.

A terceira etapa será verificar requisitos funcionais que a aplicação *mobile* necessitará em sua estrutura com o objetivo obter praticidade e conter as demandas relacionadas ao sistema em se tratando frameworks e pacotes de desenvolvimento de aplicações de dispositivos físicos, bem como artificios de conectividade de internet das coisas e *mobile security*.

A quarta etapa envolverá a prototipação do circuito que será feita com base em código de programação de identificação de usuário, que relacionará a leitura do dado biométrico com um sistema de acionamento por relé a ignição da moto. Nesse item, será embasado conforme as normas de qualidade de manufatura de qualidade de protótipos.

7.1. DIAGRAMA DE FUNCIONAMENTO DO PROJETO

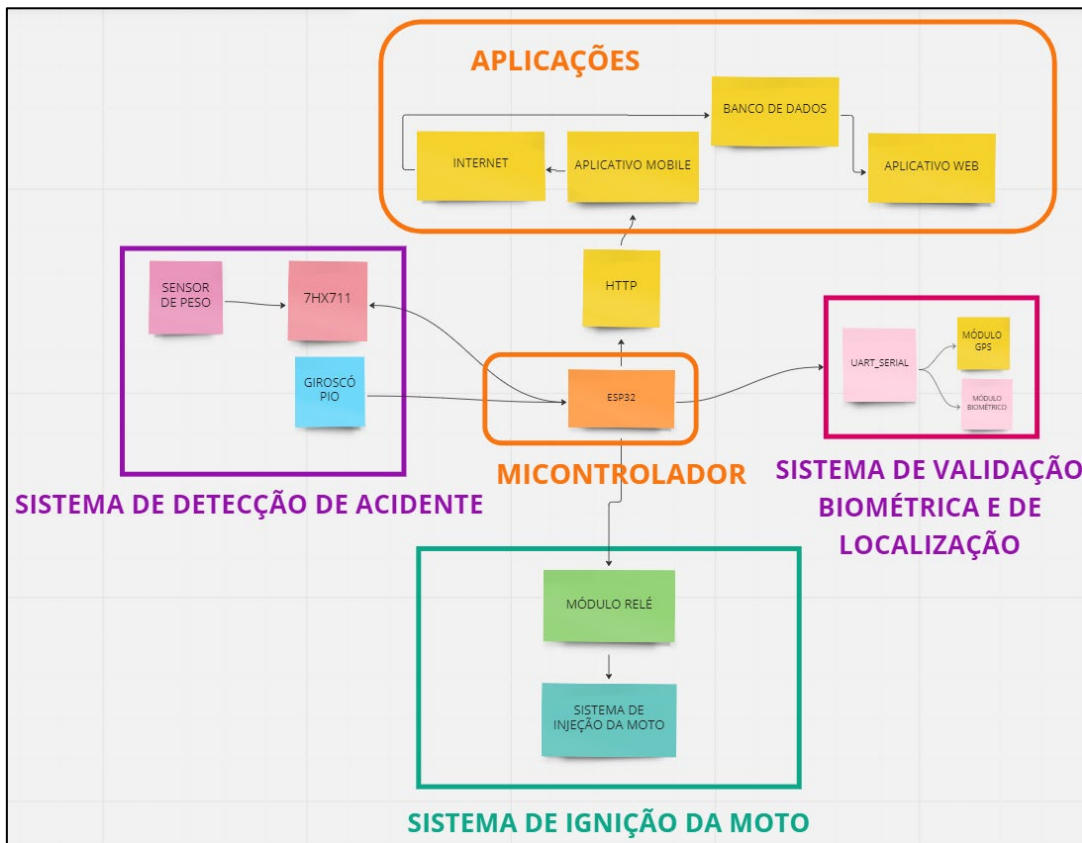
O projeto é composto basicamente de cinco blocos de extensão de funcionalidades, os quais fazem parte do segmento de detecção de acidentes de motociclistas e de prevenção contra furtos e roubos de motocicletas como mostra a Figura 3.

Todos os blocos se centralizam no microprocessador, o qual será programado para a leitura de módulos sensores, bem como funções de comando para atuadores, além de ter comunicação com dispositivo celular do usuário.

O sistema de detecção de acidente é composto basicamente da leitura de dois sensores, que são: sensor de peso com um amplificador de sinal para verificação de presença do motociclista, e o módulo giroscópio, para ver a angulação da moto em relação a superfície para detectar tombos.

O sistema de validação biométrica e de localização possui um módulo de leitura de digital, que analisa se o usuário inserido na leitura está no banco de dados, impedindo que pessoas sem permissão usem a motocicleta. O Sistema também possui um módulo gps responsável por captar as informações de coordenadas de localização, responsável para enviar a localização da motocicleta para um banco de dados caso ela seja roubada ou o piloto sofra um acidente como mostra Figura 9.

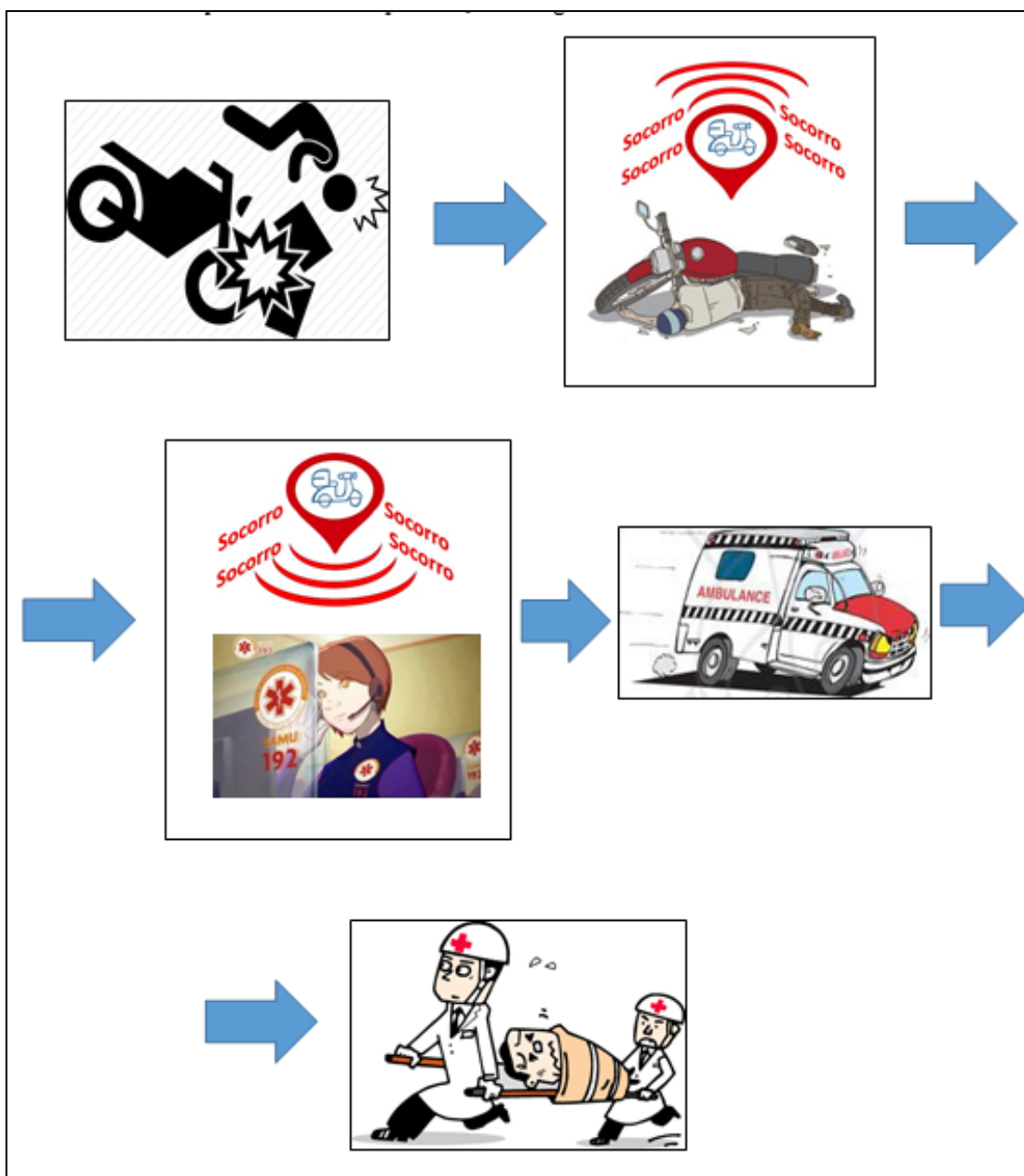
Figura 7: Diagrama de funcionamento do projeto.



Fonte: Próprio autor.

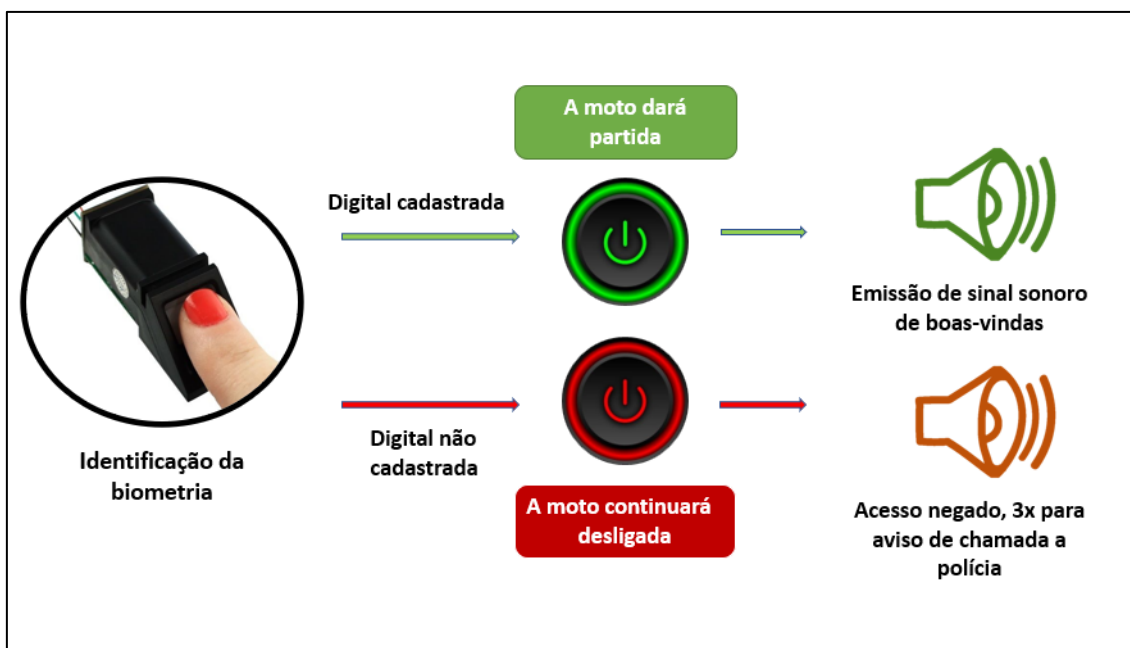
Além disso, há presença de uma integração com as leituras de sinais vitais do motociclista bem como sua localização com o sistema, para envio as instituições de socorro para auxílio aos socorristas em caso de acidente, como mostra a Figura 8.

Figura 8: Funcionamento da detecção de acidente inteligente.



Fonte: Próprio autor.

Figura 9: Funcionamento do sistema de autenticação por leitura biométrica.



Fonte: Próprio autor.

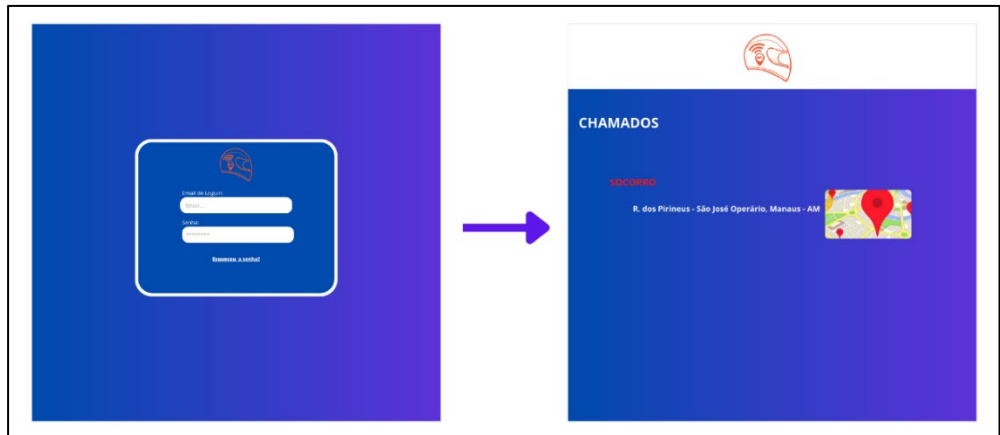
8. IMPLEMENTAÇÃO DO PROJETO

8.1. *SOFTWARE* DE ALERTA DE CHAMADOS

Primeiramente foi feito um escopo inicial das páginas do aplicativo de chamados, para ser possível a idealização e localização de cada componente e assim prever o funcionamento da aplicação integrada ao sistema físico construído.

Podemos analisar Figura 10 abaixo o fluxo de duas páginas web conectadas em relação a entrada do usuário ao site por meio de uma validação de e-mail e senha, logo em seguida a página de chamados de socorro é manifestada, onde apresentará toda a lista de socorros em que o sistema físico detectou.

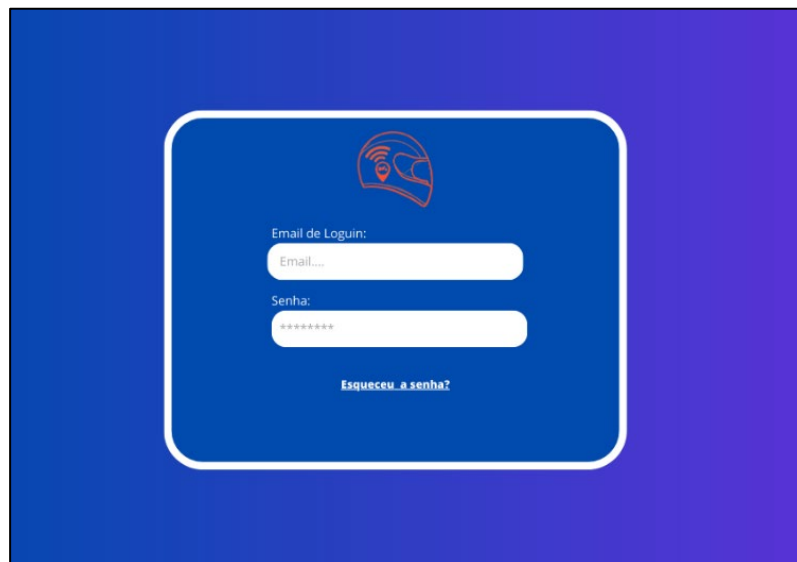
Figura 10: Fluxo de páginas do aplicativo WEB



Fonte: Próprio autor.

Na tela inicial de login foi desenvolvida os componentes de interação com o usuário onde existe as entradas de email e senha para validação de usuários cadastrados, além de um botão que direciona para a mudança de senha caso o cliente tenha perdido essa informação como mostrado na Figura 11.

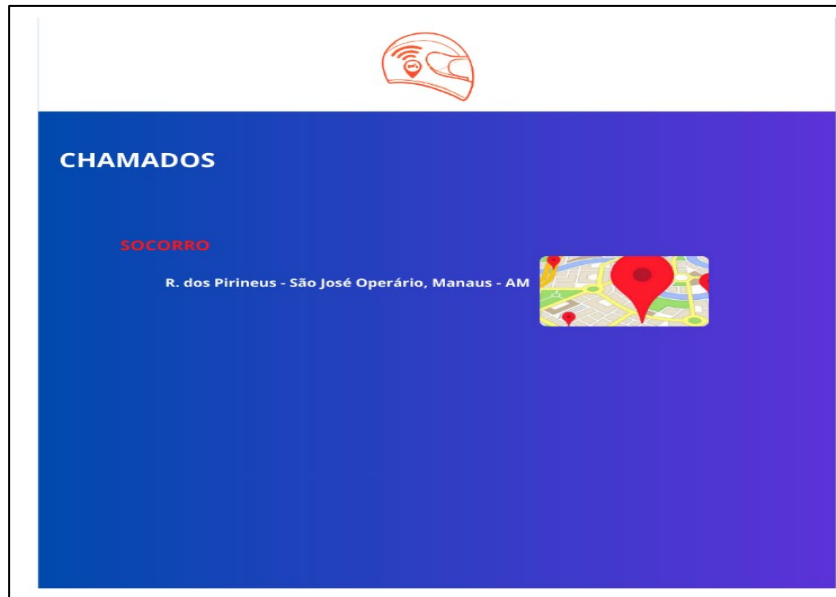
Figura 11: Esboço da página de Login.



Fonte: Próprio autor.

Na tela de chamados analisamos uma *dashboard* apresentando em destaque palavra socorro assim como a localização de onde ocorreu o acidente com o motociclista cadastro no sistema como mostrado na Figura 12.

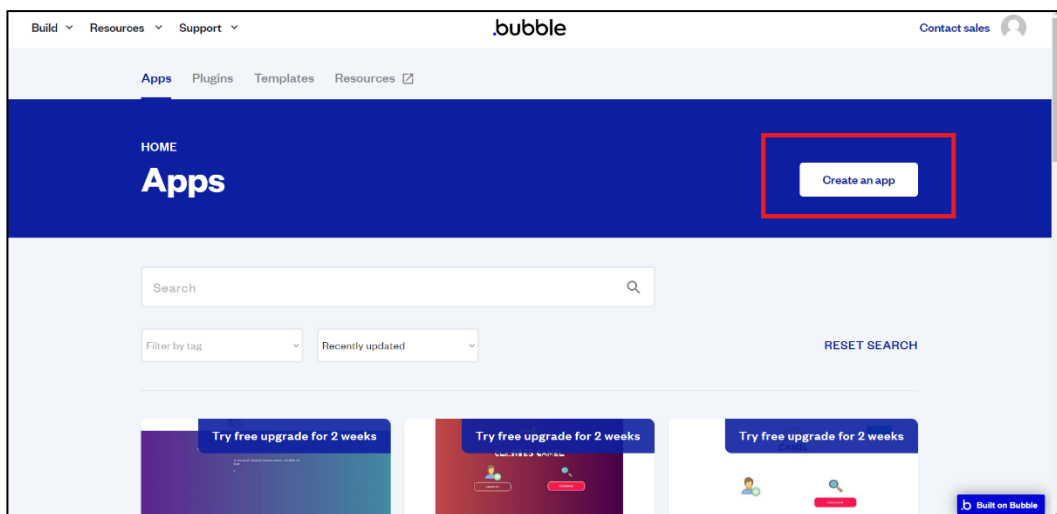
Figura 12: Esboço da página de chamados.



Fonte: Próprio autor.

Com uma conta cadastrada na plataforma de desenvolvimento bubble é possível criar uma aplicação web com conectividade com banco de dados externos, justamente o que é necessário para o projeto em questão, com isso, com a interface inicial do programa, foi criado um novo APP na opção dada como mostra na Figura 13.

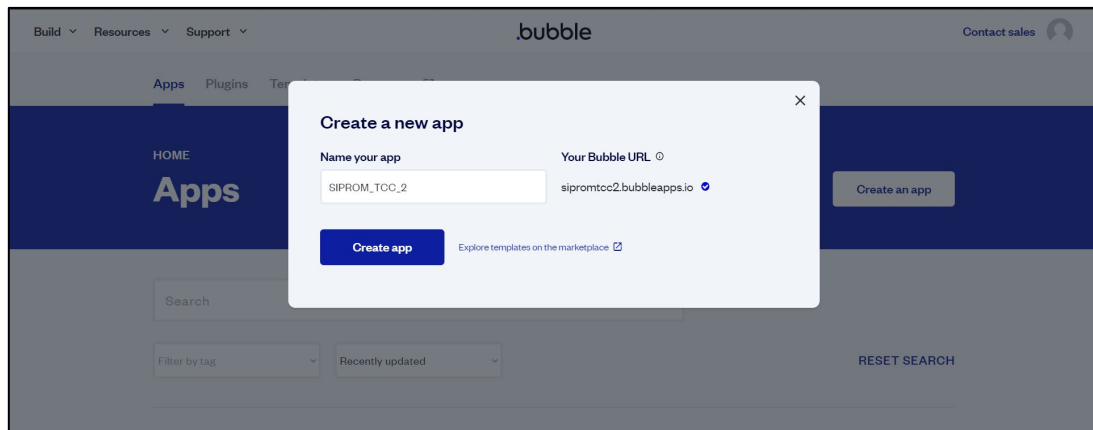
Figura 13: Interface inicial do Bubble.



Fonte: Próprio autor.

Com a opção selecionada de criação de um projeto, é pedido a nomeação dele, onde é obrigatório que seja uma identificação única, e que automaticamente é criada uma Url do seu aplicativo, como é mostrado na Figura 14.

Figura 14: Identificação do projeto web.



Fonte: Próprio autor.

Também é possível criar a aplicação através de um modelo criado por outro usuário, pois o bubble possui uma aplicação *open source*, porém o projeto em questão necessita de funcionalidades totalmente novas que não são encontradas na vasta biblioteca do site.

Para que a aplicação seja única e tenha aspectos profissionais, foi necessário a criação de um ícone com a identidade do projeto, o qual pode ser visto na Figura 15.

Figura 15: ícone da aplicação do projeto.



Fonte: Próprio autor.

É selecionado na plataforma a alteração do ícone que será mostrado na guia de procura do navegador onde o usuário irá dialogar com a página web, como mostrado na Figura 16.

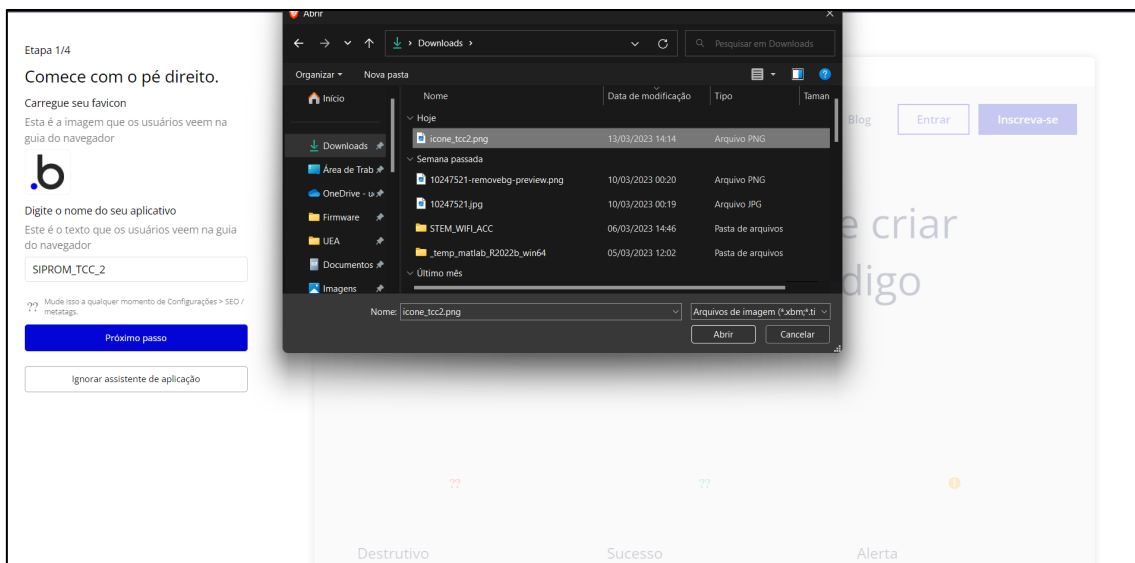
Figura 16: Alteração do ícone padrão do aplicativo.



Fonte: Próprio autor.

Logo após selecionado foi preciso fazer o *upload* do ícone em formato svg do computador, onde se encontrava o arquivo da arte criada como mostrado na Figura 17.

Figura 17: Upload do ícone para o aplicativo.



Fonte: Próprio autor.

Definimos a fonte padrão do site para amostragem de informações de dados para o usuário do aplicativo como mostrado na Figura 18.

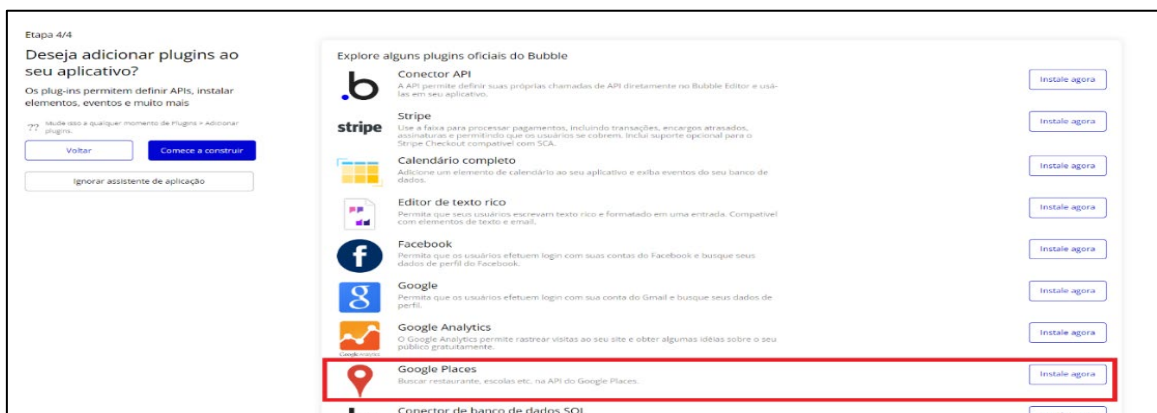
Figura 18: Definição de todas as fontes do aplicativo.



Fonte: Próprio autor.

Como formato de modelos de atualização da plataforma, é possível adicionar *plugins* de extensões com funcionalidades apropriadas para projetos dependendo das necessidades do aplicativo, nesse momento foi adicionado apenas o *google places*, já que o sistema irá intercalar dados de localização do google, como mostrado na Figura 19.

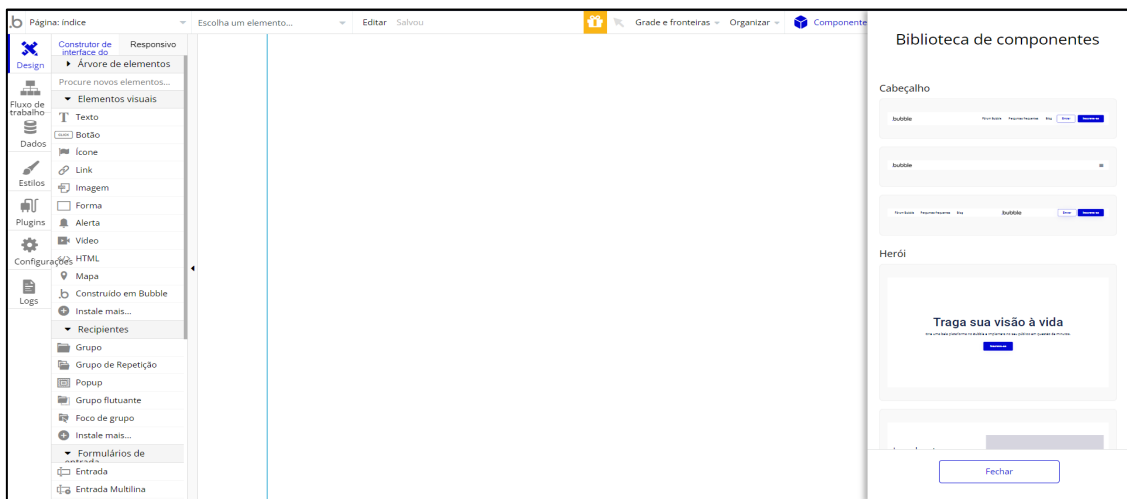
Figura 19: Página de adição de extensões.



Fonte: Próprio autor.

Logo após a instalação de extensões para o aplicativo, a interface está pronta para atender o desenvolvimento do protótipo, onde pode se encontrar ícones de funcionalidades envolvendo elementos visuais, formulários, configurações, fluxo de trabalho que está relacionado a programação e ferramentas de edição, além de uma biblioteca vasta de componentes criados por outros usuários do bubble, como mostrado na Figura 20.

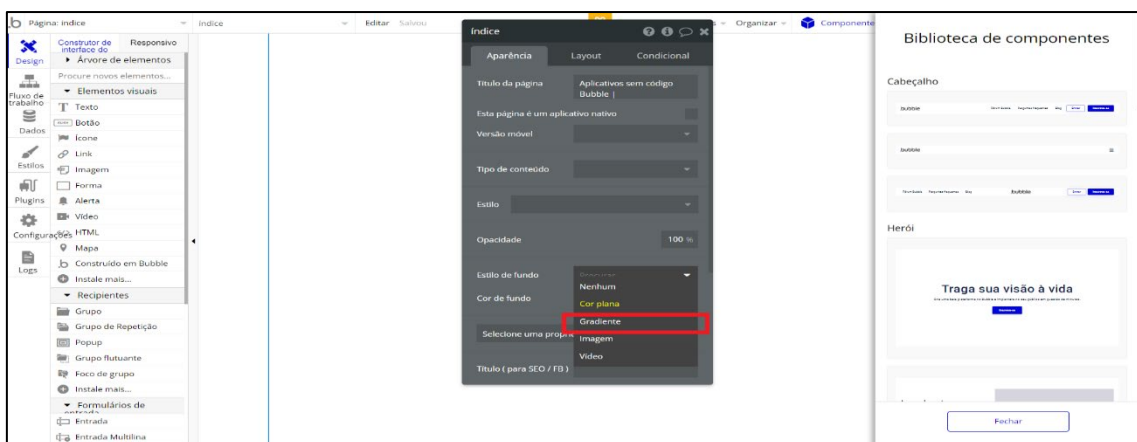
Figura 20: Interface de desenvolvimento do bubble.



Fonte: Próprio autor.

Para melhor visualização dos dados foi escolhida uma formatação de gradiente com cor de fundo como mostrado na Figura 21.

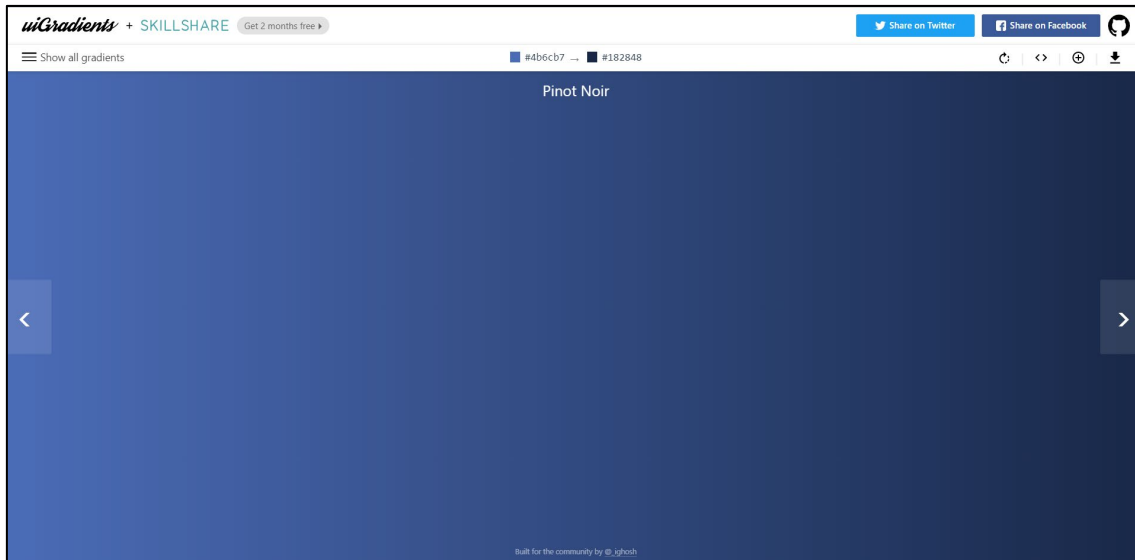
Figura 21: Escolha do formato de cor de fundo aplicativo.



Fonte: Próprio autor.

Com a opção de gradiente escolhida, foi necessário introduzir duas cores em hexadecimal com auxílio da plataforma *uigradients* que gera esse tipo de informação e manifesta em sua tela o resultado da aglutinação de cor como mostrado na Figura 22.

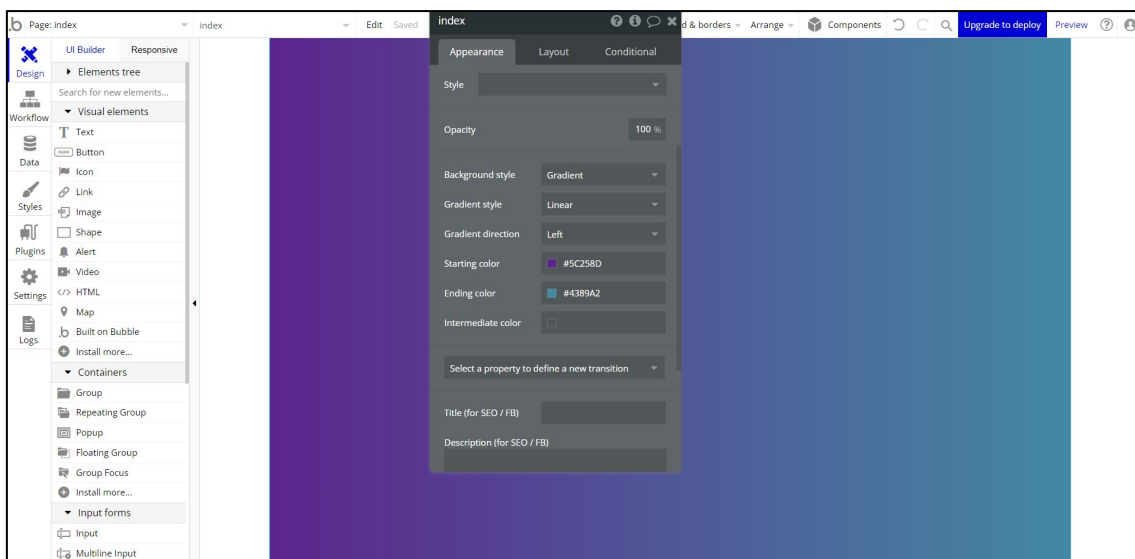
Figura 22: Plataforma do uigradients.



Fonte: Próprio autor.

Logo depois de escolhidas as cores, foi copiado e colado os códigos hexadecimais na plataforma do aplicativo como mostrado na Figura 23.

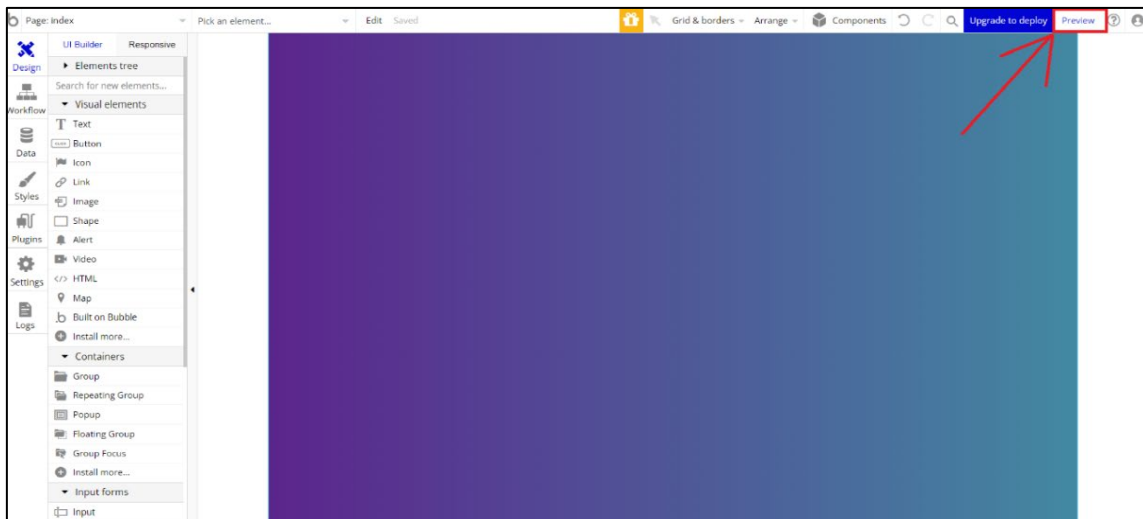
Figura 23: Configuração das cores gradientes do site.



Fonte: Próprio autor.

Para a visualização em tempo real de como está a aplicação o Bubble possibilita ao desenvolvedor uma amostragem, através da função de pré-visualização, como mostrado na figura 24.

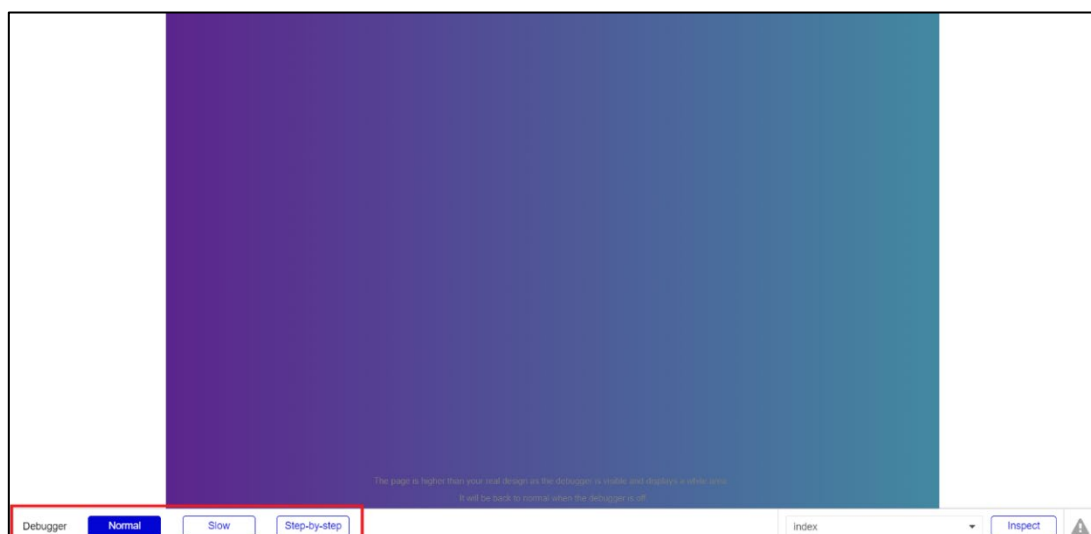
Figura 24: Botão de pré-visualização.



Fonte: Próprio autor.

Logo após a função escolhida, é aberta uma nova aba no navegador mostrando em formato real como a aplicação está no momento do desenvolvimento, em que é possível analisar as funções das programações criadas pelo método *debugger* na forma lenta e passo a passo como mostrado na Figura 25.

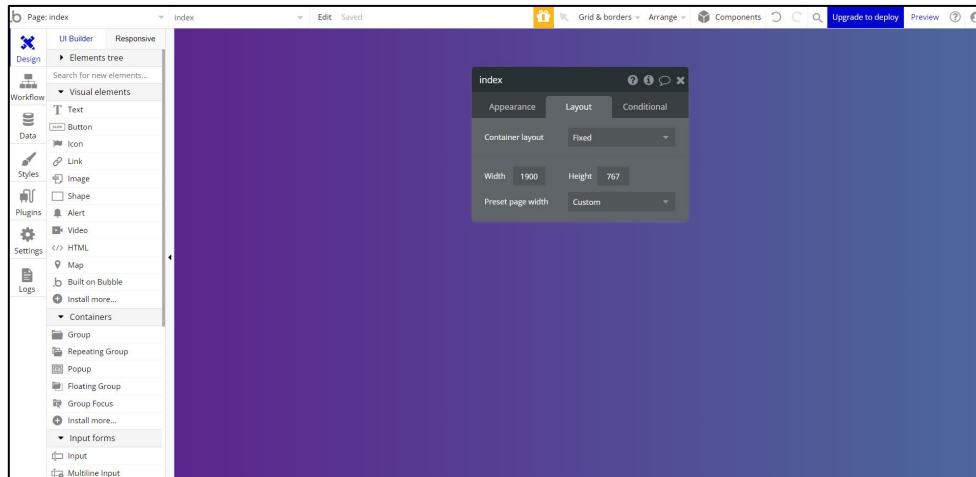
Figura 25: Emulação do aplicativo em tempo real.



Fonte: Próprio autor.

Logo depois de escolhido o formato estético do fundo da aplicação, foi configurado as dimensões do aplicativo para um formato de expansão da tela inteira em editar > layout > e dimensionado a largura para 1900 pixels, como mostrado Figura 26.

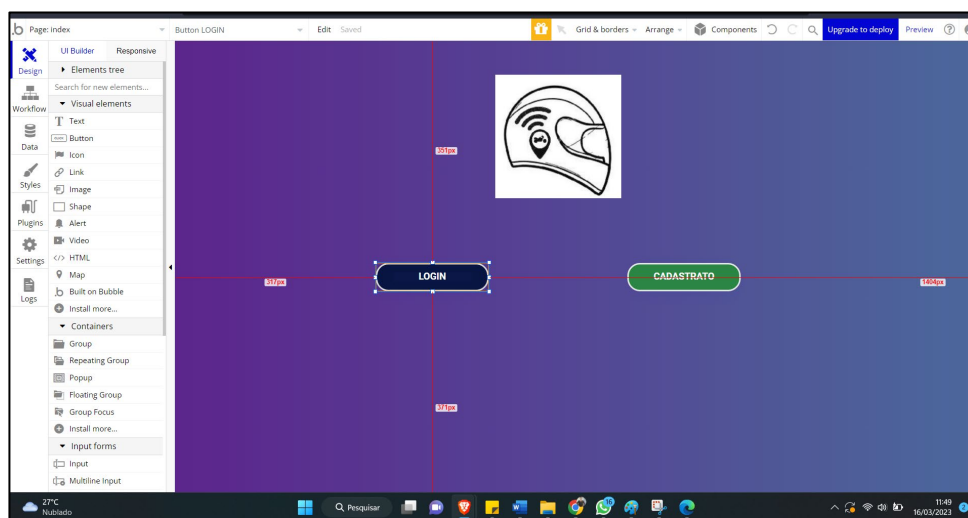
Figura 26: Dimensionamento da tela do aplicativo.



Fonte: Próprio autor.

Assim que finalizado o processo de configuração da interação com os usuários, foi criado uma tela de login e cadastro de pessoas que irão utilizar a aplicação, como mostrado na Figura 27, onde foi inserido campos de entrada de texto em relação a e-mail e senha em um balão mostrado na tela inicial como mostrado Figura 27.

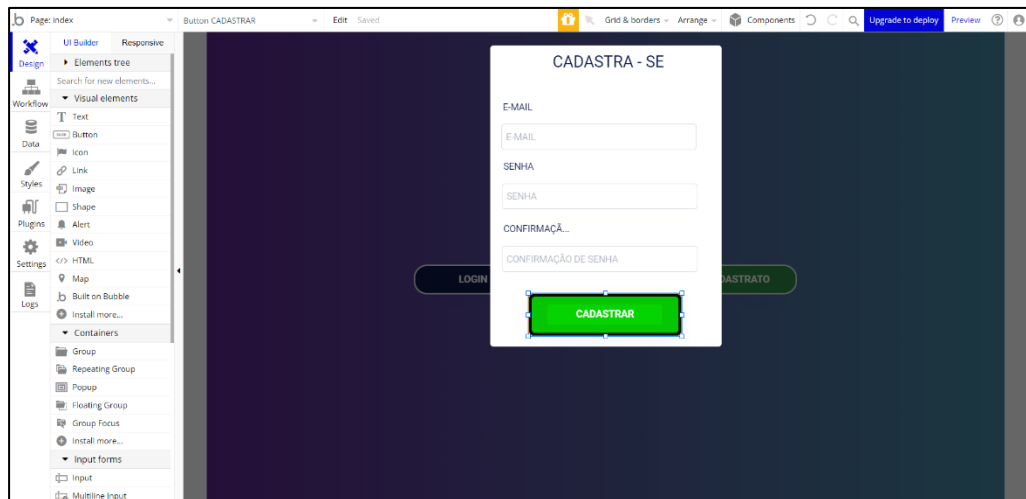
Figura 27: Tela de login e cadastro do aplicativo.



Fonte: Próprio autor.

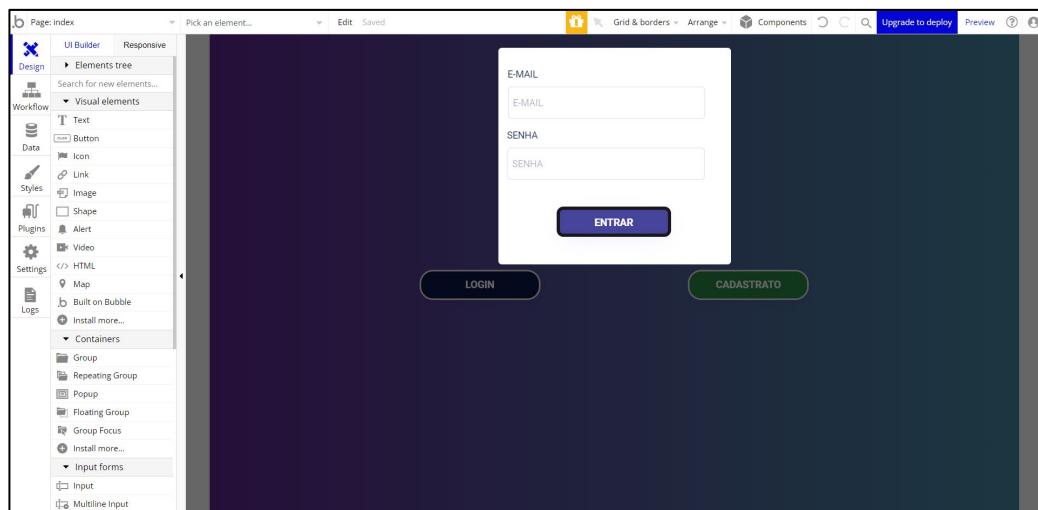
Assim sendo, logo que umas das funções é escolhida, foi mostrado uma tela referente ao botão desejado, onde o usuário poderá se cadastrar ou fazer o login em sua conta, como é mostrado nas Figura 28 e Figura 29.

Figura 28: Minitela de cadastro.



Fonte: Próprio autor.

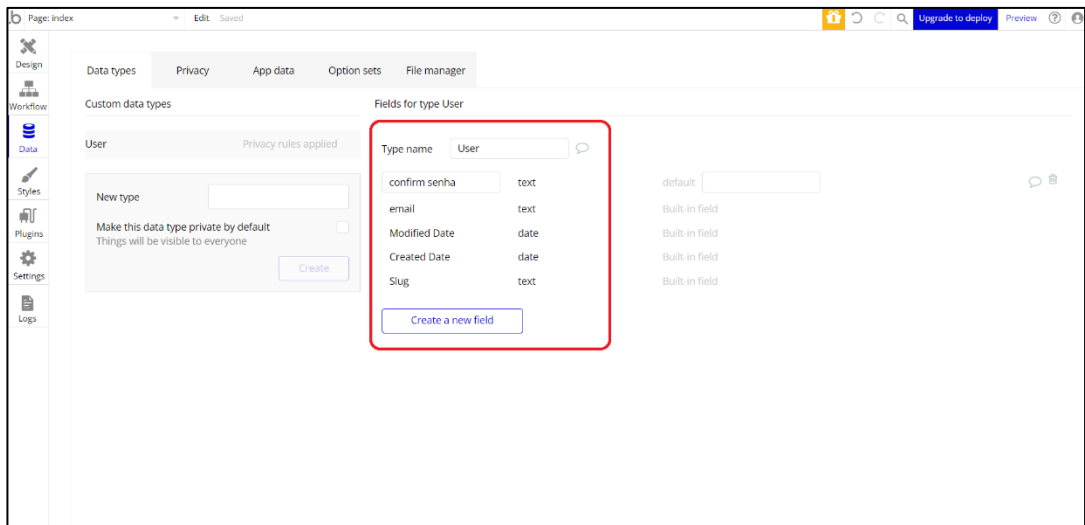
Figura 29: Minitela de login.



Fonte: Próprio autor.

Para que haja funcionalidade na etapa de cadastro e login, é necessário criar um campo de armazenamento dentro do próprio bubble na função data > tupe user como mostrado na Figura 30.

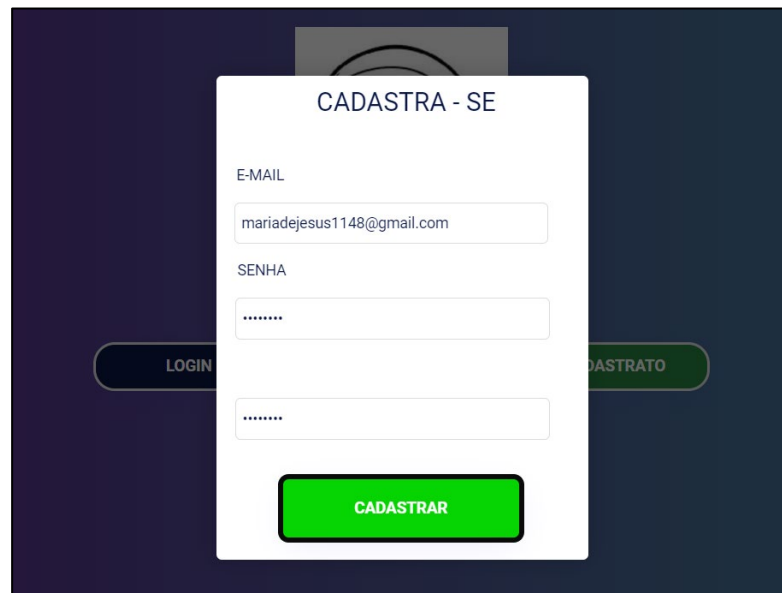
Figura 30: Campo de usuário criado.



Fonte: Próprio autor.

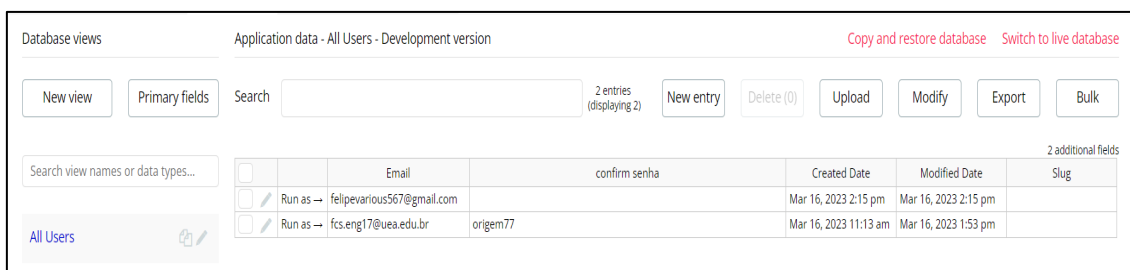
Logo depois foi validado o cadastro de novo usuário fazendo um login fictício de email e senha bem como confirmação de senha, e analisado no banco de dados. Como sistema de segurança o bubble já esconde do desenvolvedor as senhas de usuários como mostrado nas Figura 31 e Figura 32.

Figura 31: Teste de cadastro.



Fonte: Próprio autor.

Figura 32: Validação de cadastro de usuários.



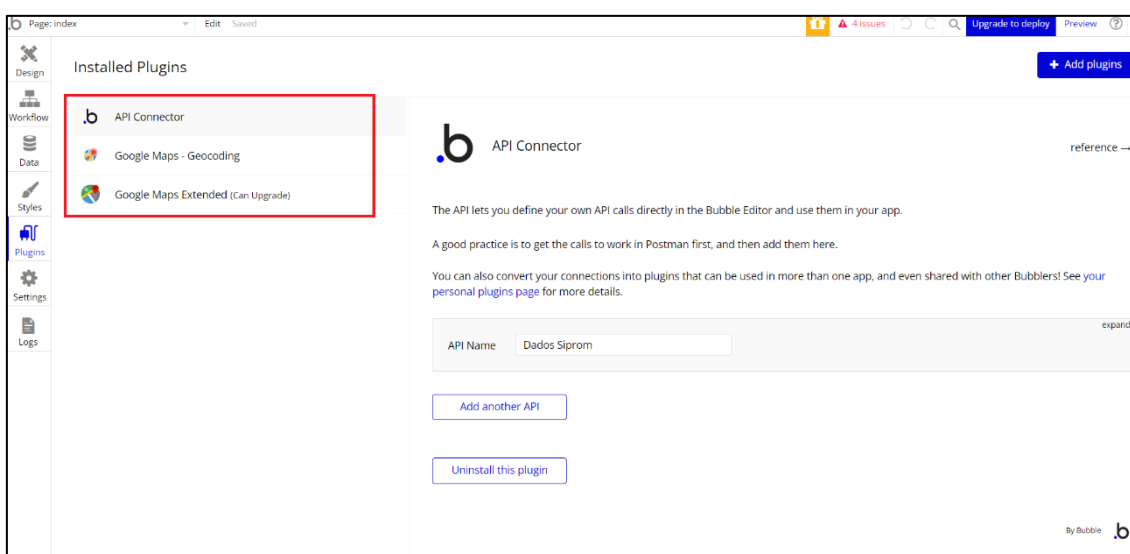
The screenshot shows a database management interface for 'Application data - All Users - Development version'. It includes a search bar, a table with columns for Email, confirm senha, Created Date, Modified Date, and Slug, and a sidebar with 'All Users' selected. The table contains two entries:

	Email	confirm senha	Created Date	Modified Date	Slug
Run as →	felipevarious567@gmail.com		Mar 16, 2023 2:15 pm	Mar 16, 2023 2:15 pm	
Run as →	fcs.eng17@uea.edu.br	origem77	Mar 16, 2023 11:13 am	Mar 16, 2023 1:53 pm	

Fonte: Próprio autor.

Para a integração de comunicação com o sistema físico, é necessário adicionar extensões relevantes com essas necessidades, além de especificações de amostragem de informações para usuários do aplicativo. Para a adição dessas funcionalidades foram escolhidas: API Connector, que integra chaves de comunicação com banco de dados, Google Maps, que identifica e traduz informações de localização e Google Maps Extended que faz a amostragem do local traduzido, como mostra a Figura 33.

Figura 33: Configuração de extensões do bubble.



The screenshot shows the 'Installed Plugins' section in the Bubble Editor. A red box highlights the 'API Connector', 'Google Maps - Geocoding', and 'Google Maps Extended (Can Upgrade)' plugins. The 'API Connector' plugin is selected, showing its configuration page with the API Name 'Dados Siprom' and buttons for 'Add another API' and 'Uninstall this plugin'.

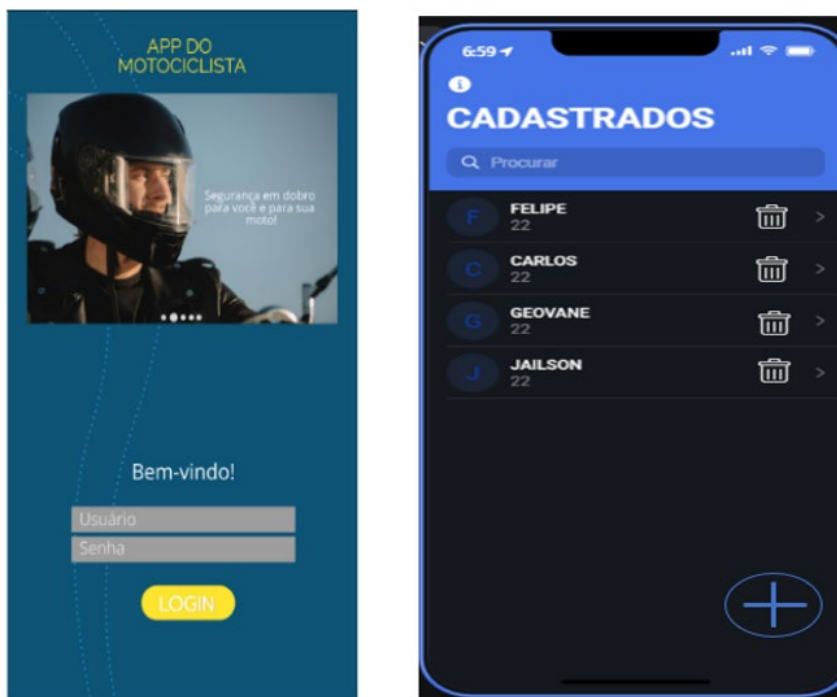
Fonte: Próprio autor.

8.2. DESENVOLVIMENTO APLICATIVO *MOBILE*

Primeiramente, foram feitos escopos de projeto em relação ao aplicativo *mobile*, em relação as páginas de interação com o usuário, para ser possível a idealização de componentes os quais permitissem a administração de usuários cadastros no sistema físico, bem como a captação de informações do microcontrolador em relação a detecção

de acidentes e a comutação de dados via wi-fi, de forma que sempre houvesse notificação por áudio para a validação de comandos do aparelho e funcionalidades do protótipo. Sendo esses esboços mostrados na Figura 34.

Figura 34: Esboço das telas do aplicativo *mobile*.

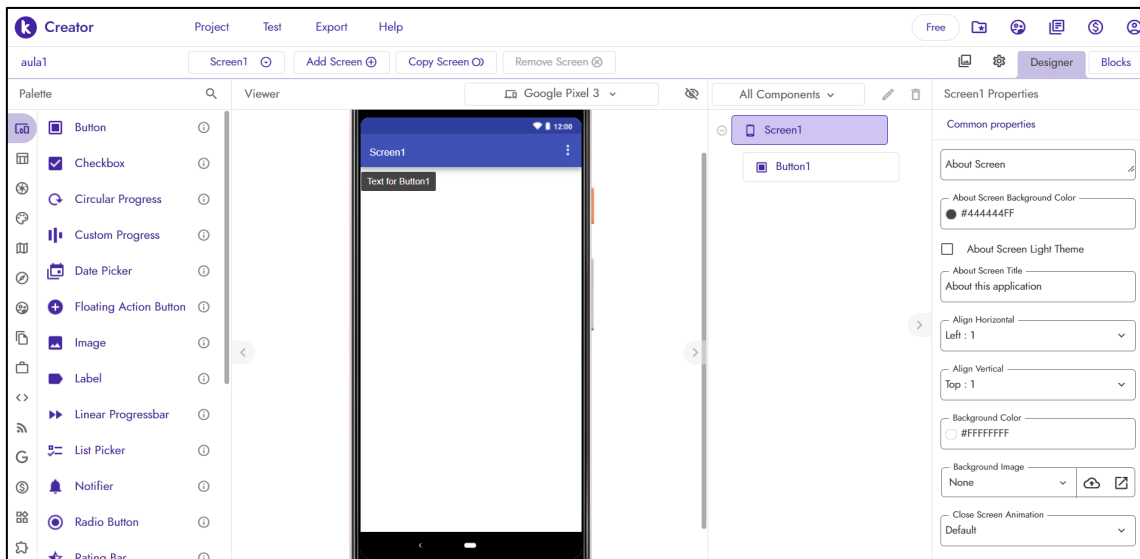


Fonte: Próprio autor.

Para o desenvolvimento da aplicação *mobile* foi utilizado a plataforma de desenvolvimento Kodular, que permite conectividade com diversos sistemas, inclusive conexão com bancos de dados externos, além de ser possível a utilização de sensores nativos do próprio dispositivo, em que, essas funcionalidades são adequadas para a integração com o projeto em questão.

Podemos analisar a interface inicial do Kodular para desenvolvimento de aplicativos *mobile* na Figura 35.

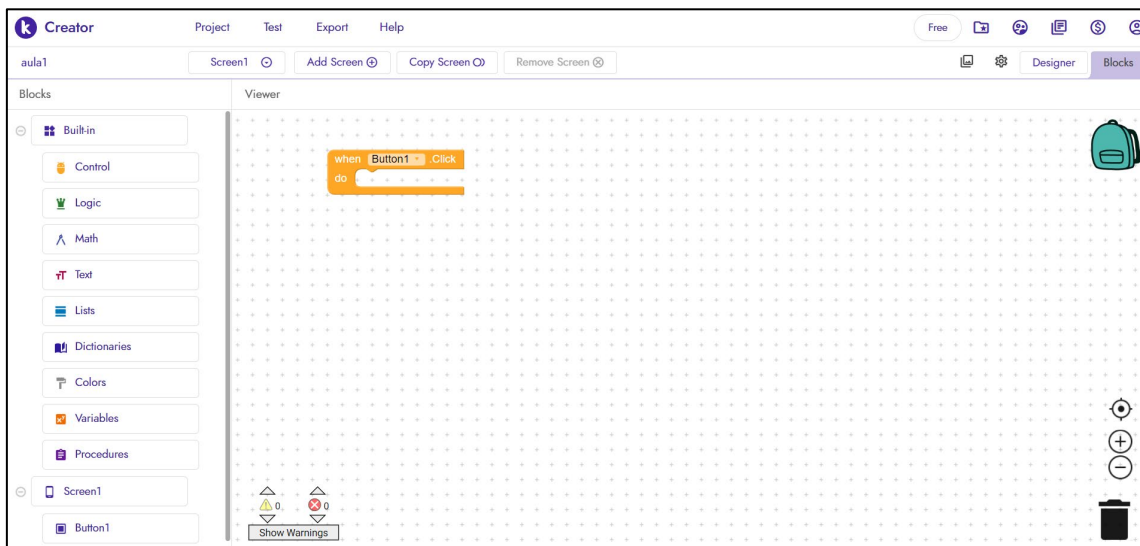
Figura 35: Interface da plataforma Kodular.



Fonte: Próprio autor.

Nessa plataforma existe a possibilidade de intercalar, componentes visuais do aplicativo com programação, na qual é feita por blocos, onde temos uma aba específica para cada tela do sistema, como mostrado na Figura 36.

Figura 36: Páginas de blocos de programação.

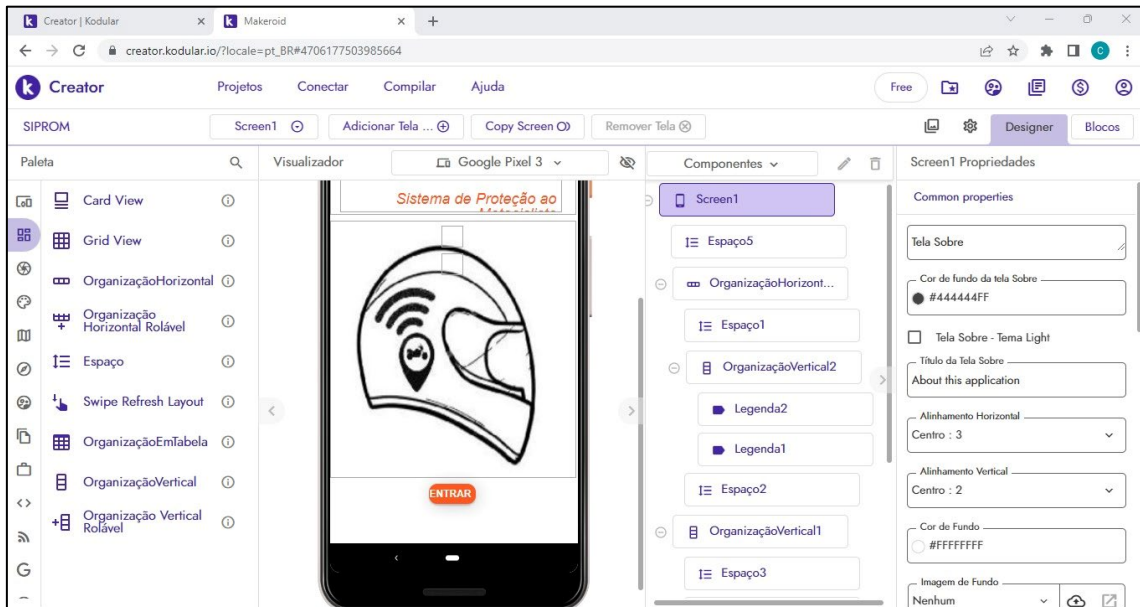


Fonte: Próprio autor.

Inicialmente, como fase inicial do projeto de aplicativo para, foi feito a tela de login de usuários utilizando a conta google do proprietário do dispositivo, ou seja, do

celular, onde é apresentado o campo de entrada e a imagem de ícone do aplicativo como mostrado na Figura 37.

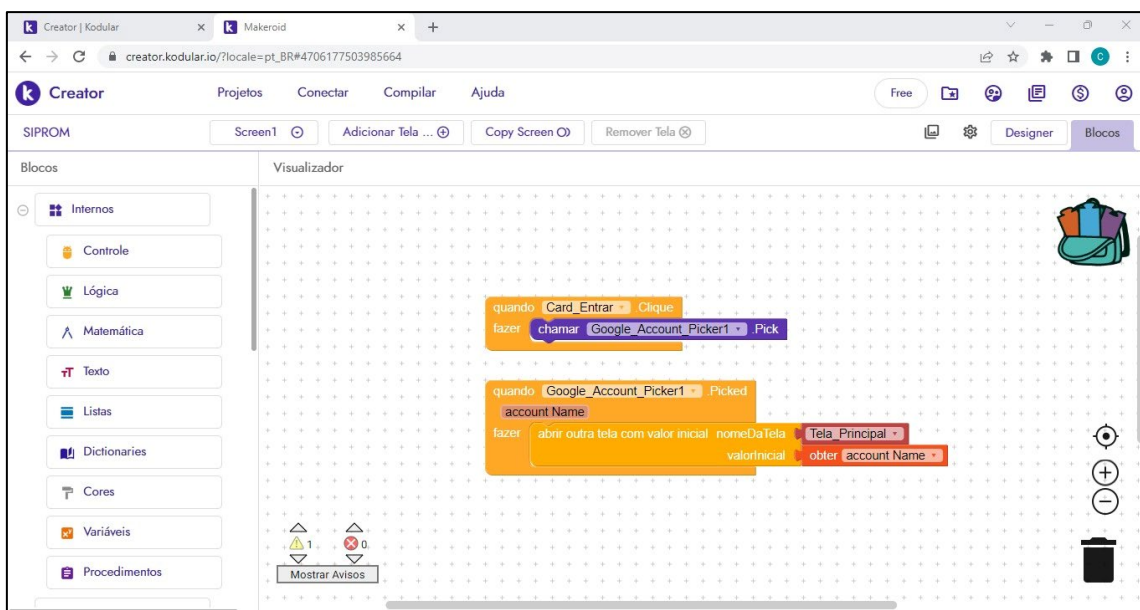
Figura 37: Tela de login do aplicativo *mobile*.



Fonte: Próprio autor.

Logo, também foi necessário desenvolver a programação de login por blocos dessa mesma página para seu funcionamento com a conta google desejada, como mostrado na Figura 38.

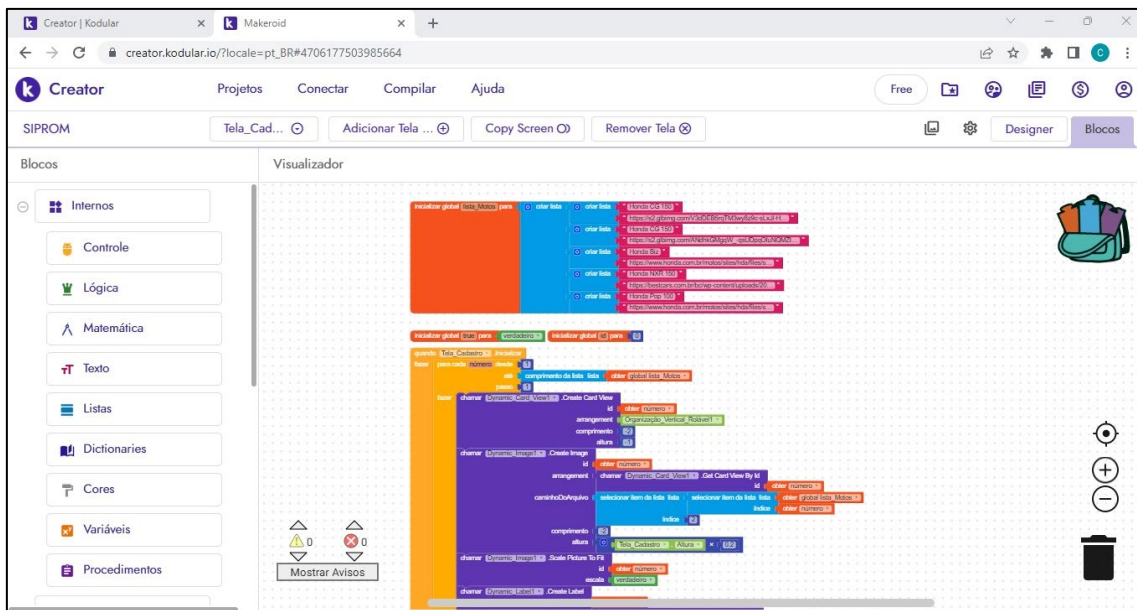
Figura 38: Programação da página de login.



Fonte: Próprio autor.

Sendo assim, foi construído posteriormente a interface de comandos que seriam enviados pelos usuários do aparelho celular para o sistema físico encontrado na motocicleta, onde foram usados botões que possuíam caracteres estáticas que seriam enviadas assim que as teclas fossem selecionadas na tela do *smartphone* como mostra a Figura 39.

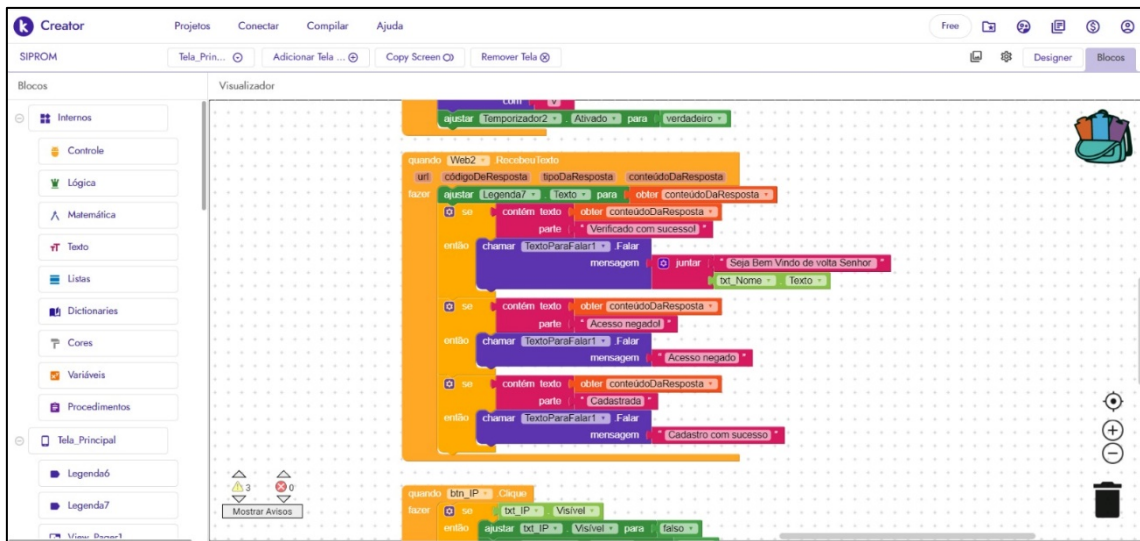
Figura 39: Programação dos comandos de envios dos usuários.



Fonte: Próprio autor.

Assim sendo, para finalização do desenvolvimento do aplicativo, foi criada as funções de notificações de funcionalidades por voz, onde o aparelho celular resgata a leitura de palavras com auxílio de fala padrão do dispositivo. Para isso foram criadas variáveis de *strings* para cada comando enviado e recebido em relação a aplicação e ao sistema físico, sendo essa função mostrada na Figura 40.

Figura 40: Programação do sistema de notificação por voz.

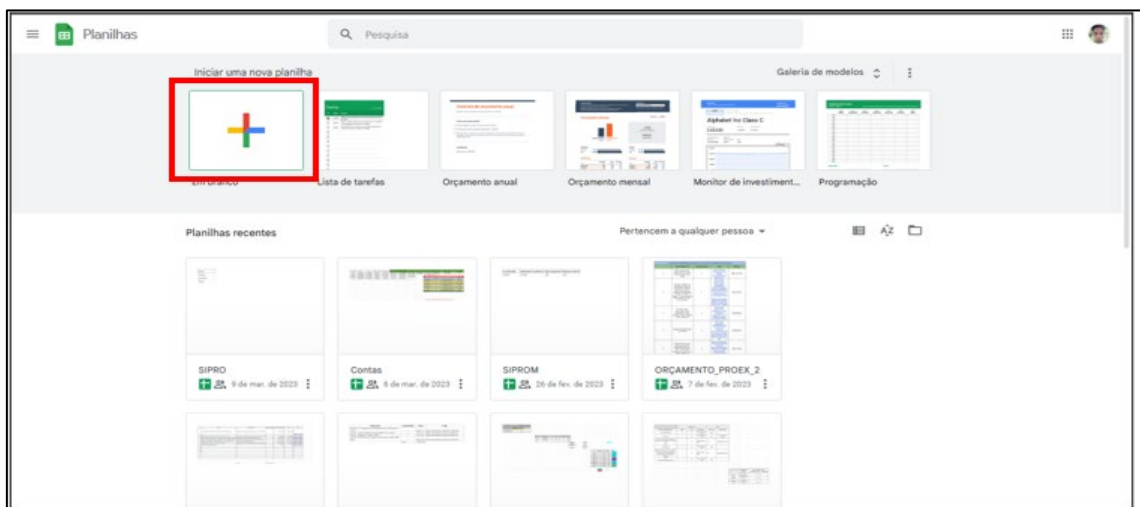


Fonte: Autor próprio.

8.3. BANCO DE DADOS.

Para o uso de banco de dados do sistema, onde irá armazenar as informações de localização bem como as informações de gerenciamento de usuários do veículo, foi utilizado o Google *Sheety*, pois ele possui uma interface de programação de aplicativos, possibilitando a transmissão e recepção de dados via internet coerente ao projeto.

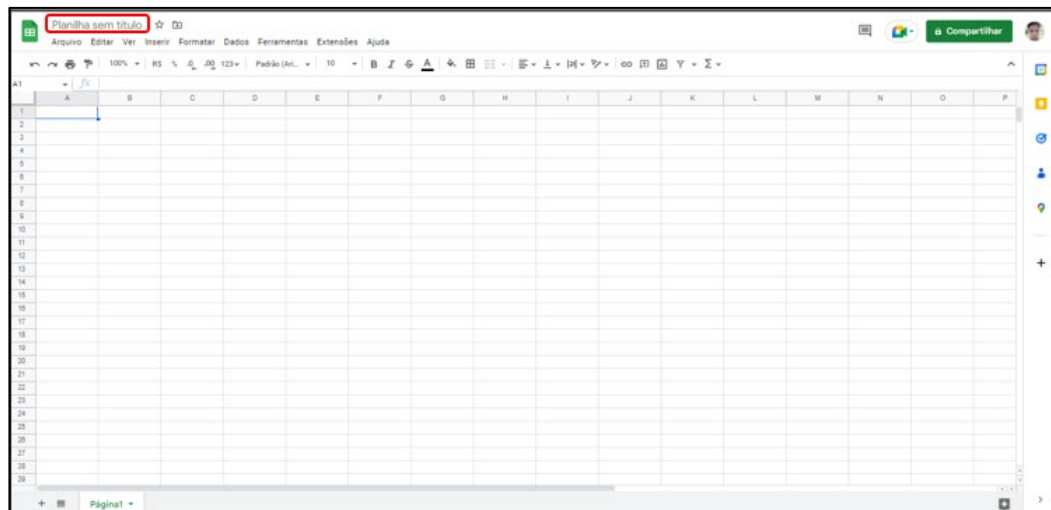
Figura 41: Iniciar uma nova planilha google.



Fonte: Próprio autor.

Logo depois de clicar no comando para criação de uma nova planilha, o google apresenta na próxima tela uma série de linhas e colunas em branco com apenas uma aba, onde será necessário nomeá-la como mostrado na Figura 42.

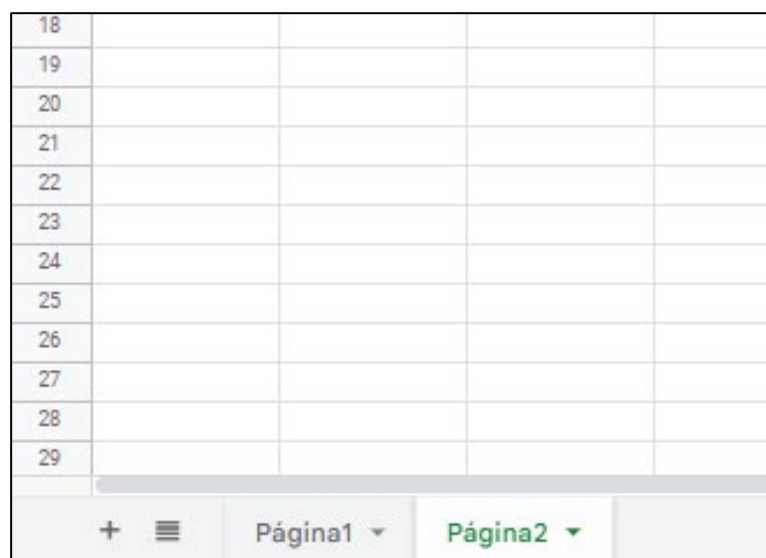
Figura 42: Criação de uma planilha no google sheets.



Fonte: Próprio autor.

Para haver compatibilidade com a função de banco de dados é necessário criar mais uma aba na planilha do projeto como mostrado na Figura 43.

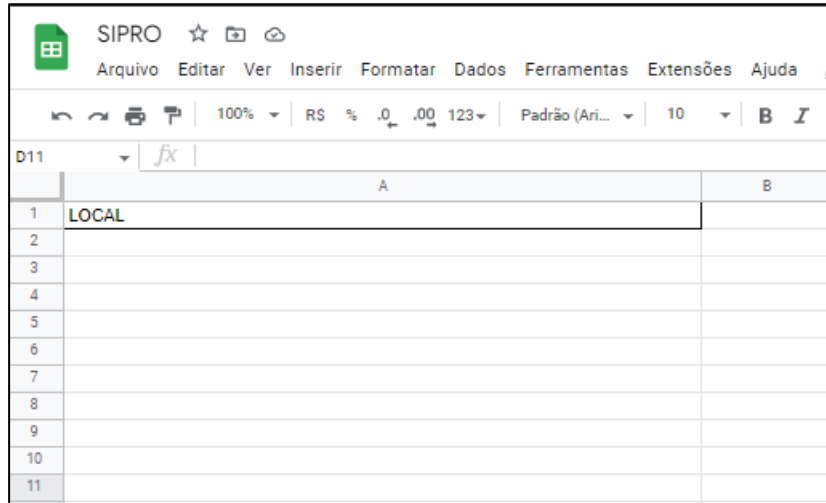
Figura 43: Criação de uma nova aba da planilha.



Fonte: Próprio autor.

Logo após esse processor é necessário criar um campo com identificação para o recebimento da informação relacionada, neste caso, a localização com na Figura 44.

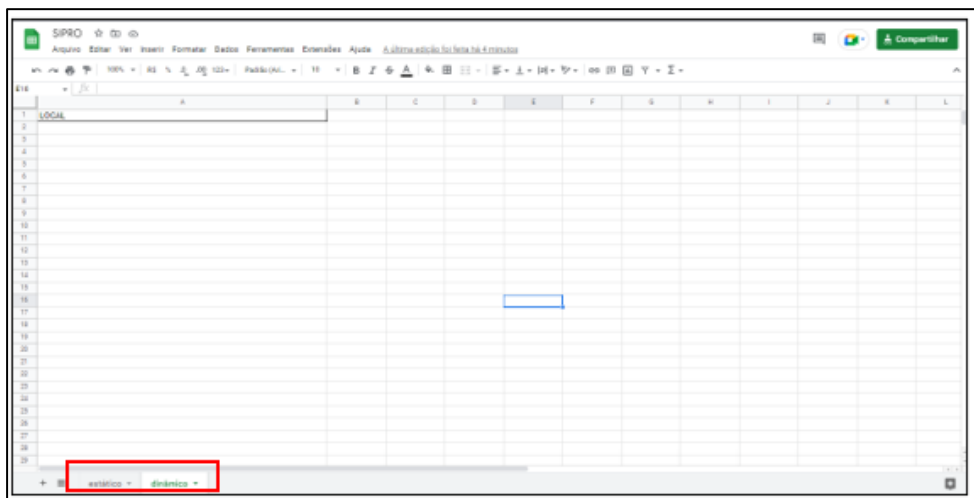
Figura 44: Identificação da coluna de informações manipuladas.



Fonte: Próprio autor.

Também é necessário a nomeação da aba que contém as informações manipulados pelo site, pois quando é criado a intercomunicação com o banco de dados será desenvolvido links para cada página, e nesse caso foi identificado como “dinâmico” como mostrado na Figura 45.

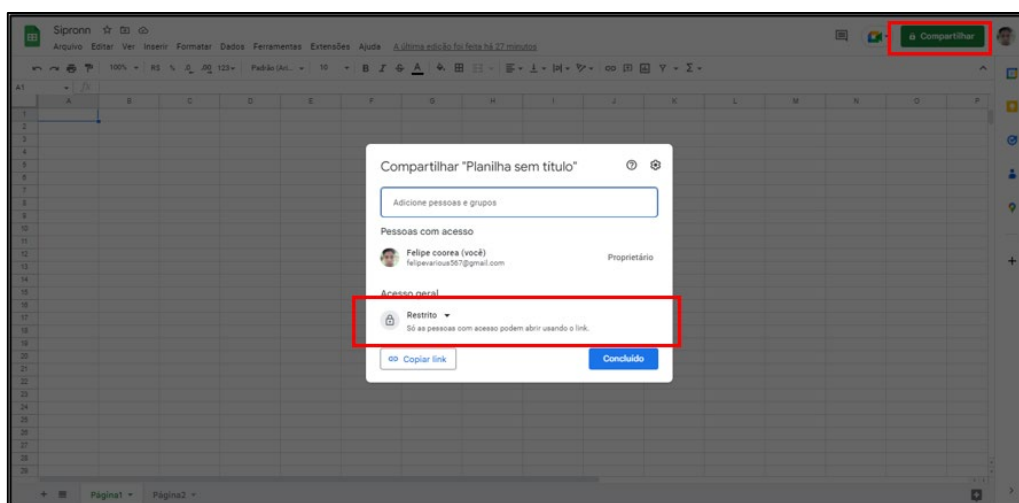
Figura 45: Identificação da aba correspondentes aos dados trabalhados.



Fonte: Próprio autor.

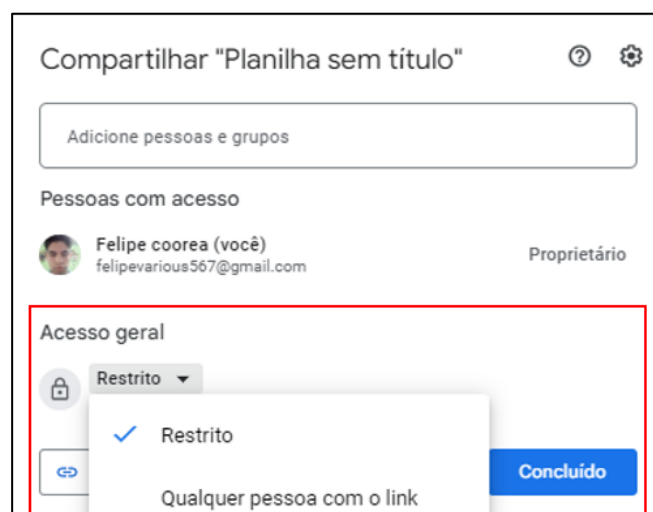
Assim sendo, com o término das configurações em relação a amostragem de informações que chegaram do sistema da planilha utilizada, é imprescindível a configuração de compartilhamento das páginas, em que é formatado de forma restrita, onde apenas o proprietário da conta é capaz de editar os campos de dados, pois é importante para que não haja invasão de usuários indesejados que possam manipular as informações obtidas. Para a edição desse aspecto foi configurado em Compartilhar > Restrito, em que só pessoas com acesso liberado podem alterar a planilha como mostrado nas Figura 46 e Figura 47.

Figura 46: Configuração de segurança para a planilha.



Fonte: Próprio autor.

Figura 47: Configuração da planilha no formato restrito.

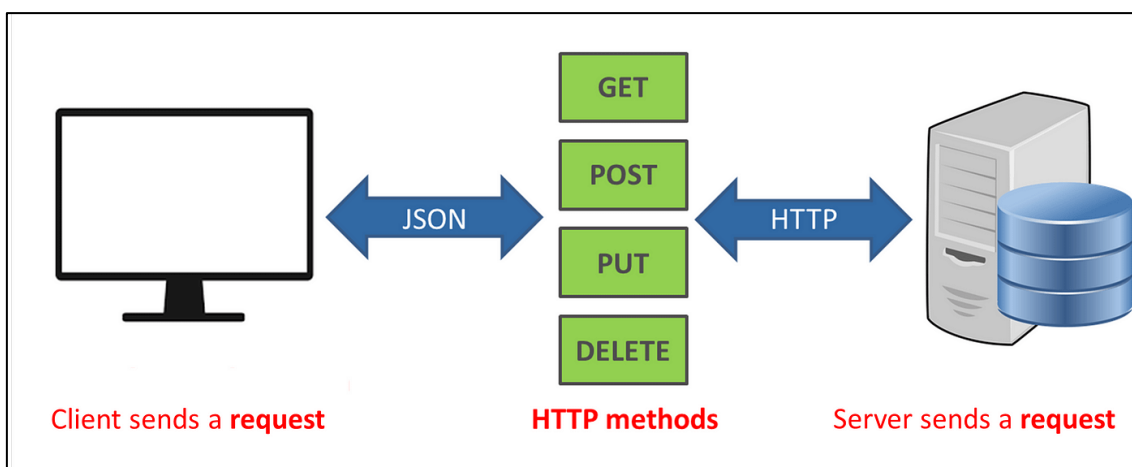


Fonte: Próprio autor.

8.4. INTEGRAÇÃO DO BANCO DE DADOS COM O APLICATIVO WEB

Para a criação do banco de dados do projeto, foi criada uma planilha do google com proteção padrão disponibilizada pela própria empresa, como mostrado anteriormente, sendo necessário ainda criar um processo de integração com o aplicativo web desenvolvido, de forma que foi utilizado o conjunto de protocolos de API para a comunicação entre as plataformas, onde o banco de dados utilizados é comum para o aplicativo *mobile* e a aplicação web. A conexão de bloco de clientes, APIs e banco de dados é manifestada na Figura 48.

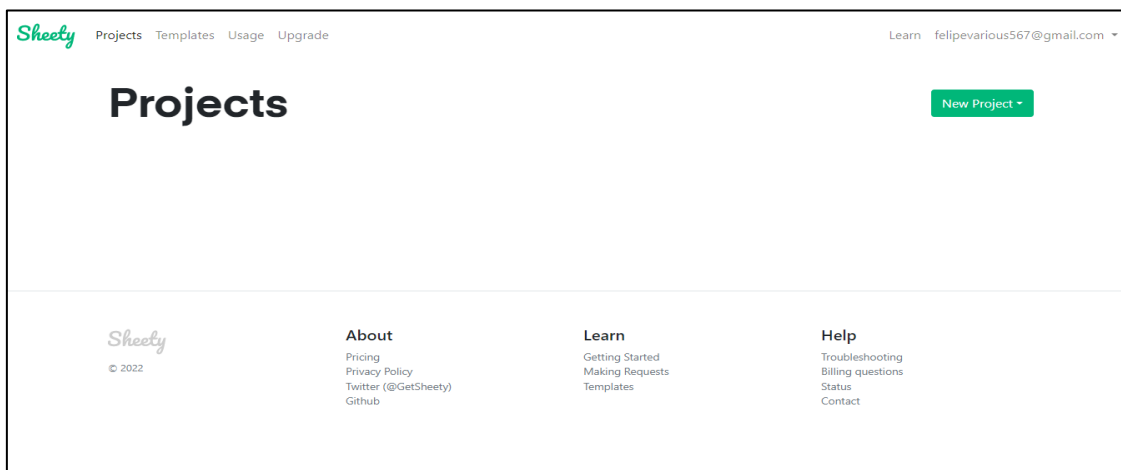
Figura 48: Ilustração do funcionamento de APIs.



Fonte: Próprio autor.

Na figura abaixo é apresentada a interface da plataforma onde iremos criar nosso banco de dados próprio através de uma planilha do google já existente, em que é possível na opção de novo projeto como mostrado na Figura 49.

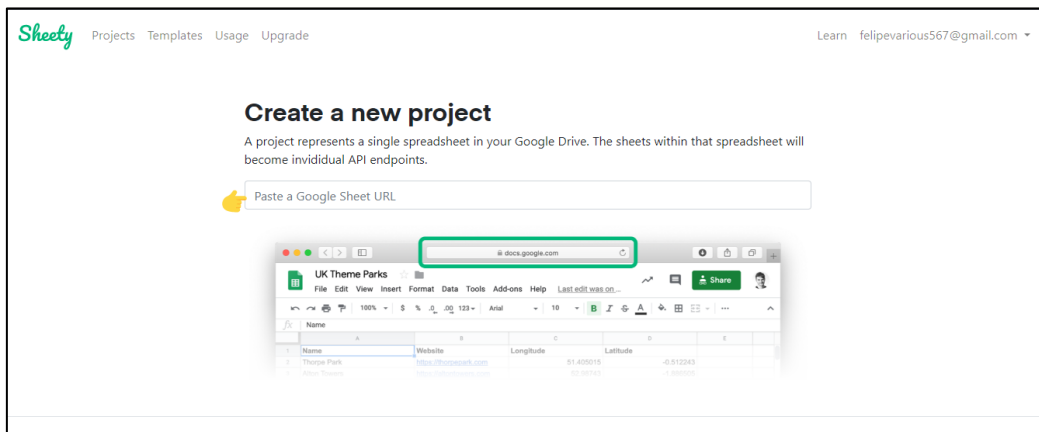
Figura 49: Interface da plataforma de banco de dados.



Fonte: Próprio autor.

Logo, quando a opção de novo projeto foi escolhida é exigido um nome qualquer para o seu banco de dados, assim como a URL da planilha que será usada como mostra na Figura 50.

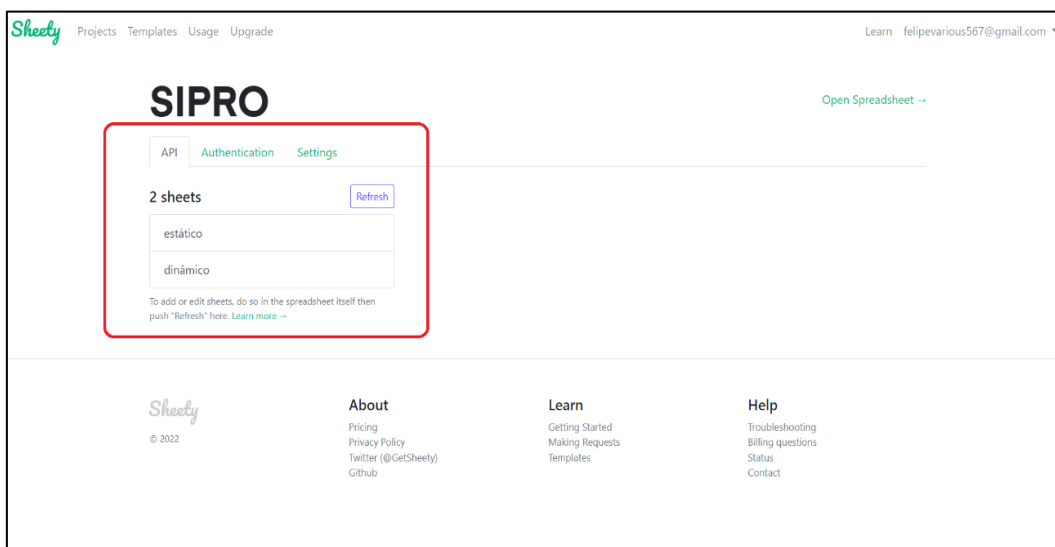
Figura 50: Campo para anexar a URL da planilha que será utilizada.



Fonte: Próprio autor.

Assim sendo, quando são inseridas as informações da planilha google do mesmo usuário proprietário conectado na conta do Sheety, foi desenvolvida a *dashboard* com as configurações de autenticação junto com os *endpoints* relacionados com as abas de informações da planilha google que está sendo usada como banco de dados, como mostrado na Figura 51.

Figura 51: Dashboard da plataforma para configurações de comunicação.



Fonte: Próprio autor.

Como tópico importante desse projeto já falado anteriormente, a segurança de informações é algo imprescindível nos tempos modernos, e para isso ser integrado ao nosso sistema de forma que apenas usuários autorizados possam visualizar as informações transmitidas, foi configurado o método de autenticação básica HTTP para nossas requisições, que é um método simples de autenticação para conexões HTTP, que envolve a apresentação de um nome de usuário e senha ao servidor em uma solicitação HTTP. É uma forma simples de autenticação que envolve a codificação base64 dos dados de autenticação em uma string, que é então incluída no cabeçalho da solicitação HTTP.

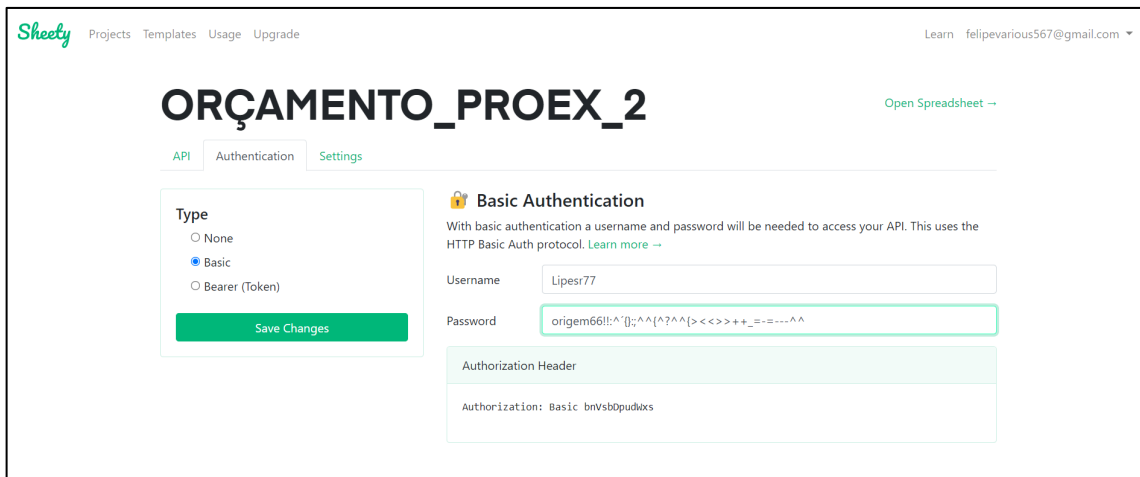
A autenticação básica HTTP não é considerada uma forma segura de autenticação, pois a codificação base64 é facilmente decodificável e as informações de autenticação podem ser facilmente interceptadas por um invasor. Portanto, é recomendável usar a autenticação HTTPS em vez da autenticação HTTP básica para garantir a segurança das informações de autenticação.

- O processo de autenticação básica HTTP é o seguinte:
- O cliente faz uma solicitação HTTP ao servidor.
- O servidor responde com um código de status 401 não autorizado e inclui um cabeçalho WWW-Authenticate na resposta.
- O cabeçalho WWW-Authenticate informa ao cliente que a autenticação é necessária e que tipo de autenticação deve ser usada. No caso da autenticação básica, o cabeçalho seria "Basic realm="nome da área de autenticação"".
- O cliente responde enviando outra solicitação HTTP ao servidor com um cabeçalho Authorization incluindo as informações de autenticação codificadas em base64. O cabeçalho Authorization deve ser definido da seguinte forma: "Basic " + base64_encode(username + ":" + password)".
- O servidor verifica as informações de autenticação e, se elas estiverem corretas, processa a solicitação do cliente.

Em resumo, a autenticação básica HTTP é um método simples de autenticação que é adequado para ambientes internos e de teste. No entanto, em ambientes de produção, é recomendável usar métodos mais seguros de autenticação, como a autenticação baseada em token ou a autenticação OAuth.

Logo são inseridas as informações de usuário e senha para essa autenticação como mostrado na Figura 52

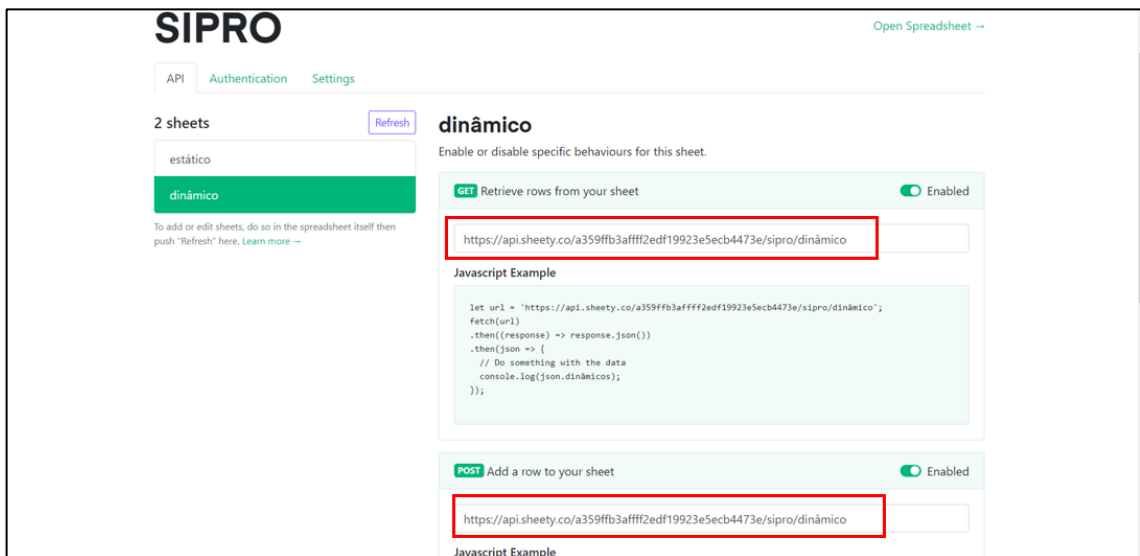
Figura 52: Autenticação básica HTTP do sistema.



Fonte: Próprio autor.

Sendo assim, foram analisados os *endpoints* gerados pela plataforma, em se tratando das informações dinâmicas que iremos utilizar como mostrado na Figura 53.

Figura 53: URLs de requisição do sistema.



Fonte: Próprio autor.

Como na aplicação Web existe somente a requisição de visualização de dados que são imputados no banco de dados do sistema, onde não é necessário modificação ou criação de linhas de informações, iremos apenas utilizar a URL de GET, que solicita um recurso específico do servidor e retorna uma resposta que contém as informações solicitadas.

A resposta da API para uma solicitação GET normalmente inclui os dados solicitados em um formato específico, como JSON, XML ou CSV, dependendo da API. Em geral, a

resposta da API também incluirá um código de status HTTP, que indica se a solicitação foi bem-sucedida ou se ocorreu algum erro.

Logo, foi habilitado a funcionalidade da URL de GET para ser utilizada no sistema, em que o próprio site já mostra as linhas de códigos em Javascript, sendo essas configurações mostradas na Figura 54.

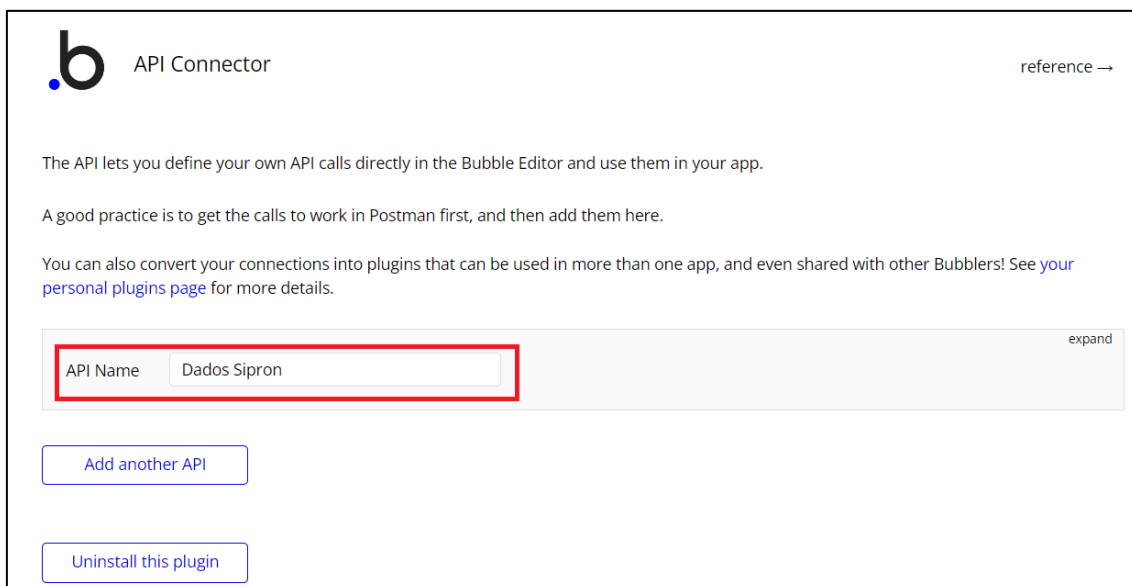
Figura 54: Função de GET habilitada.



Fonte: Próprio autor.

Com a extensão de APIs conector do bubble já instalada, foi necessário nomear a chamada que iremos utilizar “Dados Sipro” como mostrado na Figura 55

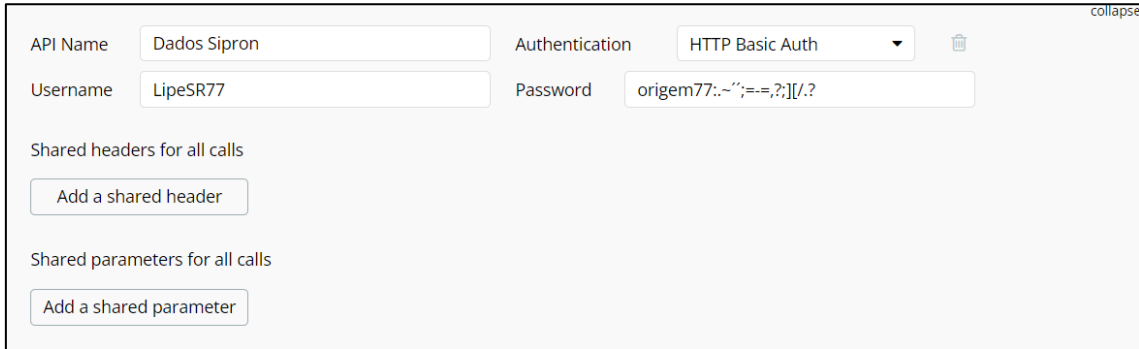
Figura 55: Identificação padrão da API.



Fonte: Próprio autor.

Logo depois, é configurado no bubble o tipo de autenticação HTTP básico, imputando o nome de usuário e a senha como mostrado na Figura 56.

Figura 56: Configuração de autenticação da API do bubble.

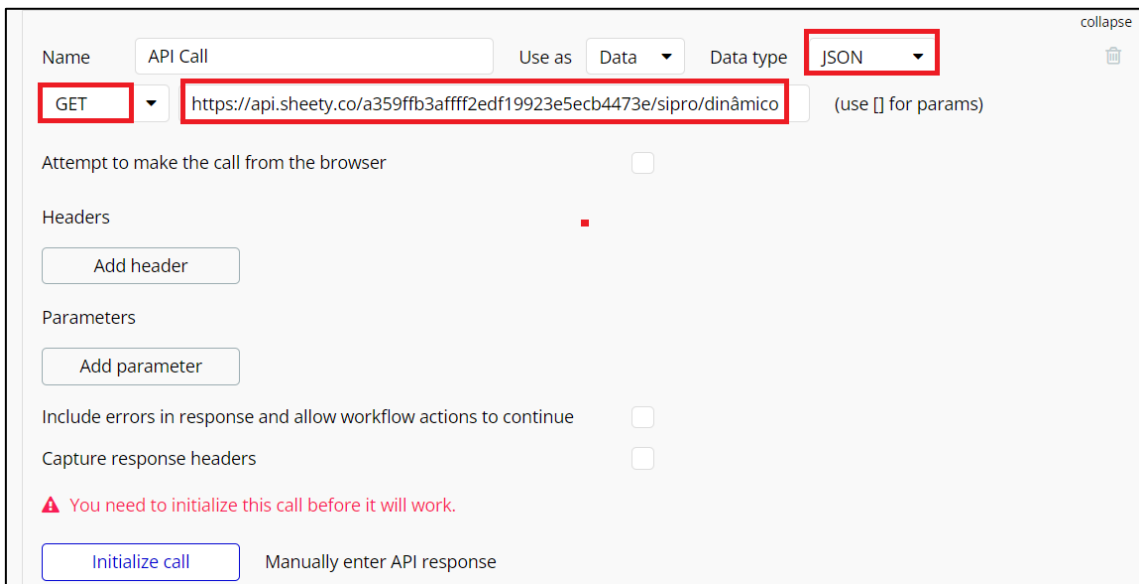


The screenshot shows the API configuration interface in Bubble. The 'API Name' is 'Dados Sipron'. The 'Authentication' is set to 'HTTP Basic Auth'. The 'Username' is 'LipeSR77' and the 'Password' is 'origem77:~";=-,?;][/.?'. Below these fields are sections for 'Shared headers for all calls' and 'Shared parameters for all calls', each with an 'Add a shared header/parameter' button.

Fonte: Próprio autor.

Assim sendo, também foi preciso configurar o tipo de requisição que seria utilizada pelo aplicativo, no caso GET, já falada anteriormente, bem como o formato de chegada da mensagem, a qual é padrão JSON, e pôr fim a URL correlacionada aos parâmetros verificados, sendo essas configurações mostradas na Figura 57.

Figura 57: Configuração de parâmetros de chamadas.



The screenshot shows the API configuration interface in Bubble. The 'Name' is 'API Call'. The 'Use as' is 'Data' and the 'Data type' is 'JSON'. The 'Method' is 'GET' and the 'URL' is 'https://api.sheety.co/a359ffb3affff2edf19923e5ecb4473e/sipro/dinâmico'. There are buttons for 'Add header' and 'Add parameter'. A warning message says 'You need to initialize this call before it will work.' and there is an 'Initialize call' button.

Fonte: Próprio autor.

Com todas as configurações de comunicação de dados feitas em relação ao banco de dados e ao aplicativo web, foi inicializado uma chamada com a requisição de leitura para as informações coletadas, sendo esses dados mostrados em uma página do próprio bubble como mostrado na Figura 58.

Figura 58: Recebimento em formato JSON dos dados da planilha.

```
{
  "dinâmico": [
    {
      "local": "Parque Dez de Novembro, Manaus - AM, 69055-111",
      "id": 2
    },
    {
      "local": "Parque Dez de Novembro, Manaus - AM, 69055-111",
      "id": 3
    },
    {
      "local": "Parque Dez de Novembro, Manaus - AM, 69055-111",
      "id": 4
    },
    {
      "local": "Parque Dez de Novembro, Manaus - AM, 69055-111",
      "id": 5
    }
  ]
}
```

Fonte: Próprio autor.

Como a informação de localização vai ser mostrada no mapa padrão do google, foi necessário atualizar o tipo de informação de texto para endereço de *maps*, como mostrado na Figura 59.

Figura 59: Formatação de dados para localização do google.

Returned values - Dados

You can modify the data types that are returned by the call. This affects how you can use the data in Bubble. If you chose 'Ignore field', the fields won't be shown in the dropdowns.

dinâmico (list) <small>(see fields below)</small>	Dados dinâmico ▼
local Parque Dez de Novembro, Manaus - AM, 69055-111	geographic address ▼
id 2	text ▼

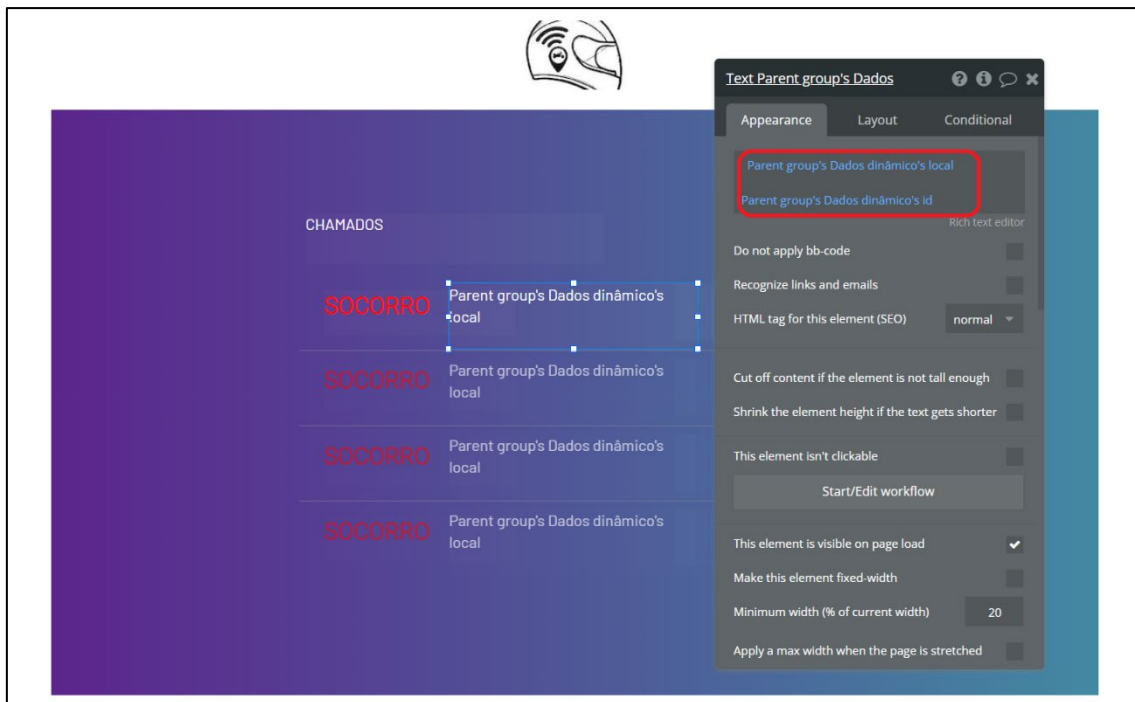
Show raw data

SAVE Cancel

Fonte: Próprio autor.

Com o sistema de comunicação desenvolvida, foi necessário depois criar uma página web formatada para a amostragem de informações do banco de dados, no caso os chamados de socorro com as localizações relacionadas específicas dos acidentes como mostrado na Figura 60, no texto de apresentação do local do chamado, foi configurado com base na comunicação externa, captar os dados do banco de dados do google, o qual foi utilizado para o projeto.

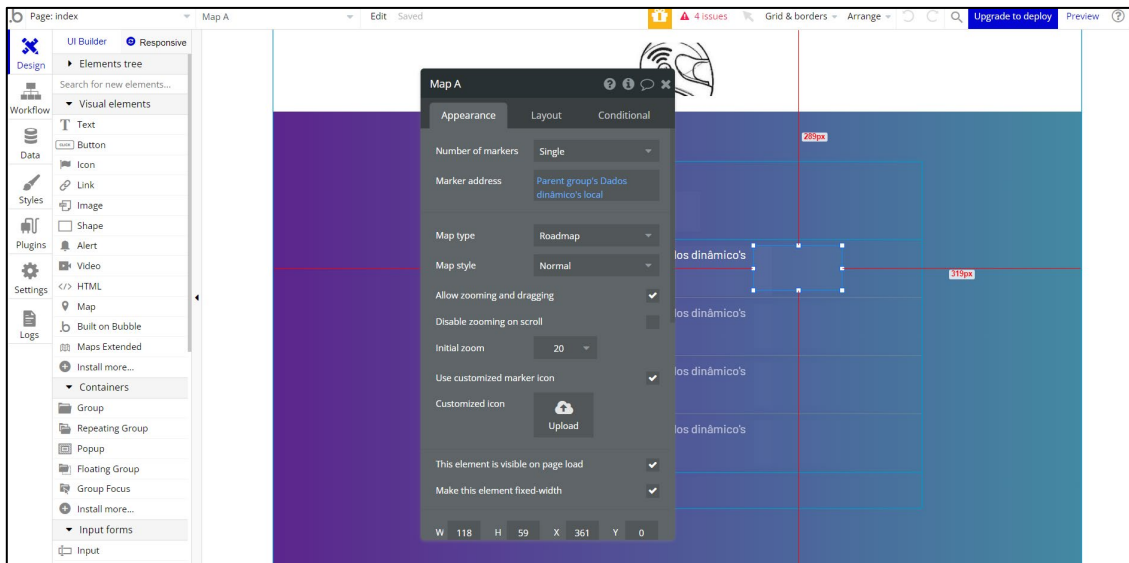
Figura 60: Texto dinâmico do acidente.



Fonte: Próprio autor.

Sendo assim, também foi desenvolvido um mapa de amostragem, onde é apresentado em tempo real o local do acidente com base nas informações recebidas do banco de dados externos, como mostrado na Figura 61.

Figura 61: Criação de mapa para amostragem do local de chamado.



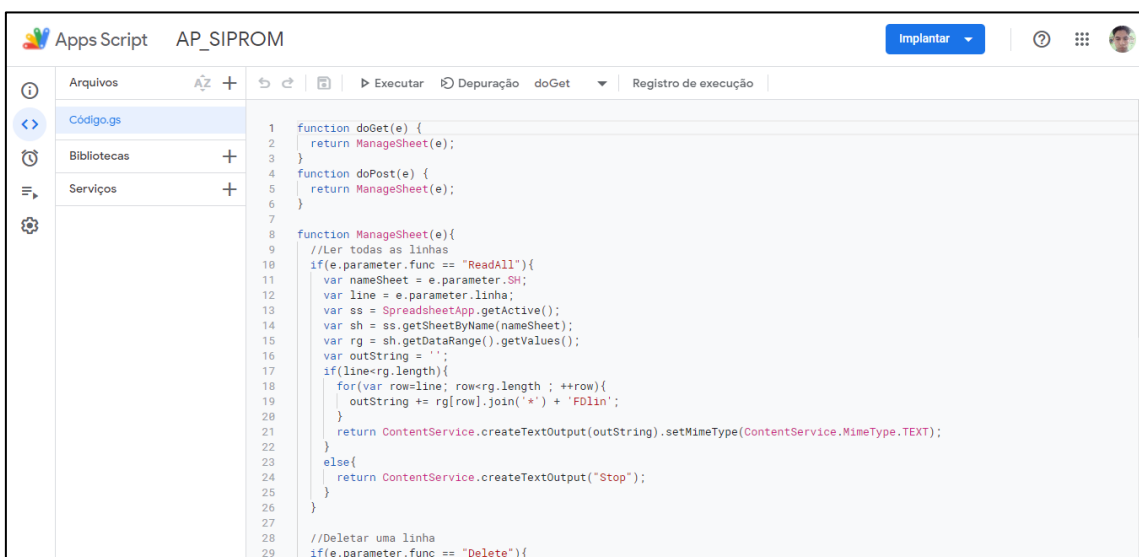
Fonte: Próprio autor.

8.5. INTEGRAÇÃO DO BANCO DE DADOS COM O APLICATIVO *MOBILE*

8.5.1. Configuração do banco de dados.

Para esta fase do projeto foi configurando dentro da planilha, onde foi necessário construir funções de manipulação de dados para adição de novas linhas no banco de dados, sendo que, foram geradas implantações de URL para métodos de adição como mostrado na Figura 62.

Figura 62: Implantação de métodos de adição de dados.



Fonte: Próprio autor.

Logo que as funções foram criadas, foi possível gerar a URL de imputação de dados para ser configurada no aplicativo *mobile*, como mostra na Figura 63.

Figura 63: URL de implantação de dados.

Ativo	Configuração
Sem título	Versão Versão 2 em 6 de mar., 15:36
Sem título	Descrição
Arquivados	Código de implantação AKfycbyDzMNEly73FYJyJwSZtjgeq3Mj0uqBhr63rqGf4ua_GFWe2injtvMcFVIQHJOCR000 Copiar
Nenhuma implantação arquivada	App da Web URL https://script.google.com/macros/s/AKfycbyDzMNEly73FYJyJwSZtjgeq3Mj0uqBhr63r... Copiar
	Executar como
	Cancelar Implantar

Fonte: Próprio autor.

8.5.2. Configuração do aplicativo *mobile* para comunicação

Assim sendo, depois que foi capturada a URL da planilha para adição de dados, foi programado no aplicativo *mobile* para o envio de dados para essa informação de link online, sendo essas configurações mostradas na Figura 64.

Figura 64: Configuração da programação do aplicativo para envio de dados.



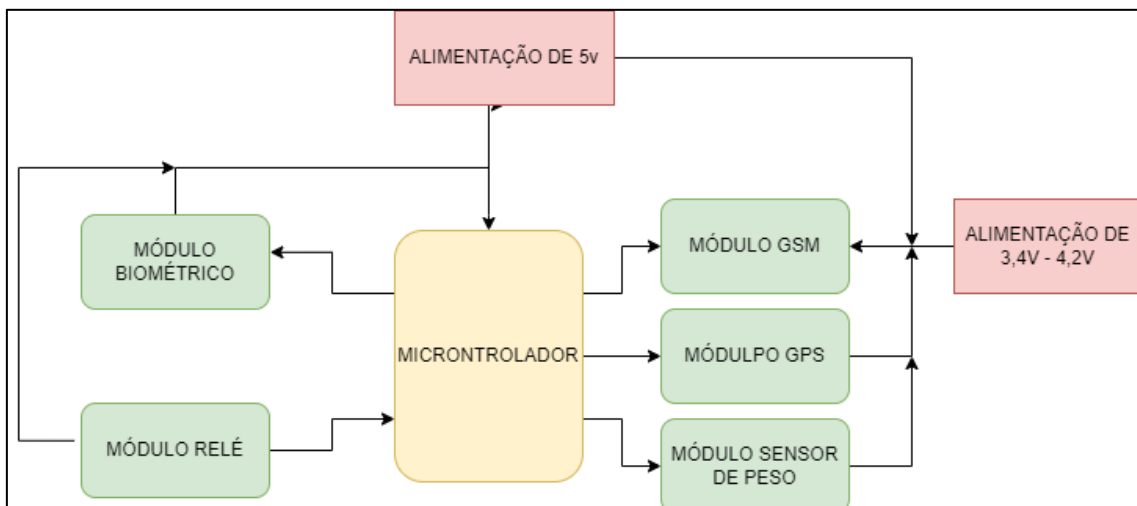
Fonte: Próprio autor.

8.6. DESENVOLVIMENTO DA PLACA DE CIRCUITO IMPRESSO.

Para planejamento dos módulos periféricos, assim como alimentações secundárias do circuito, foi feito primordialmente um esboço do diagrama de blocos, onde foi esquematizado os componentes necessários para o projeto como mostra Figura 65.

Assim sendo, temos duas alimentações, uma para o circuito principal onde é presente o microprocessador, módulo biométrico, GPS, sensor de peso e relé, enquanto temos um regulador na escala de 3,4Volts a 4,2Volts para o módulo gsm, sendo que todos compartilham o mesmo terra.

Figura 65: Diagrama de blocos da arquitetura do circuito.

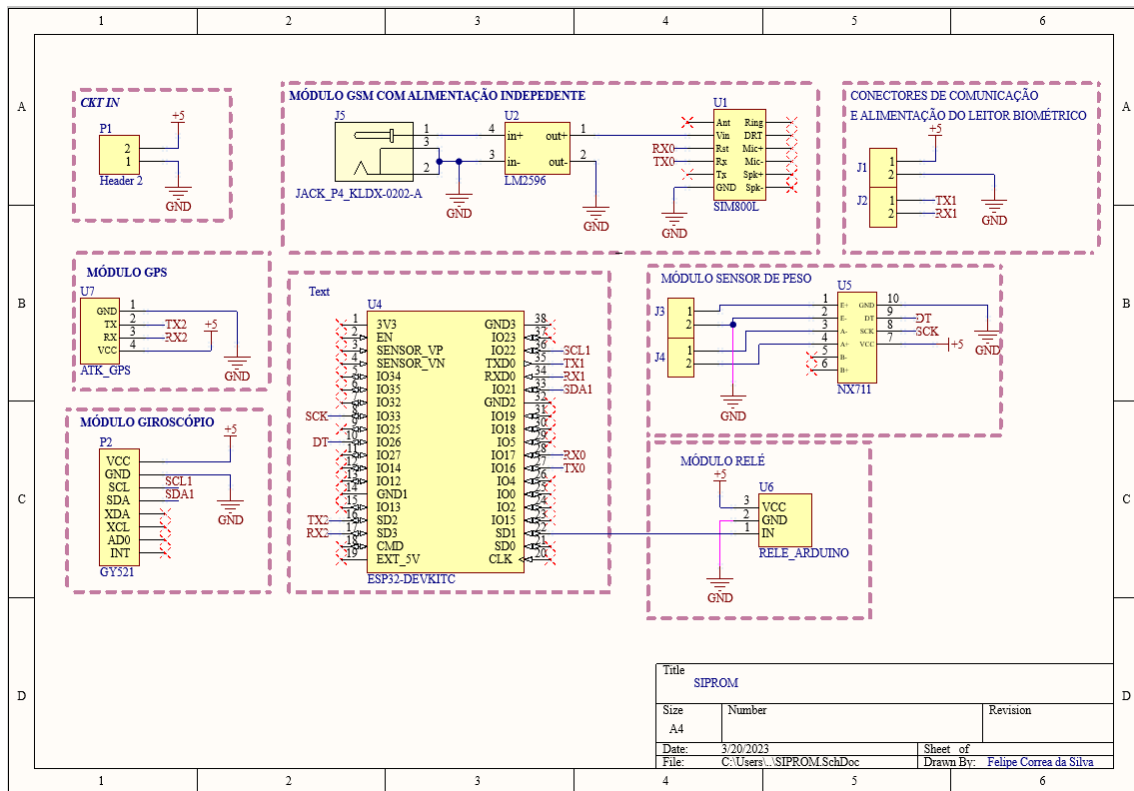


Fonte: Próprio autor.

8.6.1. Desenvolvimento da placa no altium designer

Para desenvolvimento de uma placa de circuito impresso contendo as trilhas e furos referente ao esquema elétrico, foi construído primeiramente no Altium designer as conexões dos módulos e do microprocessador do sistema desenvolvido, em que de forma organizacional foi dividido em blocos para cada componente, com as devidas identificações e separações, como mostra Figura 66.

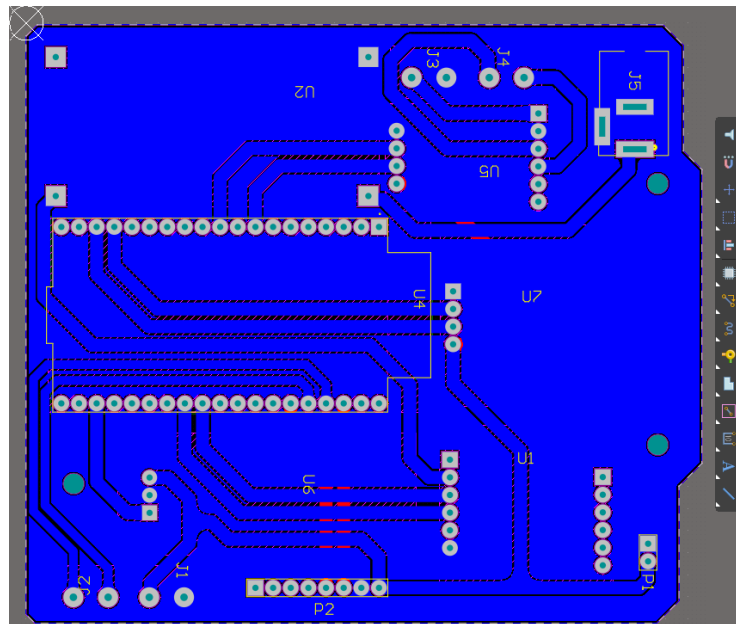
Figura 66: Esquema elétrico do projeto.



Fonte: Autor próprio.

Sendo assim, logo depois, foi desenvolvido as trilhas de conexões elétricas, bem como os furos relacionados aos pinos dos componentes e o contorno de corte da placa, como mostrado na Figura 67.

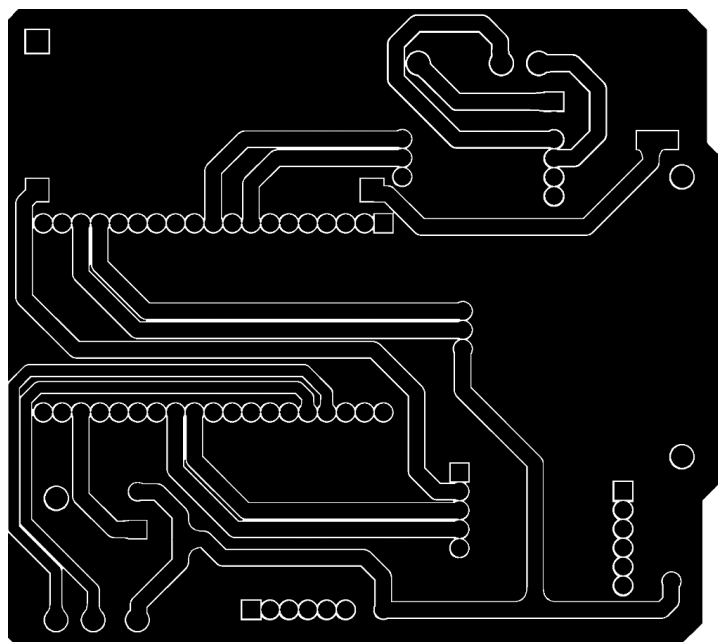
Figura 67: Designe da placa de circuito impresso contendo as trilhas e furos.



Fonte: Autor próprio.

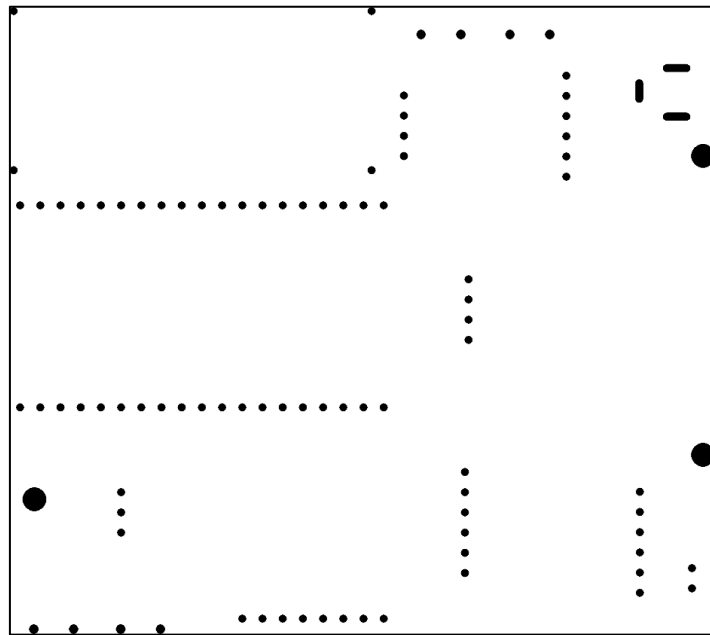
Logo depois desse processo, foi necessário exportar os componentes de manufatura em arquivos bitmap para geração de arquivos gcode para a máquina CNC, sendo esse processo mostrado nas Figura 68 e Figura 69.

Figura 68: Imagem da placa em formato Bitmap.



Fonte: Autor próprio.

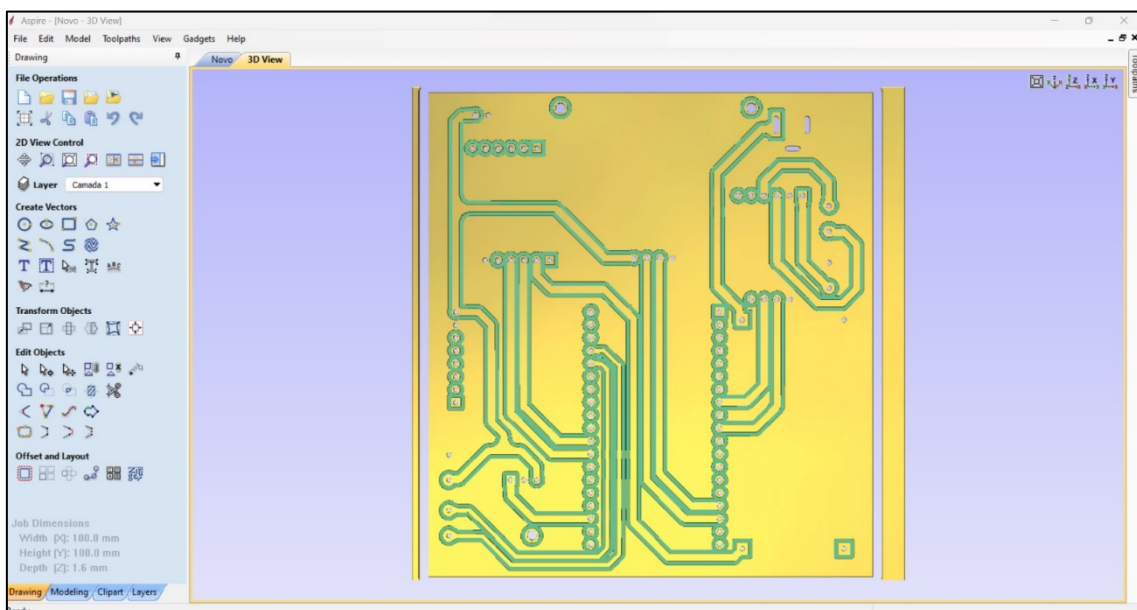
Figura 69: Imagem dos pontos de furos em formato bitmap.



Fonte: Autor próprio.

Assim sendo, foi necessário configurar o arquivo para tipo gcode no programa Aspire, onde conteve informações de passo a passo da máquina, bem como dados dos tipos de brocas utilizadas para as extensões da placa, em se tratando das trilhas e furos, sendo esse processo mostrado da Figura 70.

Figura 70: Configuração do arquivo gcode no Aspire.



Fonte: Autor próprio.

8.6.2. Manufatura da placa física

Em seguida, com todas as configurações em *software* realizadas e os arquivos gcode gerados, foi imputado esses dados na máquina de manufatura de placas, e logo em seguida foi protótipada, sendo que, inicialmente é necessária uma placa de fenolite cobreada de dimensões padrão para o desenvolvimento do circuito, como mostrado na Figura 71.

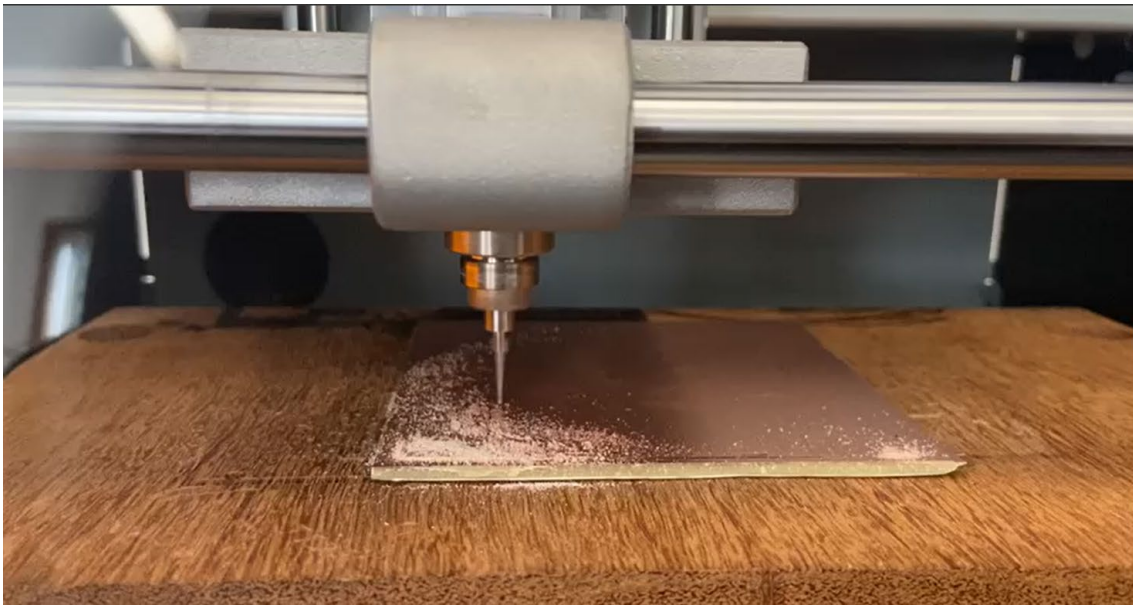
Figura 71: Placa de fenolite padrão cobreada.



Fonte: Autor próprio.

Assim sendo, essa placa de fonolite cobreada é inserida e fixada na máquina CNC para início do processo de manufatura do sistema físico desenvolvido no Altium, onde será retirada caminhos do metal com a broca utilizada de acordo com os esquemas de trilhas criado, como mostra na Figura 72.

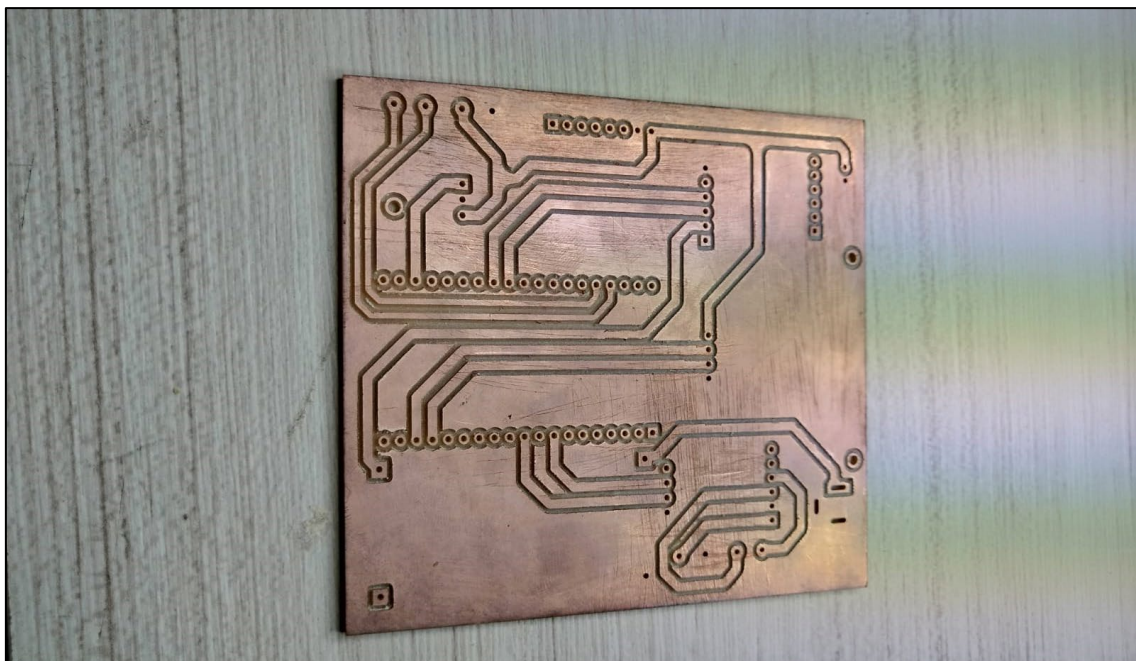
Figura 72: Manufatura da placa de circuito impresso na máquina de CNC.



Fonte: Autor Próprio.

Assim que a máquina CNC concluiu com o processo de manufatura das trilhas, furos e contorno da placa, foi analisado o resultado como mostra na Figura 73.

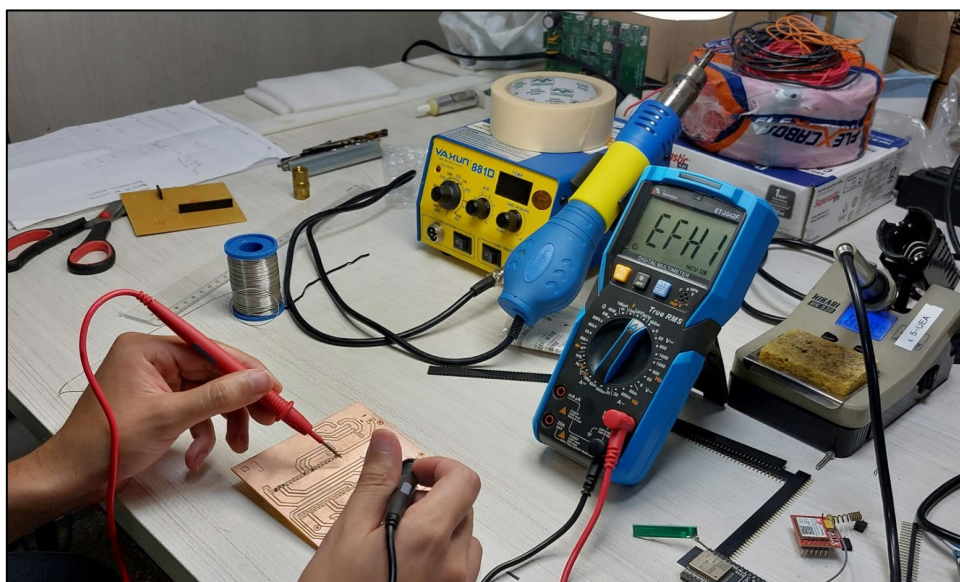
Figura 73: Placa de fenolite depois do processo de manufatura da placa CNC.



Fonte: Autor próprio.

Para termos de análise funcionalidades da placa, em relação a ausência de curtos e mal contatos, foi testado com multímetro na função de condutividade, as trilhas do sistema, como mostra a Figura 74.

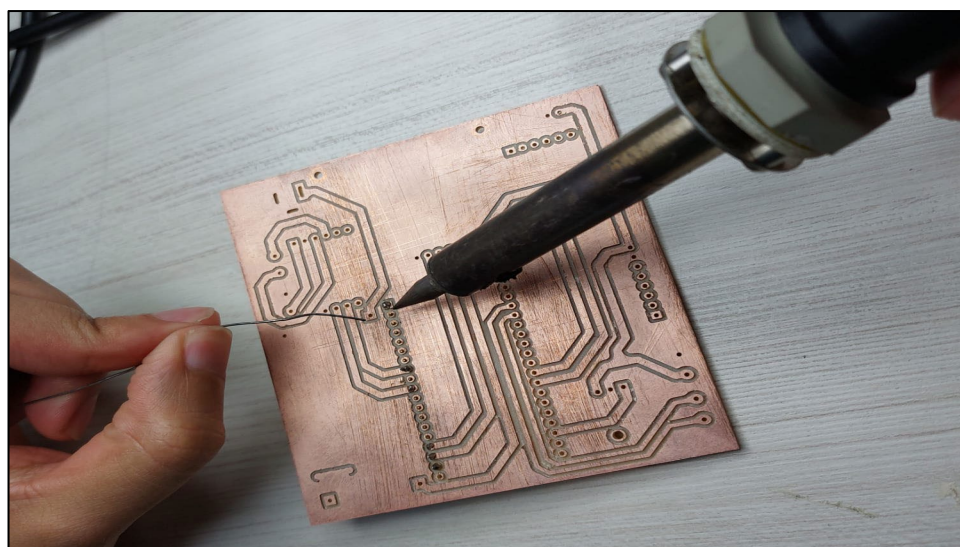
Figura 74: Teste de condutividade das trilhas da placa.



Fonte: Próprio autor.

Para prosseguimento do projeto, foi soldado *headers* de conexão, pois como se trata de módulos eletrônicos que podem queimar, de forma prática para a substituição, foi a solução encontrada. Sendo esse processo mostrado na Figura 75.

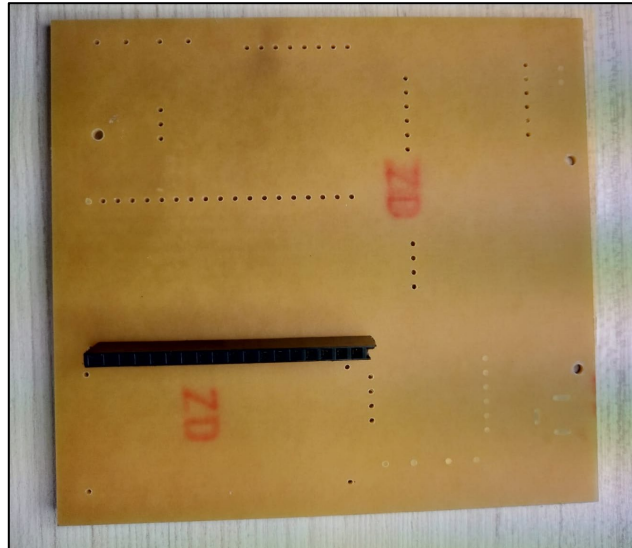
Figura 75: Soldagem dos headers na placa manufacturada.



Fonte: Próprio autor.

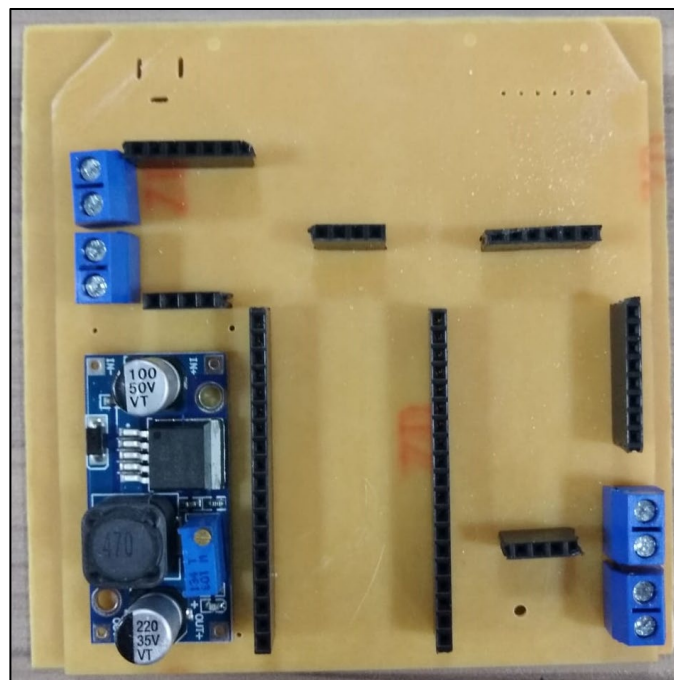
Com o processo de solda finalizado, foi analisado o resultado da fixação dos headers e dos conectores, vendo uma Figura 76 do anterior, e uma Figura 77 do posterior.

Figura 76: Placa manufaturada sem conectores.



Fonte: Próprio autor.

Figura 77: Placa manufaturada com conectores.

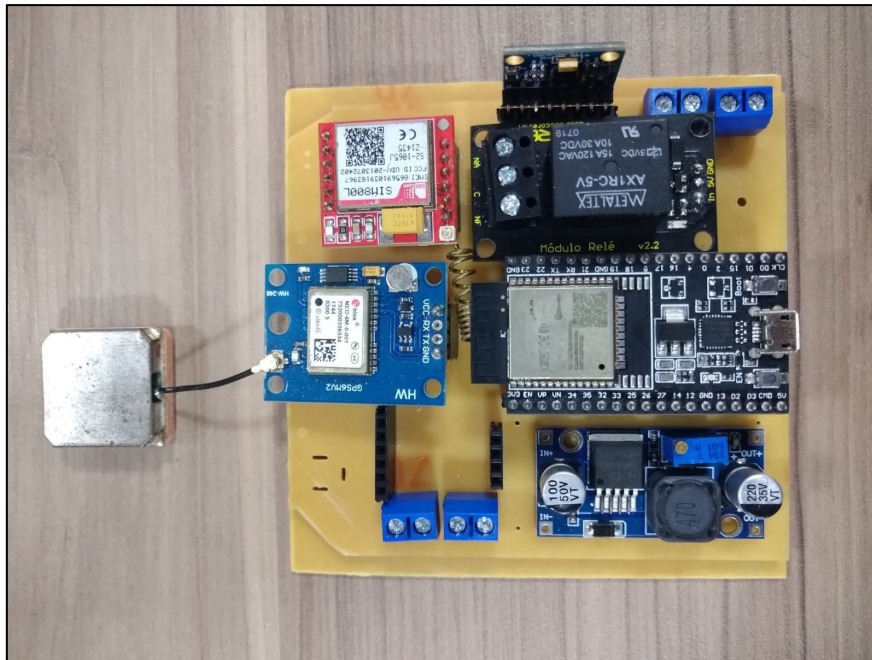


Fonte: Próprio autor.

8.6.3. Acoplagem de componentes na de circuito impresso

Sendo assim, para finalização do desenvolvimento da placa de circuito impresso, foi acoplado todos os componentes e módulos, bem como sensores nos conectores do sistema, como mostrado na Figura 78.

Figura 78: Placa de circuito impresso finalizada.



Fonte: Próprio autor.

8.7. DESENVOLVIMENTO DE PEÇAS D3

Para a implementação real do projeto em uma motocicleta, foi necessário o desenvolvimento de peças de suporte para o equipamento que seria instalado, bem como *cases* do circuito protótipado manufaturado para o sistema, além de periféricos de módulos existentes do equipamento.

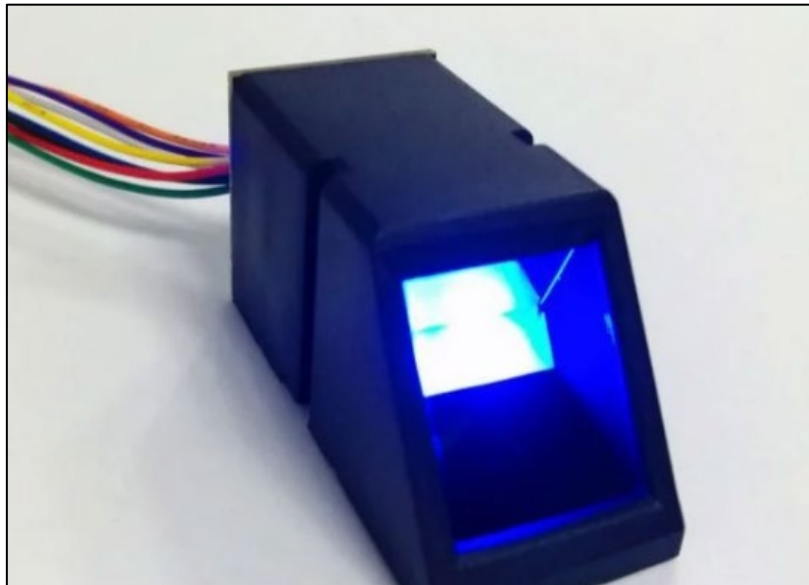
Assim sendo, foi utilizado o programa fusion 360 que é capaz de projetar desenhos 3d para as mais variadas aplicações, inclusive a engenharia, em relação ao auxílio de peças que irão suportar placas de circuitos impressos e sensores utilizados.

8.7.1. Suporte para módulo biométrico

Como o sistema utiliza um módulo biométrico R307 como mostrado na Figura 79, foi desenvolvido um módulo para suporte para esse equipamento que estivesse mesclado com um prendedor adequado a um guidão de motocicleta, como verificado e medido na moto utilizado para o projeto, viu-se que o diâmetro da circunferência era de

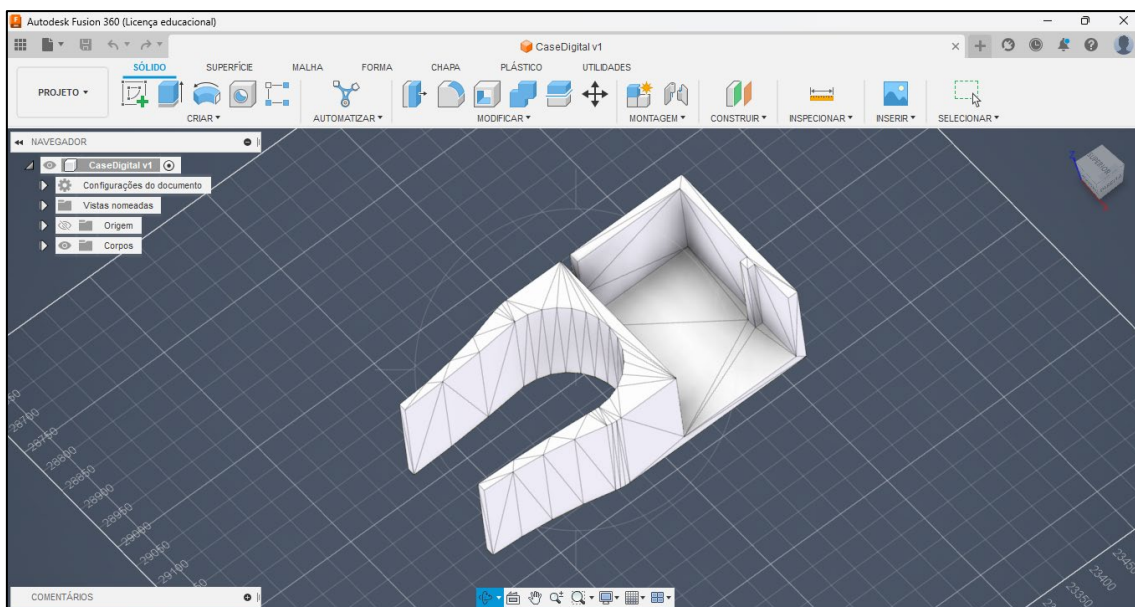
2 cm, logo, foi construído uma peça 3d que atendesse esses requisitos, como mostrado na Figura 80.

Figura 79: Módulo biométrico R307.



Fonte: Próprio autor.

Figura 80: Modelagem do suporte para o módulo leitor biométrico.



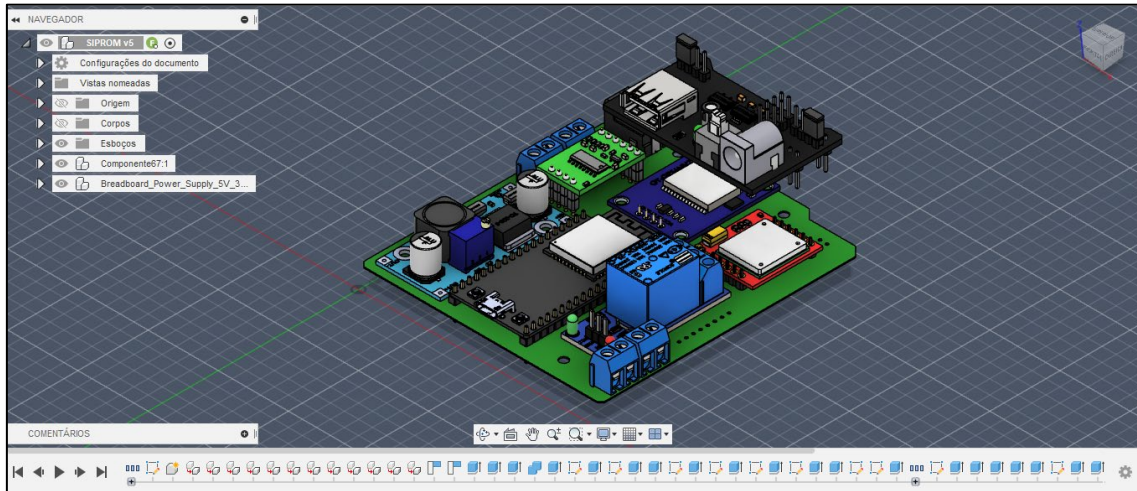
Fonte: Próprio autor.

8.7.2. *Case da placa de circuito impresso do projeto*

Para a necessidade de um armazenamento seguro do sistema em se tratando da placa de circuito impresso, foi analisado e projetado um *case* com base na placa gerada

por programa de computador mostrada anteriormente, de forma que foram respeitadas as limitações de dimensões como mostra a Figura 81.

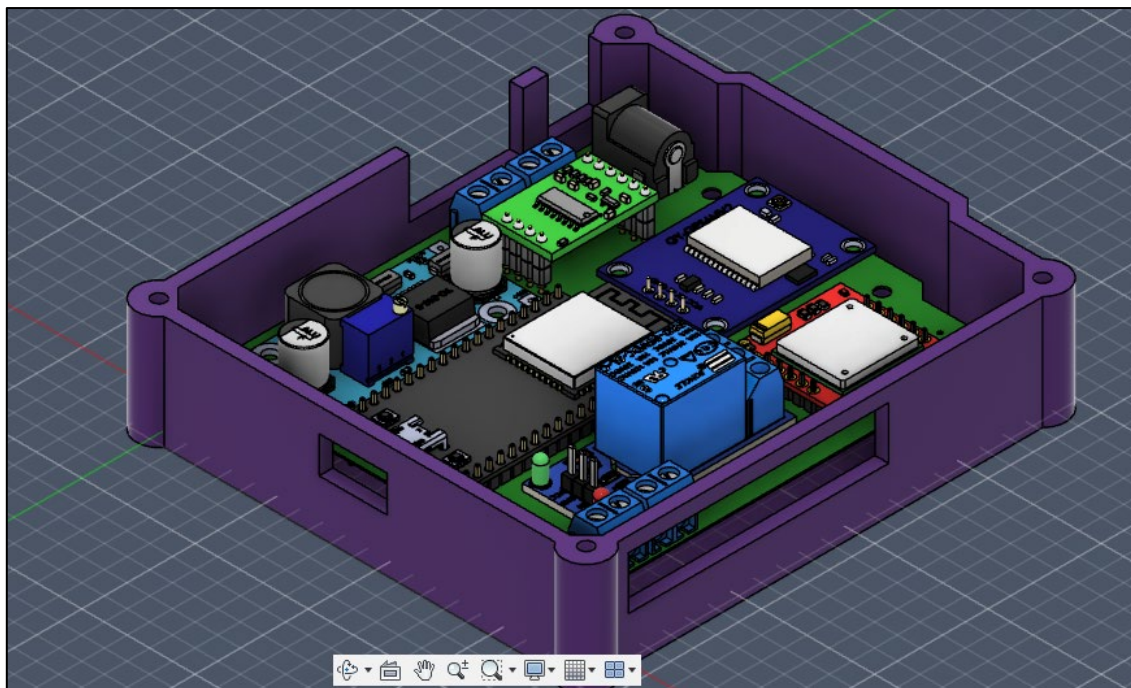
Figura 81: Placa de circuito impresso no Fusion.



Fonte: Próprio autor.

Primeiramente foi construído o periférico da lateral do *case* com furos e rasgos relacionados a saídas e entradas de cabos como mostrado na, junto com circunferências nas extremidades superiores para parafusos, onde será encaixado uma tampa.

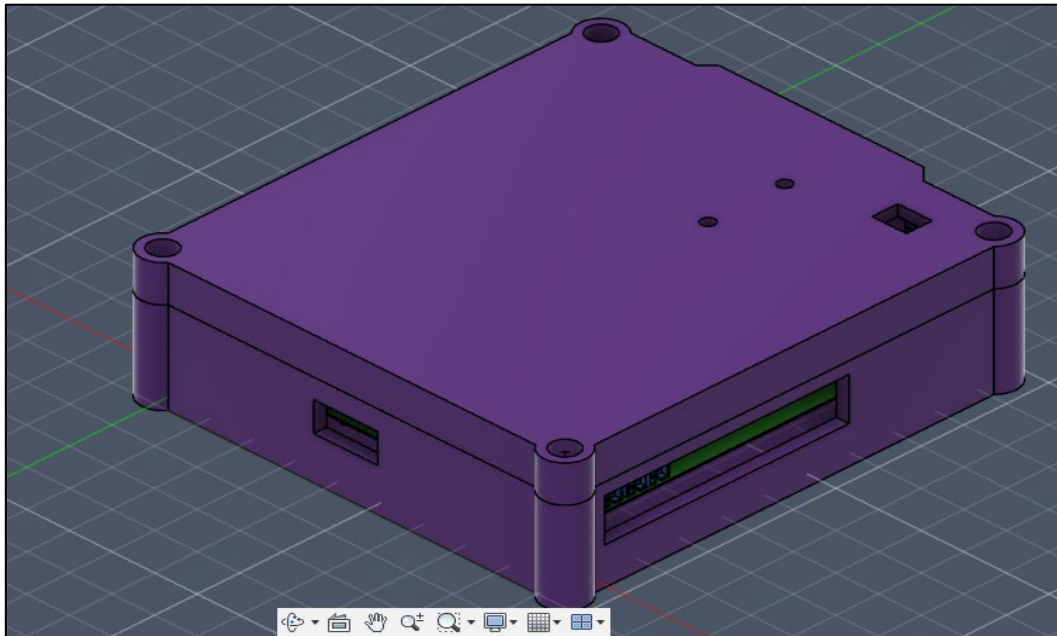
Figura 82: *Case* lateral da placa de circuito impresso.



Fonte: Próprio autor.

Assim, foi encapsulada com uma tampa superior com rasgos para integração com outro *case* de suporte para a fonte regulável do sistema, como mostrado na Figura 83.

Figura 83: *Case* fechado com tampa.

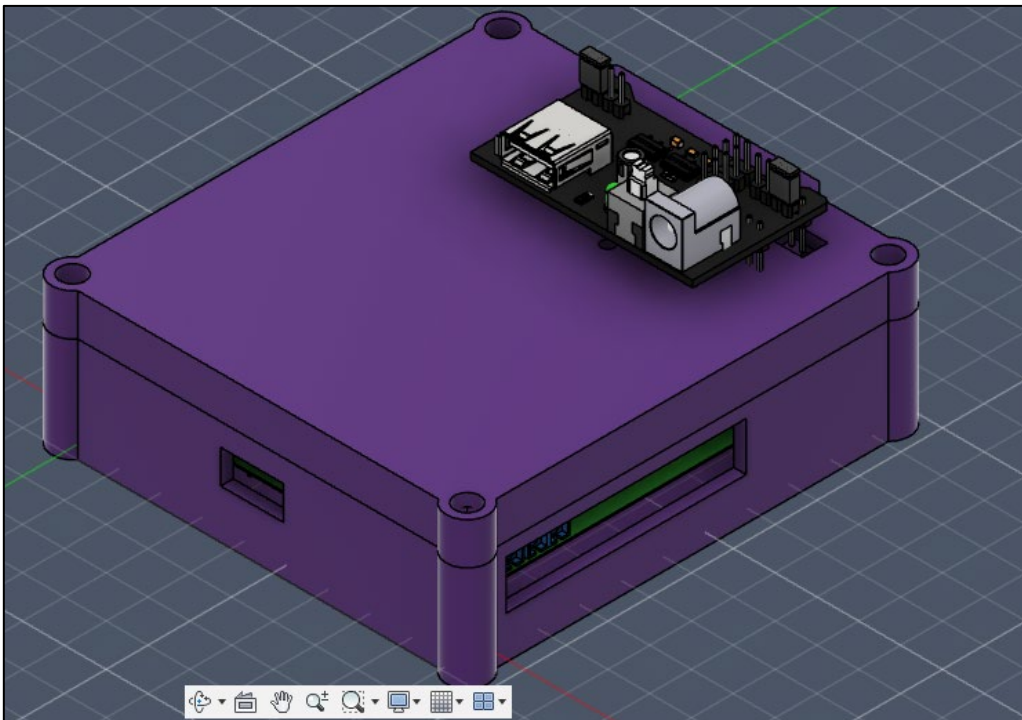


Fonte: Próprio autor.

8.7.3. *Case* da placa de fonte regulável

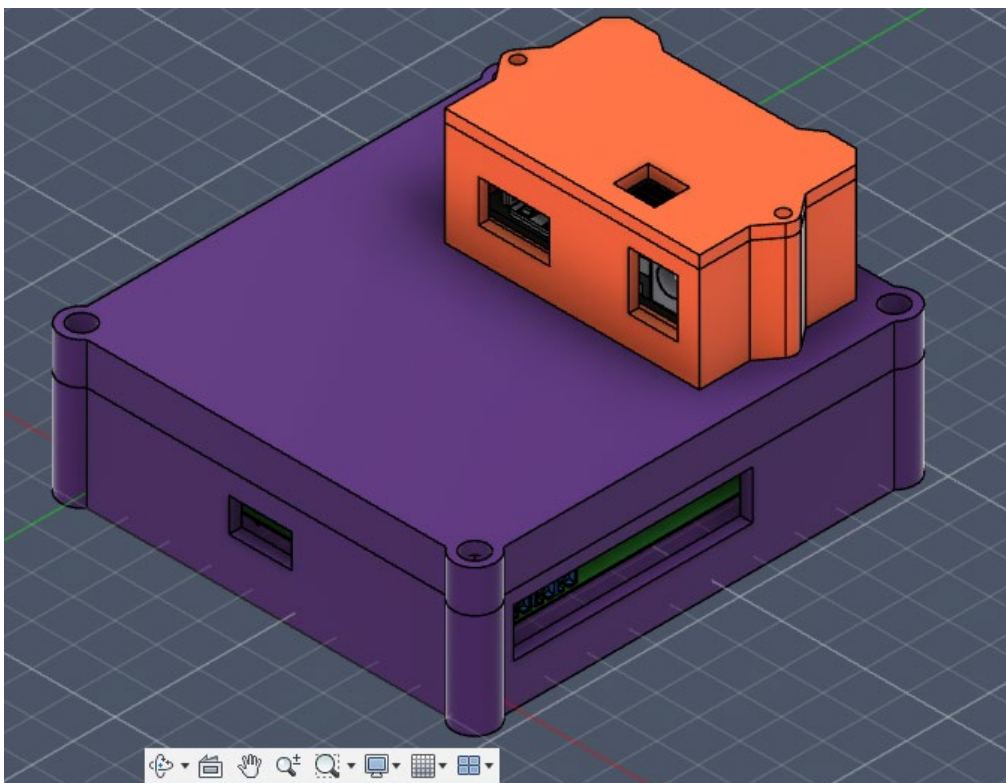
Logo após isso, foi desenvolvido uma segundo *case* responsável por servir como suporte para a fonte regulável de tensão que alimentará a placa de circuito impresso como mostrado na Figura 84, onde está possuirá uma fixação por parafuso no *case* maior para integração dos dispositivos, como mostra a Figura 85.

Figura 84: *Case* principal com placa de alimentação na parte superior.



Fonte: Próprio autor.

Figura 85: Fixação dos dois *cases*.

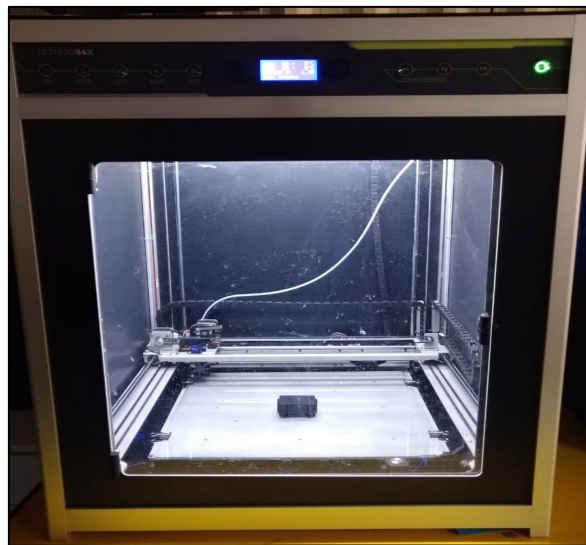


Fonte: Próprio autor.

8.7.4. Impressão das peças 3d

Para finalização da manufatura das peças 3D, foram imprimidas todas as modelagens construídas no fusion 360 em uma impressora 3D como mostra a Figura 86, em se tratando dos dois *cases* de placas de circuito impresso e do suporte protetor do módulo de leitura biométrico, como mostrado nas Figura 87 e Figura 88.

Figura 86: Manufatura das peças modelas na impressora 3D.



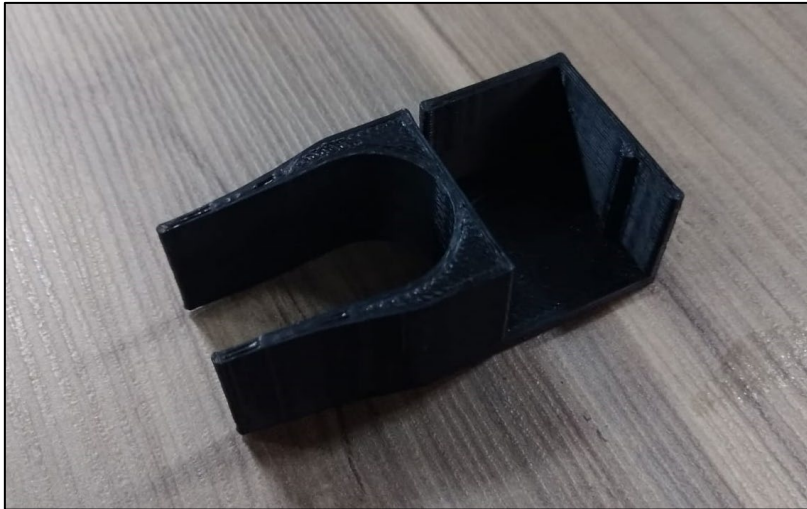
Fonte: Próprio autor.

Figura 87: *Cases* da placa principal de fonte regulável de tensão.



Fonte: Próprio autor.

Figura 88: Suporte protetor do leitor biométrico.



Fonte: Próprio autor.

8.7.5. Acoplamento dos módulos nos *cases* 3d

Assim que todos as peças 3D foram manufaturadas, foi acoplada as duas placas do sistema nos *cases*, bem como o módulo sensor biométrico no suporte como mostra na Figura 89, foi verificado também que foi necessário fazer cortes diagonais nas bordas da placa para o cabimento da mesma na caixa protetora como mostra Figura 90.

Figura 89: Acoplamento de peças nos módulos.

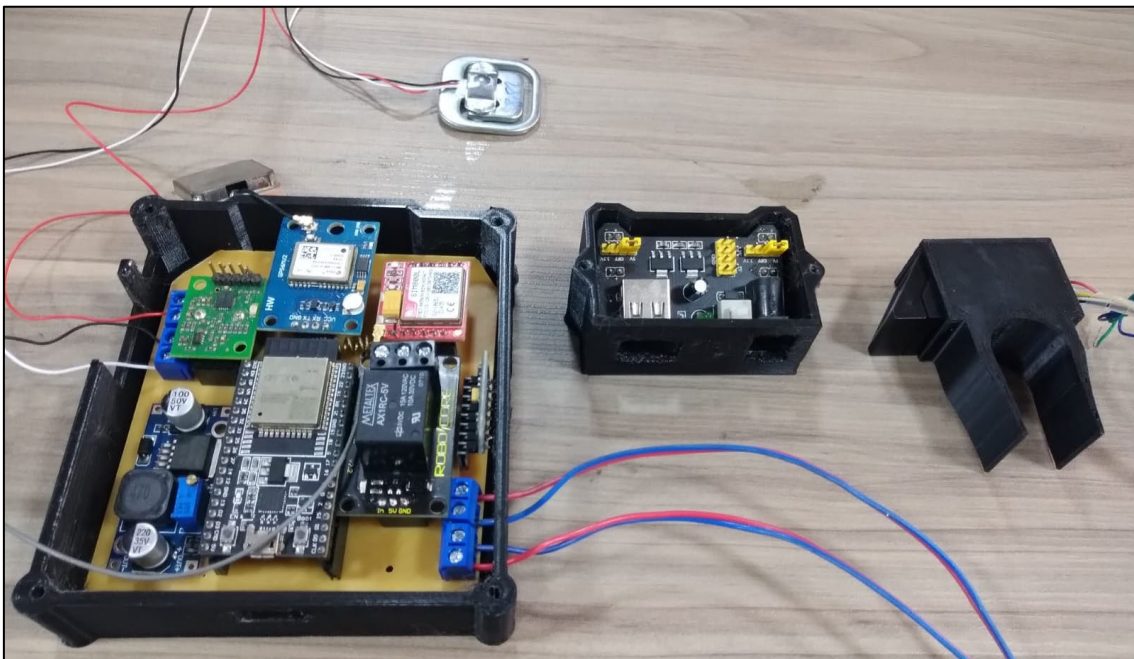
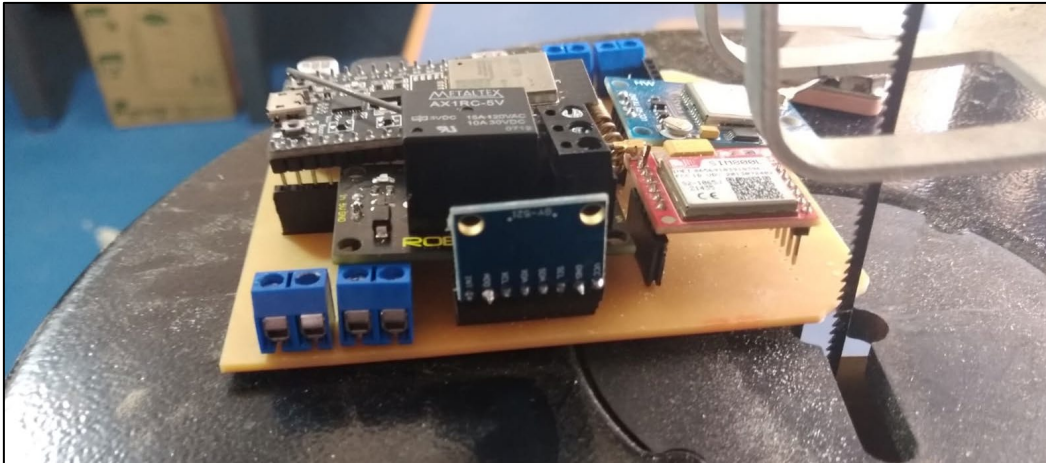


Figura 90: Corte das bordas da placa manufacturada.



Fonte: Autor próprio.

8.8. DESENVOLVIMENTO DE FIRMWARE

Primeiramente, foi desenvolvido o *firmware* de conexão do esp32 com a internet, onde o microcontrolador poderá se comunicar com o aplicativo *mobile*, para envio e recepção de dados, os quais são responsáveis pelas funcionalidades do sistema, como mostra a Figura 91.

Figura 91: Programação da conexão do esp32 com a internet.

```
const char* ssid = "ASUS_A001F";
const char* password = "terminal7";

WiFiServer server(80);
int i=0;
void setup(){
  Serial.begin(115200);
  pinMode(4, OUTPUT); // set the LED pin mode
  pinMode(2, OUTPUT);
  digitalWrite(4, HIGH);
  digitalWrite(2, LOW);
  delay(10);
  // We start by connecting to a WiFi network
  Serial.print("\nConnecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  boolean tr=true;
  while (tr){
    i++;
    if (WiFi.status() != WL_CONNECTED) {
      digitalWrite(2, HIGH);
      delay(500);
      digitalWrite(2, LOW);
      delay(500);
      Serial.print(".");
    }
  }
}
```

Fonte: Próprio autor.

Foi feito o código de validação de conectividade do módulo sensor biométrico com o microcontrolador esp32, aonde ele retorna mensagens via serial UART sobre a conexão como mostrado na Figura 92.

Figura 92: Função de conexão do esp32 com o sensor de biometria.

```
void setupFingerprintSensor(){
  //Inicializa o sensor
  fingerprintSensor.begin(57600);
  //Verifica se a senha está correta
  if(!fingerprintSensor.verifyPassword()){
    //Se chegou aqui significa que a senha está errada ou o sensor está problemas de conexão
    Serial.println(F("Não foi possível conectar ao sensor. Verifique a senha ou a conexão"));
    while(true);
  }
}
```

Fonte: Próprio autor.

Assim sendo, logo depois foi feito o código de comandos do módulo biométrico, em se tratando do cadastro, verificação e apagamento, isso sendo feito identificação de caracteres transmitidos como mostra a Figura 93.

Figura 93: Comandos de funcionalidades do módulo biométrico.

```
if (currentLine.endsWith("GET /C")) {
  storeFingerprint();
  client.println("HTTP/1.1 200 OK");
  client.println("Content-type:text/html");
  client.println();
  client.print("Cadastrada");
  client.println();
}
if (currentLine.endsWith("GET /V")) {
  checkFingerprint();
  if (aux2 == 1){
    client.println("HTTP/1.1 200 OK");
    client.println("Content-type:text/html");
    client.println();
    client.print("Verificado com sucesso!");
    client.println();
  }
  else{
    client.println("HTTP/1.1 200 OK");
    client.println("Content-type:text/html");
    client.println();
    client.print("Acesso negado!");
    client.println();
  }
}
```

Fonte: Próprio autor.

Para uma validação das execuções de rotinas contidas no esp32, foi programado impressões de texto na serial do microcontrolador, como mostra Figura 94.

Figura 94: Impressão da execução de funções do esp32.

```
void printMenu()
{
  Serial.println();
  Serial.println(F("Digite um dos números do menu abaixo"));
  Serial.println(F("1 - Cadastrar digital"));
  Serial.println(F("2 - Verificar digital"));
  Serial.println(F("3 - Mostrar quantidade de digitais cadastradas"));
  Serial.println(F("4 - Apagar digital em uma posição"));
  Serial.println(F("5 - Apagar banco de digitais"));
}
```

Fonte: Autor próprio.

Nesse bloco da programação, foi codificado para quando um usuário estiver interagindo com o leitor biométrico, o microcontrolador procurar por dados já existentes, quando essa informação for verdadeira, o sistema ativa o contato do módulo relé, o qual aciona a motocicleta, e quando a verificação de busca não for encontrada, a chave eletromecânica não ativa e o veículo continua desligado, sendo esse código mostrado na Figura 95.

O código de validação é controlado por variável dinâmica que muda de estado conforme é constado ou não que o usuário está no banco de dados, conforme o valor dessa variável, é imputado nível alto ou baixo no módulo relé, ou seja, ativa ou desativa.

Figura 95: *Firmware* de validação de usuários.

```
//Verifica se a digital está cadastrada
void checkFingerprint()
{
  Serial.println(F("\nAguardando sinal do proprietário...\n"));

  //Espera até pegar uma imagem válida da digital
  while (fingerprintSensor.getImage() != FINGERPRINT_OK);

  //Converte a imagem para o padrão que será utilizado para verificar com o banco de digitais
  if (fingerprintSensor.image2Tz() != FINGERPRINT_OK)
  {
    //Se chegou aqui deu erro, então abortamos os próximos passos
    Serial.println(F("Erro image2Tz"));
    return;
  }

  //Procura por este padrão no banco de digitais
  if (fingerprintSensor.fingerFastSearch() != FINGERPRINT_OK)
  {
    //Se chegou aqui significa que a digital não foi encontrada
    Serial.println(F("\nAcesso negado!\n"));
    aux2 = 0;
    return;
  }

  //Se chegou aqui a digital foi encontrada
  //Mostramos a posição onde a digital estava salva e a confiança
  //Quanto mais alta a confiança melhor
  // Serial.print(F("Digital encontrada com confiança de "));
  // Serial.print(fingerprintSensor.confidence);
  // Serial.print(F(" na posição "));
  // Serial.println(fingerprintSensor.fingerID);
  Serial.print(F("\nVerificado com sucesso!\n"));
  aux2 = 1;
  digitalWrite(4, LOW);
  delay(1500);
  digitalWrite(4, HIGH);
}
```

Fonte: Próprio autor.

Foi criado um código que faz a leitura do módulo giroscópio analisando o ângulo da inclinação do sensor em relação a superfície, e ao mesmo tempo essas informações são enviadas para o aplicativo via internet, como mostrado na Figura 96.

Figura 96: Código de leitura e envio dos dados do sensor giroscópio.

```
void loop(){
  Vector rawGyro = mpu.readGyroRaw();
  float gyroX = rawGyro.x;
  float gyroY = rawGyro.y;
  float gyroZ = rawGyro.z;

  Serial.print("Gyro X: ");
  Serial.print(gyroX);
  Serial.print(" Y: ");
  Serial.print(gyroY);
  Serial.print(" Z: ");
  Serial.println(gyroZ);

  sendData(gyroX, gyroY, gyroZ);
  delay(5000);
}

void sendData(float gyroX, float gyroY, float gyroZ) {
  if (WiFi.status() == WL_CONNECTED) {
```

Fonte: Próprio autor.

Foi desenvolvido um código que faça leituras de peso de uma célula de carga para validação de um usuário sentado na moto, e envie essa informação via http para o aplicativo, como mostra Figura 97.

Contudo, é necessário que seja calibrada as células de carga com pesos padrões para uma leitura mais precisa, como mostra a Figura 98.

Figura 97: Envio das informações de leitura de peso.

```
float readCells() {
  int cellReading1 = analogRead(CELL_PIN_1);
  int cellReading2 = analogRead(CELL_PIN_2);

  float peso1 = map(cellReading1, 0, 4095, 0, 3300) / 1000.0;
  float peso2 = map(cellReading2, 0, 4095, 0, 3300) / 1000.0;

  float force1 = peso1 / 0.5; // substitua o valor 0.5 pelo valor de calibração da célula
  float force2 = peso2 / 0.5; // substitua o valor 0.5 pelo valor de calibração da célula

  return force1 + force2; // retorna a força total em unidades de medida
}

void sendData(float force) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    http.begin(serverName);

    http.addHeader("Content-Type", "application/json");

    String requestBody = "{\"force\": " + String(force) + "}";

    int httpResponseCode = http.POST(requestBody);
  }
}
```

Fonte: Próprio autor.

Figura 98: Calibração das células de carga.

```
balanca.set_scale(calibration_factor); // a balança está em função do fator de calibração */

Serial.print("Peso: "); // Printa "Peso:" na COM */
peso = balanca.get_units(), 10; // imprime peso */
if (peso < 0) // se a unidade for menor que 0 será considerado 0 */
{
  peso = 0.00; // Para o caso do peso ser negativo, o valor apresentado será 0 */
}
Serial.print(peso); // Printa o peso na serial */
Serial.print(" kg"); // Printa "kg" na serial */
Serial.print(" Fator de calibração: "); // Printa "Fator de calibração:" na serial */
Serial.print(calibration_factor); // Printa o fator de calibração na serial */
Serial.println(); // Pula linha no serial */
delay(500); // atraso de 500ms = 0,5s*/

if(Serial.available()) // caso sejam inseridos caracteres no serial */
{
  char temp = Serial.read();
  if(temp == '+') // Se o + for pressionado */
  calibration_factor += 1; // incrementa 1 no fator de calibração */
  else if(temp == '-') // Caso o - seja pressionado */
  calibration_factor -= 1; // Decrementa 1 do fator de calibração */
}
}
```

Fonte: Próprio autor.

Sendo assim, também foi desenvolvido um *firmware* que faz leituras do módulo gps e envie essas informações para o aplicativo *mobile*, quando o sistema detectar um acidente, como mostra Figura 99.

Figura 99: *Firmware* do módulo GPS.

```
if (gps.location.isValid()) {
  float latitude = gps.location.lat();
  float longitude = gps.location.lng();

  Serial.print("Latitude: ");
  Serial.println(latitude, 6);
  Serial.print("Longitude: ");
  Serial.println(longitude, 6);

  sendData(latitude, longitude);
}

while (GPS_SERIAL.available() > 0) {
  gps.encode(GPS_SERIAL.read());
}

void sendData(float latitude, float longitude) {
  if (WiFi.status() == WL_CONNECTED) {
    HTTPClient http;

    http.begin(serverName);

    http.addHeader("Content-Type", "application/json");

    String requestBody = "{\"latitude\": " + String(latitude, 6) + ", \"longitude\": " + String(longitude, 6) + "}";

    int httpResponseCode = http.POST(requestBody);

    if (httpResponseCode > 0) {
      Serial.print("Resposta do servidor: ");
      Serial.println(httpResponseCode);
    } else {
      Serial.println("Erro na requisicao");
    }

    http.end();
  } else {
    Serial.println("Erro na conexao WiFi");
  }
}
```

Fonte: Próprio autor.

8.9. INSTALAÇÃO DO SISTEMA EM UMA MOTO REAL

8.9.1. Acoplamento dos *cases* e suporte nos periféricos da moto

Foi acoplado os *cases* integrados com as placas de circuito impresso e o módulo biométrico em moto real para os testes de funcionalidade do sistema como mostra Figura 100.

Figura 100: Acoplamento dos sistemas em moto real.

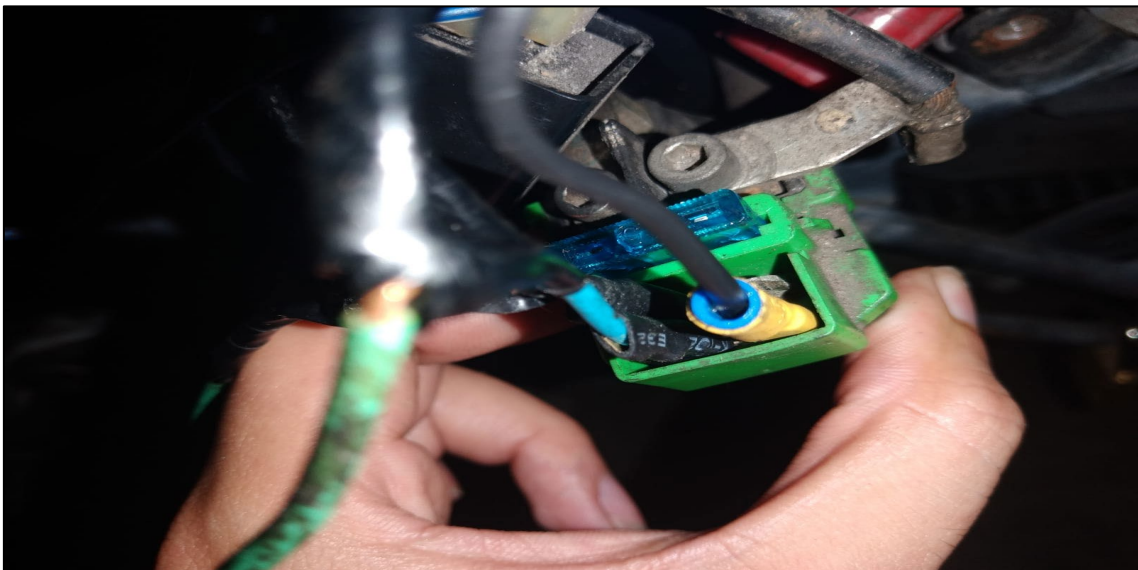


Fonte: Próprio autor.

8.9.2. Integração do chaveamento automático no relé de ignição

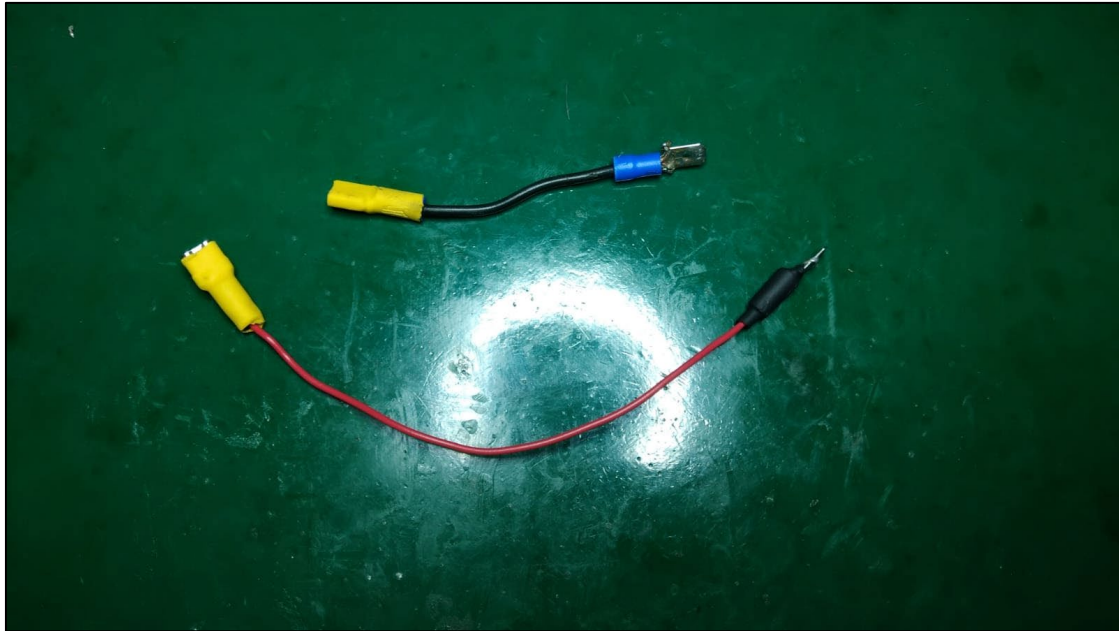
Foi integrado no sistema de relé de ignição o sistema de chaveamento inteligente nos pinos de inicialização da parte elétrica da moto, como mostra na Figura 101, para isso, foi necessário manufaturar cabos de conexão, como mostrado na Figura 102

Figura 101: instalação do sistema de chaveamento no relé de ignição.



Fonte: Próprio autor.

Figura 102: Cabos de conexão manufacturados.



Fonte: Próprio autor.

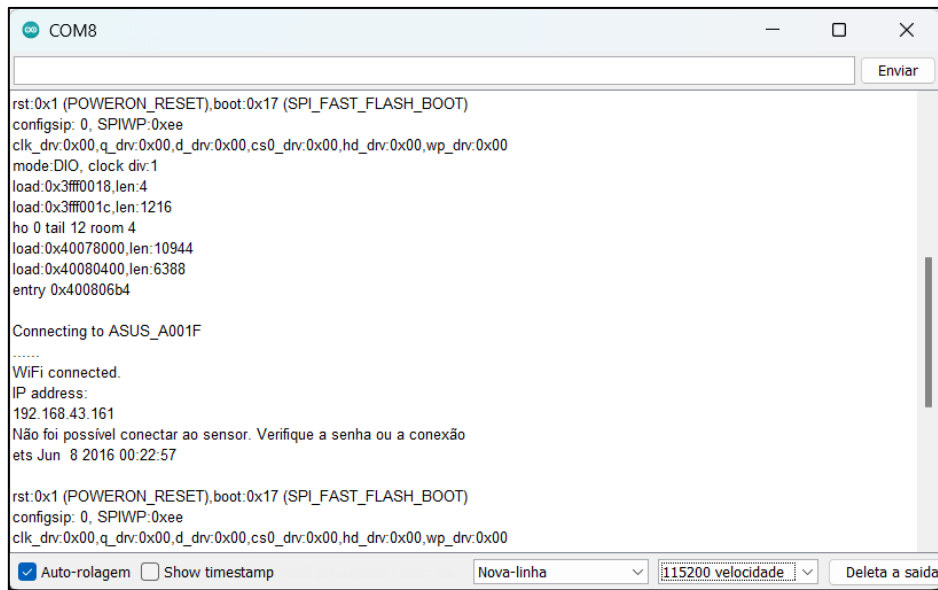
9. ANÁLISE DE RESULTADOS

9.1. TESTE DO SISTEMA DE ANTIFURTO E ROUBO DE MOTOCICLETAS.

Foi verificado as validações pelos protocolos de comunicação serial para os testes de funcionalidades, onde é impresso as execuções de rotinas pela UART do ESP32.

Primeiramente foi testado a conexão do microcontrolador com a internet do celular, onde foi validado pelo IP da rede, como mostrado Figura 103.

Figura 103: Teste de conexão do microcontrolador com a internet.



```
COM8
Enviar

rst:0x1 (POWERON_RESET),boot:0x17 (SPI_FAST_FLASH_BOOT)
config: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
mode:DIO, clock div:1
load:0x3fff0018,len:4
load:0x3fff001c,len:1216
ho 0 tail 12 room 4
load:0x40078000,len:10944
load:0x40080400,len:6388
entry 0x400806b4

Connecting to ASUS_A001F
.....
WiFi connected.
IP address:
192.168.43.161
Não foi possível conectar ao sensor. Verifique a senha ou a conexão
ets Jun  8 2016 00:22:57

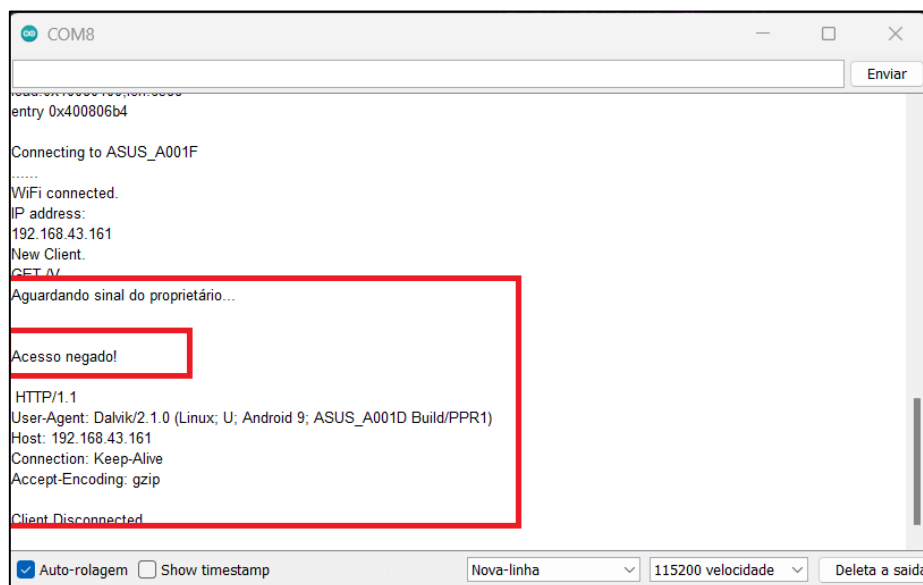
rst:0x1 (POWERON_RESET),boot:0x17 (SPI_FAST_FLASH_BOOT)
config: 0, SPIWP:0xee
clk_drv:0x00,q_drv:0x00,d_drv:0x00,cs0_drv:0x00,hd_drv:0x00,wp_drv:0x00
[checked] Auto-rolagem [ ] Show timestamp Nova-linha 115200 velocidade Deleta a saída
```

Fonte: Próprio autor.

9.1.1. Validação de funcionalidades de usuários com o módulo biométrico

Para essa validação, foi analisado as mensagens mandadas pelo microcontrolador pela porta serial, e verificado as amostragens de informações, sendo a validação de usuário como primeiro teste feito, como mostra a Figura 104.

Figura 104: teste de acesso negado de usuários não cadastrados.



```
COM8
Enviar

entry 0x400806b4

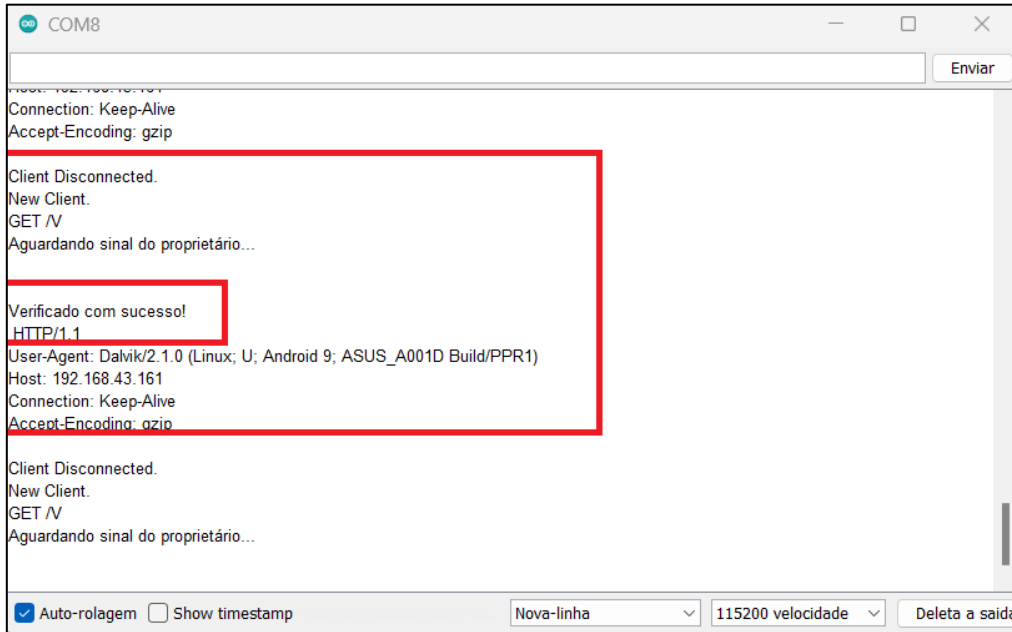
Connecting to ASUS_A001F
.....
WiFi connected.
IP address:
192.168.43.161
New Client.
GET /
Aguardando sinal do proprietário...
Acesso negado!
HTTP/1.1
User-Agent: Dalvik/2.1.0 (Linux; U; Android 9; ASUS_A001D Build/PPR1)
Host: 192.168.43.161
Connection: Keep-Alive
Accept-Encoding: gzip
Client_Disconnected

[checked] Auto-rolagem [ ] Show timestamp Nova-linha 115200 velocidade Deleta a saída
```

Fonte: Próprio autor.

Como segunda validação, foi verificado o teste de acesso permitido de usuários cadastrados no sistema, como mostra a Figura 105.

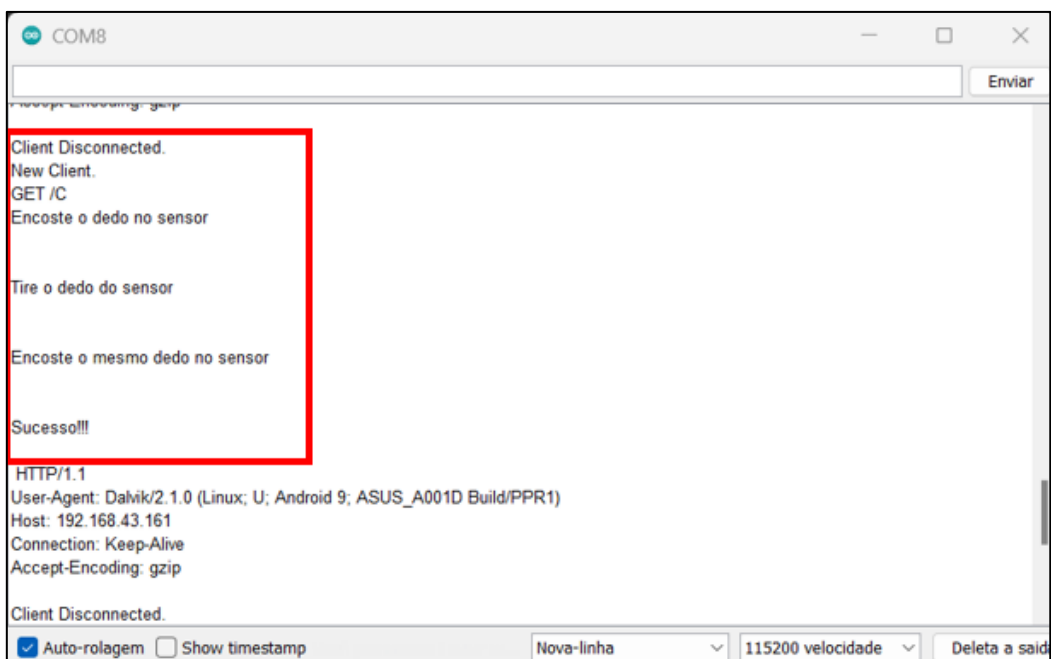
Figura 105: Teste de acesso permitido de usuários.



Fonte: Próprio autor.

Por fim, foi validado a funcionalidade de cadastro de usuários, inserindo novas digitais no leitor biométrico como mostra a Figura 106.

Figura 106: Teste de cadastro de usuários.

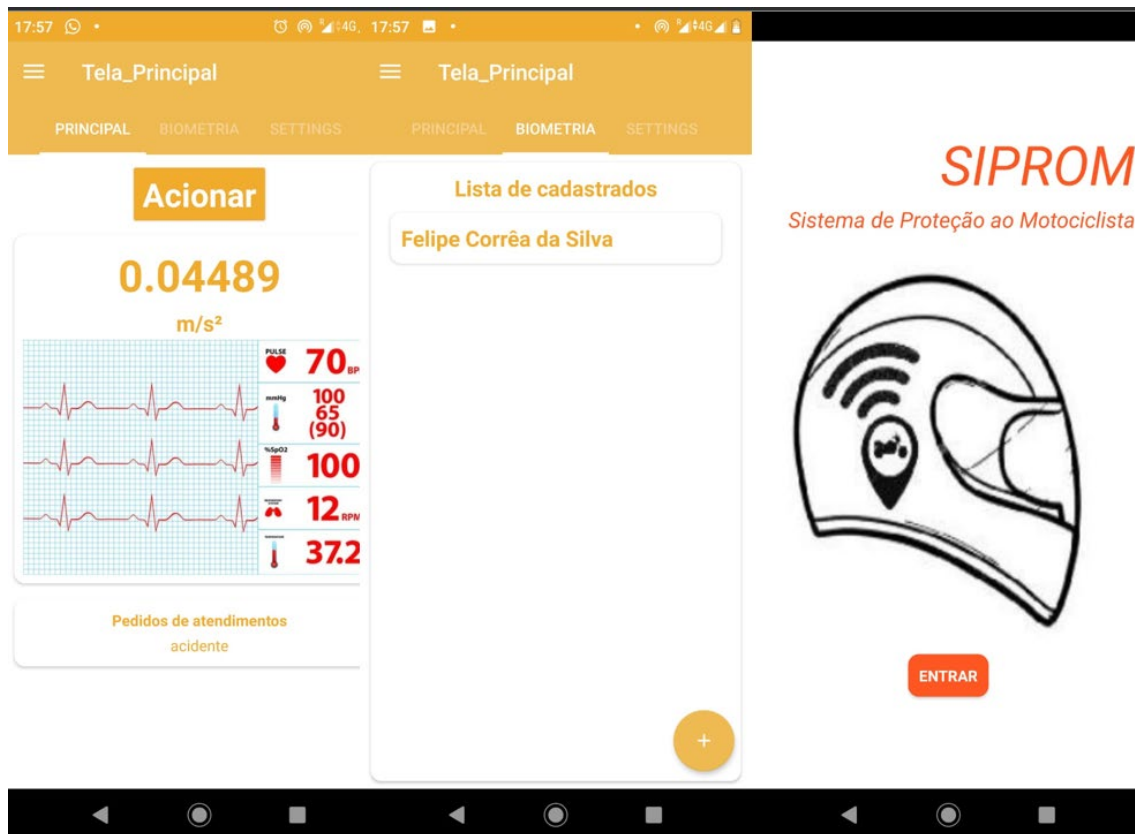


Fonte: Próprio autor.

9.1.2. Validação do aplicativo *mobile*

Foi verificada as funcionalidades dos componentes de interação do aplicativo com o usuário, verificando cada tela desenvolvida, como mostra a Figura 107.

Figura 107: Telas do aplicativo desenvolvido.



Fonte: Próprio autor.

9.1.3. Validação em campo do sistema de antifurto e roubo de motocicletas

Logo depois dos sistemas terem sido instalados em uma moto real, foi feito o teste de validação de usuários para ignição do veículo, onde inicialmente foi validado uma digital não cadastrada, onde a moto não deu partida, como mostra a figura

Figura 108: Teste de uma digital não cadastrada no sistema instalado na moto.



Fonte: Próprio autor.

Figura 109: Teste de uma digital cadastrada no sistema instalado na moto.



Fonte: Próprio autor.

9.2. VALIDAÇÃO DO SISTEMA DE DETECÇÃO DE ACIDENTE

Primeiramente, foi feito uma simulação de acidente, fazendo o processo de inclinar a moto e tirando a pressão do banco, onde estaria o piloto, como mostra Figura 110, assim sendo, o sistema respondeu a simulação e deu início aos passos de chamado de socorro.

Figura 110: Simulação de um acidente com o sistema instalado na moto.



Fonte: Próprio autor.

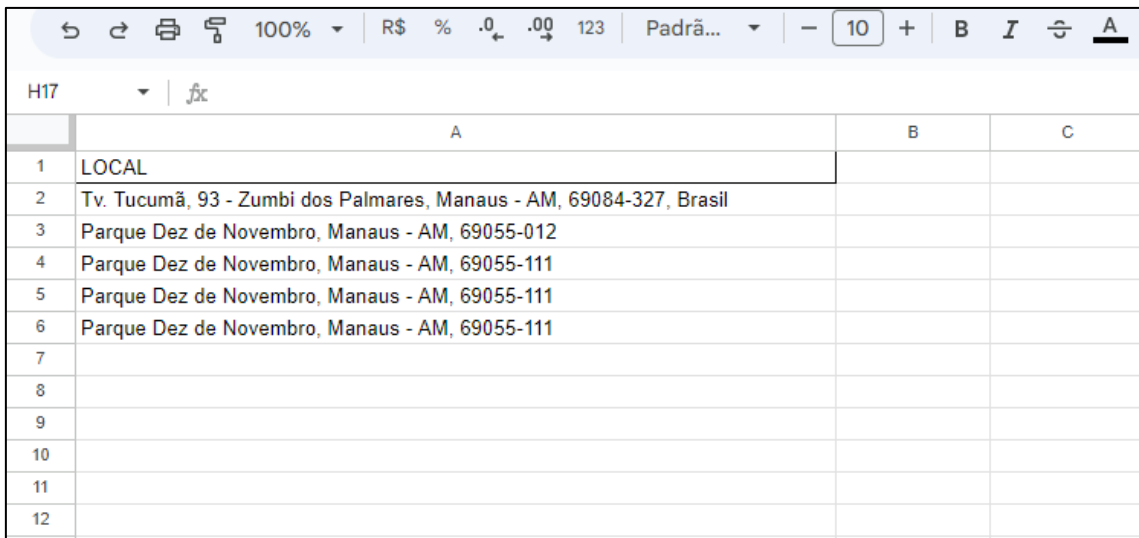
O sistema validou o acidente com os dados recebidos e iniciou o processo de envio de chamado de socorro junto a localização para o banco de dados como mostra a Figura 111.

Figura 111: Aplicativo *mobile* detectando um acidente.



Dentro do banco de dados foi recebido as informações de localização do acidente, bem como a listagem dos locais, como mostra Figura 112. Dando prosseguimento, o aplicativo web recebeu as informações de socorro e dados de coordenadas como mostra Figura 113.

Figura 112: Banco de dados recebendo os dados de localização.

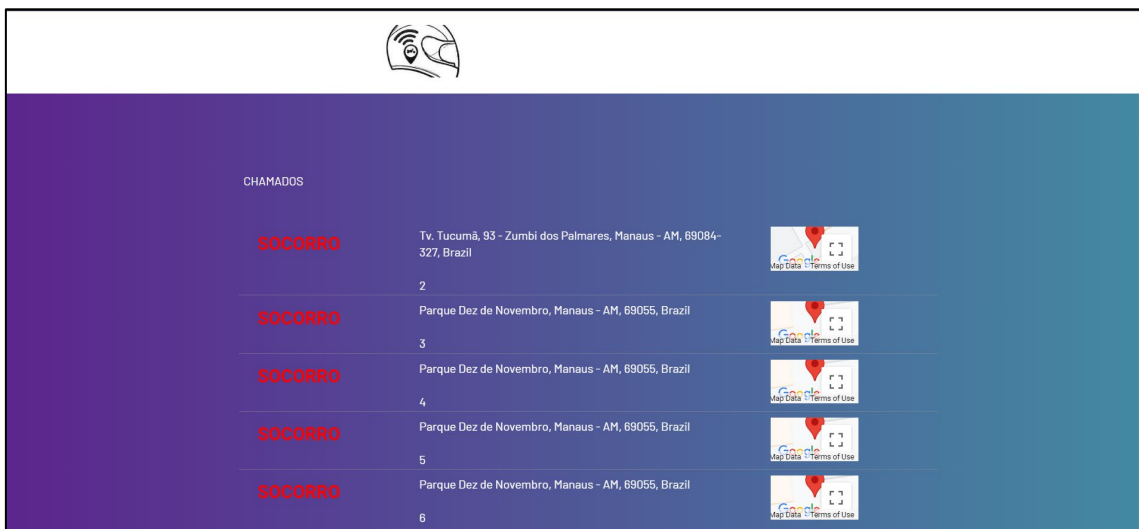


The image shows a screenshot of a spreadsheet application. The interface includes a top toolbar with various icons and a status bar. The spreadsheet has a header row with columns labeled A, B, and C. The data rows are as follows:

	A	B	C
1	LOCAL		
2	Tv. Tucumã, 93 - Zumbi dos Palmares, Manaus - AM, 69084-327, Brasil		
3	Parque Dez de Novembro, Manaus - AM, 69055-012		
4	Parque Dez de Novembro, Manaus - AM, 69055-111		
5	Parque Dez de Novembro, Manaus - AM, 69055-111		
6	Parque Dez de Novembro, Manaus - AM, 69055-111		
7			
8			
9			
10			
11			
12			

Fonte: Próprio autor.

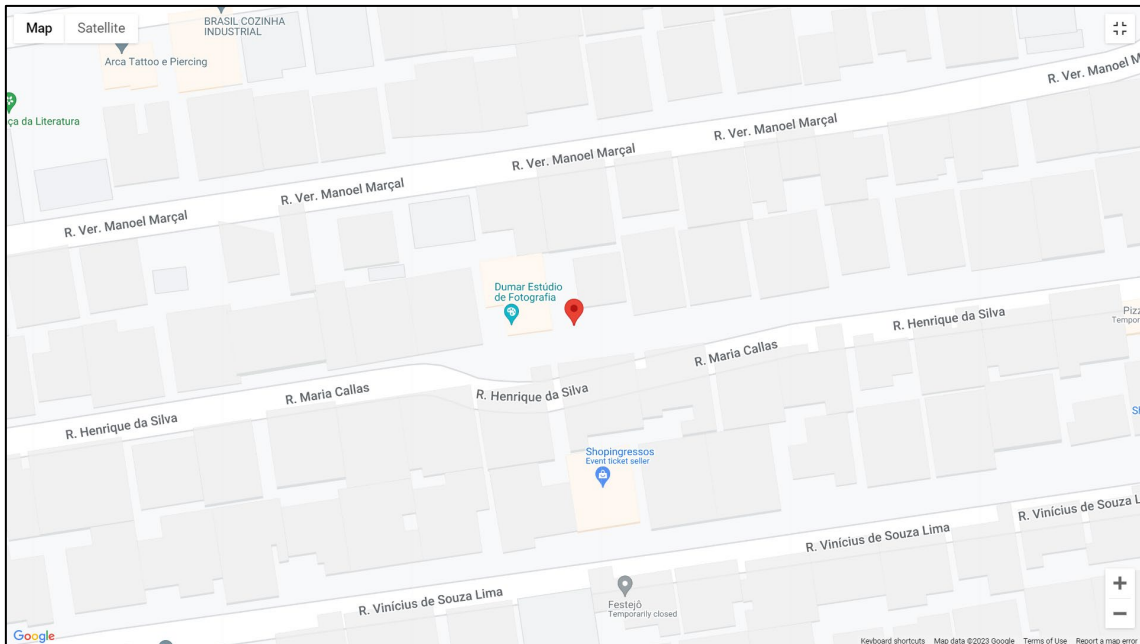
Figura 113: Aplicativo de amostras de chamados.



Fonte: Próprio autor.

O mapa desenvolvido recebeu a informação e traduziu em coordenadas de localização do google maps, como mostra Figura 114, sendo que também é possível ver a amostragem do mapa em satélite, como mostra Figura 115.

Figura 114: Localização do acidente no google maps.



Fonte: Próprio autor.

Figura 115: Localização do acidente em imagem de satélite.



Fonte: Próprio autor.

CONCLUSÃO

Assim sendo, através da experiência adquirida no desenvolvimento do projeto de conclusão de curso, e com a análise do impacto social do sistema criado, foi possível perceber que a extensão responsável pela prevenção contra furtos e roubos de moto pode proporcionar mais segurança aos usuários de motocicletas, inclusive trabalhadores que usam esse veículo como instrumento de trabalho, além de ser um equipamento auxiliar para os já existentes no mercado. Além disso, o sistema de bloqueio junto com o de localização pode proporcionar mais chances de recuperação de veículo caso sejam roubados, bem como controle e administração de outros usuários que compartilham a motocicleta pelo proprietário.

Logo, ao concluir o sistema de detecção de acidente e feita a análise aprofundada do ponto de vista da saúde social, foi verificado que tal projeto pode melhorar o tempo de pronto socorro de vítimas de acidentes envolvendo motociclistas, assim como, na parte de logística e precisão no percurso dos agentes de saúde.

Com a finalização de todo processo de criação e desenvolvimento dos sistemas de detecção de acidente e prevenção contra furtos e roubos, foram adquiridos conhecimentos aprofundados de cada segmento do trabalho descrito, possibilitando uma capacitação profissional e acadêmica das áreas mais modernas recorrentes no setor de engenharia atualmente, os quais envolviam estudo de sistemas de eletrônica junto com programação, codificação de aplicativos, bem como a manufatura de extensões físicas do projeto, como a placa de circuito impresso e peças 3D, em que tais áreas estavam dentro do projeto geral de forma intrínseca.

Sendo assim, após a conclusão do projeto, de forma conclusiva, o sistema obteve êxito em suas funcionalidades, em relação as expectativas dos objetivos específicos e geral descritos no trabalho, logo, o protótipo pode ser passivo de melhoramentos futuros.

TRABALHOS FUTUROS

Para os trabalhos futuros, por meio da experiência e visão adquirida do projeto, foi verificado a possibilidade de adição de comandos nas funcionalidades de administração de usuários no banco de dados, em se tratando de: modificação de digital, apagamento de usuário, além das notificações de acionamento de veículo para o aplicativo do proprietário e codificação de dados do banco de dados.

Em relação ao sistema de detecção de acidente, é possível a criação de um sistema de validação e confirmação do usuário usando uma interação com o sistema, bem como a adição de mais módulos que auxiliem na detecção deste tipo de situação de risco. Além disso, também seria feita uma integração com um relógio inteligente, o qual poderia mandar as informações de aferições de sinais vitais, sendo essas informações enviadas junto com o chamado de socorro e localização para um banco de dados.

Uma outra extensão, é o envio de informações pré-existentes do quadro de saúde do proprietário, as quais podem auxiliar no pré-tratamento do pronto socorro do usuário, para possibilidade de melhora até o atendimento no hospital.

REFERÊNCIAS BIBLIOGRÁFICAS

DE SIMONE, Marco C.; GUIDA, Domenico. Identification and control of a unmanned ground vehicle by using Arduino. UPB Sci. Bull. Ser. D, 2018, vol. 80, p. 141-154.

GANZ, Brian L.; LIEDBLAD, Benjamin Mike; THIEMANN, Henry. Method and system for controlling a vehicle with a smartphone. U.S. Patent No 9,569,954, 14 Feb. 2017.

HIDAYANTI, Fitria; RAHMAH, Fitri; WIRYAWAN, Aryadharma. Design of Motorcycle Security System with Fingerprint Sensor using Arduino Uno Microcontroller. International Journal of Advanced Science and Technology, 2020, vol. 29, no 05, p. 4374-4391.

HUSSIN, Muhamad Asyraf Mat; ZAINI, Norliza. Android-Based motorcycle safety notification system. En 2017 IEEE Conference on Systems, Process and Control (ICSPC). IEEE, 2017. p. 88-93.

RAUSS, Ryan. Combining Raspberry Pi and Arduino to form a low-cost, real-time autonomous vehicle platform. En 2016 American Control Conference (ACC). IEEE, 2016. p. 6628-6633.

PRATAMA, Fitriyanto Andy; RAKHMADI, Frida Agung. Design of Motorcycle Security System Using Fsr (Force Sensitive Resistor) Sensor, Arduino Uno Microcontroller and Sim800L Module. En Proceeding International Conference on Science and Engineering. 2019. p. 185-187.

PURWANTO, Kunnu; ISWANTO, Hariadi TK; MUHTAR, Muhammad Yusvin. Microcontroller-based RFID, GSM and GPS for motorcycle security system. Int. J. Adv. Comput. Sci. Appl, 2019, vol. 10, p. 447-451.

RAKUL, R. S.; RAVIA, S.; THIRUKKURALKANI, K. N. Implementation of vehicle mishap averting system using arduino microcontroller. Int. J. Eng. Res. Technol.(IJERT), 2016, vol. 5, no 4.

SAHA, Himadri Nath, et al. Smart Motorcycle Vest Using Arduino and Vibration Sensing Module. En 2018 9th IEEE Annual Ubiquitous Computing, Electronics & *Mobile* Communication Conference (UEMCON). IEEE, 2018. p. 1079-1085.

TOMBENG, Marchel Timothy; TAGHULIHI, Andrew Andreas; WAWORUNDENG, Jacqueline MS. Implementation of Wireless Xbee Authentication System of Motorcycle. CogITo Smart Journal, 2019, vol. 5, no 1, p. 45-55.

Vairavan, R., Kumar, S. A., Ashiff, L. S., & Jose, C. G. (2018). Obstacle Avoidance Robotic Vehicle Using Ultrasonic Sensor, Arduino Controller. International Research Journal of Engineering and Technology (IRJET), 2140-2142

FONTES CONSULTADAS

SOUZA,SILANE. **A cada 57 minutos uma moto é roubada em Manaus, segundo levantamento da SSP.** Disponível em: <http://www.acritica.com>. Acesso em: 10 de agosto de 2022.

LOBEL,FABRÍCIL. **Acidentes de moto deixam 2,5 milhões de pessoas com invalidez permanente em 10 anos.** Disponível em: <http://www.folha.uol.com.br>. Acesso em: 17 de agosto de 2022.

TEIXEIRA,WILLIAN. **Frota nacional de motos cresce mais de 90% em 10 anos.** Disponível em: <https://www.motociclistasmoonline.com.br> . Acesso em: 17 de agosto de 2022.

BOM DIA SP. **Acidentes fatais com motociclista aumentam quase 18% em um ano em SP.** Disponível em: <https://www.g1.globo.com> . Acesso em: 17 de agosto de 2022.

PORTARIA DETRAN. **Resolução nº245, de 27 de setembro de 2007.**Dispositivo antifurto, Alfredo Peres da Silva; Conselho Nacional do Trânsito. Acesso em: 21 de agosto de 2022

NODE.JSa. **Node.js®.** Disponível em: <<http://nodejs.org/>>. Acesso em: 09 agosto de 2022.

MORAIS, J. **ESP32 - Segurança e proteção da flash.** Disponível em: <<https://embarcados.com.br/protacao-da-flash-no-esp32/>>. Acesso em: 22 mar. 2023.

KODULAR. **Overview of Sensors - Kodular Docs.** Disponível em: <<https://docs.kodular.io/components/sensors/>>. Acesso em: 22 mar. 2023.

API - Bubble Docs. Disponível em: <<https://manual.bubble.io/core-resources/api>>. Acesso em: 22 mar. 2023.

HTTP: entenda o que é, para que serve e como funciona. **Rock Content**, 23 jan. 2019.

FOLHA DO ABC **Sobre os dados de furti.** 2021. Disponível em:<<http://www.folhadoabc.com.br/index.php/secoes/cidade/item/17025-abc-tem-alto-indice-de-roubo-e-furto-de-motocicletas>>. Acesso em: 02 de setembro de 2022.

de dados”, Disponível em: <<https://www.oracle.com/br/database/what-is-database/>>, Acesso em 20 de setembro de 2022.

WIRELESS, “GSM e GPS”, Disponível em:
<https://www.filipeflop.com/categoria/wireless-e-iot/gsm-e-gps/> Acesso em 22 de setembro de 2022.

MECÂNICA ONLINE, “Saiba quais são os veículos mais roubados e furtados no estado de São Paulo” disponível em: <https://mecanicaonline.com.br/2022/03/saiba-quais-modelos-de-motocicletas-sao-mais-roubados-e-furtados-no-estado-de-sp/> Acesso em 24 de setembro de 2022.

MOTONLINE. O objetivo do sistema de ignição é fornecer uma centelha (faísca gerada. Disponível em: <<https://www.motonline.com.br/noticia/o-objetivo-do-sistema-de-ignicao-e-fornecer-uma-centelha-faisca-gerada/>>.

Módulo GPS GY-NEO6MV2 – Guia completo de como usá-lo com o Arduino - BLOG MASTERWALKER SHOP. Disponível em:
<<https://blogmasterwalkershop.com.br/arduino/modulo-gps-gy-neo6mv2-guia-completo-de-como-usa-lo-com-o-arduino>>. Acesso em: 23 mar. 2023.

AMARAL, H. **EasyEDA - Simulador de circuitos on-line.** Disponível em:
<<https://embarcados.com.br/easyeda/>>. Acesso em: 23 mar. 2023.

MELLO, M. **Prototipagem de Placas de Circuito Impresso: entenda como funciona.** Disponível em: <<https://victorvision.com.br/blog/prototipagem-de-placas-de-circuito-impresso/>>. Acesso em: 23 mar. 2023.

APÊNDICE A – CÓDIGO FONTE DO MICROCONTROLADOR DO SISTEMA

```
#include <WiFi.h>
#include <Adafruit_Fingerprint.h>
#include "HX711.h"
HX711 balanca(3, 2);
float calibration_factor = 48011.00
float peso;
//Senha padrão do sensor de digitais
const uint32_t pass = 0x0;
int aux1=0, aux2=0, aux3=0, aux4=0;
Adafruit_Fingerprint fingerprintSensor = Adafruit_Fingerprint(&Serial2, pass);
const char* ssid      = "ASUS_A001F";
const char* password = "terminal7";
WiFiServer server(80);
int i=0;
void setup(){
    Serial.println("Remova todos os pesos da balança");
    delay(1000);
    /* atraso de 1000ms = 1s */
    Serial.println("Após estabilização das leituras, coloque o peso conhecido na balança");
    delay(1000);
    Serial.println("Pressione + para incrementar o fator de calibração");
    Serial.println("Pressione - para decrementar o fator de calibração");
    delay(1000);
    balanca.set_scale();
    balanca.tare();

    long zero_factor = balanca.read_average();
}

Serial.begin(115200);
pinMode(4, OUTPUT);      // set the LED pin mode
pinMode(2, OUTPUT);
digitalWrite(4, HIGH);
digitalWrite(2, LOW);
delay(10);
// We start by connecting to a WiFi network
```

```

Serial.print("\nConnecting to ");
Serial.println(ssid);
WiFi.begin(ssid, password);
boolean tr=true;
while (tr){
    i++;
    if (WiFi.status() != WL_CONNECTED) {
        digitalWrite(2, HIGH);
        delay(500);
        digitalWrite(2, LOW);
        delay(500);
        Serial.print(".");
    }
    else{
        tr = false;
        i=0;
    }
    if(i > 10){
        tr = false;
    }
}
if(i<10){
    Serial.println("\nWiFi connected.");
    Serial.println("IP address: ");
    Serial.println(WiFi.localIP());
    server.begin();
}
//Inicializa o sensor de digitais
delay(1000);
setupFingerprintSensor();
}
void setupFingerprintSensor(){
    //Inicializa o sensor
    fingerprintSensor.begin(57600);
    //Verifica se a senha está correta
    if(!fingerprintSensor.verifyPassword()){
        //Se chegou aqui significa que a senha está errada ou o sensor está
        problemas de conexão
    }
}

```

```

    Serial.println(F("Não foi possível conectar ao sensor. Verifique a senha
ou a conexão"));
    while(true);
}
}
int value = 0;
void loop(){
    Vector rawGyro = mpu.readGyroRaw();
    float gyroX = rawGyro.x;
    float gyroY = rawGyro.y;
    float gyroZ = rawGyro.z;
    Serial.print("Gyro X: ");
    Serial.print(gyroX);
    Serial.print(" Y: ");
    Serial.print(gyroY);
    Serial.print(" Z: ");
    Serial.println(gyroZ);
    sendData(gyroX, gyroY, gyroZ);
    delay(5000);
}
void sendData(float gyroX, float gyroY, float gyroZ) {
    if (WiFi.status() == WL_CONNECTED) {
        if (i>10){
            checkFingerprint();
        }
        aux2=0;
        if (i<10){
            WiFiClient client = server.available(); // listen for incoming clients
            if (client) { // if you get a client,
                Serial.println("New Client."); // print a message out the
                serial port
                String currentLine = ""; // make a String to hold
                incoming data from the client
                while (client.connected()) { // loop while the client's
                connected
                    if (client.available()) { // if there's bytes to read from
                    the client,
                        char c = client.read(); // read a byte, then
                        Serial.write(c); // print it out the serial
                        monitor

```

```

        if (c == '\n') { // if the byte is a newline
character
                // if the current line is blank, you got two newline characters in
a row.
                // that's the end of the client HTTP request, so send a response:
                if (currentLine.length() == 0){
                // HTTP headers always start with a response code (e.g. HTTP/1.1
200 OK)
                // and a content-type so the client knows what's coming, then a
blank line:
                //client.println("HTTP/1.1 200 OK");
                //client.println("Content-type:text/html");
                //client.println();

                // the content of the HTTP response follows the header:
                //
                client.print("Click <a href=\"/H\">here</a> to turn the LED on
pin 5 on.<br>");
                //
                client.print("Click <a href=\"/L\">here</a> to turn the LED on
pin 5 off.<br>");

                // The HTTP response ends with another blank line:
                //
                client.println();
                // break out of the while loop:
                break;
                }
                else{ // if you got a newline, then clear currentLine:
                currentLine = "";
                }
                }
                else if (c != '\r') { // if you got anything else but a carriage
return character,
                currentLine += c; // add it to the end of the currentLine
                }

        if (currentLine.endsWith("GET /C")) {
                storeFingerprint();
                client.println("HTTP/1.1 200 OK");
                client.println("Content-type:text/html");
                client.println();
                client.print("Cadastrada");
                client.println();

```

```

    }
    if (currentLine.endsWith("GET /V")) {
        checkFingerprint();
        if (aux2 == 1){
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println();
            client.print("Verificado com sucesso!");
            client.println();
        }
        else{
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html");
            client.println();
            client.print("Acesso negado!");
            client.println();
        }
    }
    if (currentLine.endsWith("GET /M")) {
    }
    if (currentLine.endsWith("GET /A")) {
        digitalWrite(4, LOW);
        delay(1500);
        digitalWrite(4, HIGH);
        client.println("HTTP/1.1 200 OK");
        client.println("Content-type:text/html");
        client.println();
        client.print("Verificado com sucesso!");
        client.println();
    }
    if (currentLine.endsWith("GET /B")) {
    }
}

// close the connection:
client.stop();
Serial.println("Client Disconnected.");
}

```

```

    }
}
//Exibe o menu no monitor serial
void printMenu()
{
    Serial.println();
    Serial.println(F("Digite um dos números do menu abaixo"));
    Serial.println(F("1 - Cadastrar digital"));
    Serial.println(F("2 - Verificar digital"));
    Serial.println(F("3 - Mostrar quantidade de digitais cadastradas"));
    Serial.println(F("4 - Apagar digital em uma posição"));
    Serial.println(F("5 - Apagar banco de digitais"));
}
//Espera até que se digite algo no monitor serial e retorna o que foi digitado
String getCommand()
{
    while(!Serial.available()) delay(100);
    return Serial.readStringUntil('\n');
}
//Cadastro da digital
void storeFingerprint()
{
    // Serial.println(F("Qual a posição para guardar a digital? (1 a 149)"));
    //Lê o que foi digitado no monitor serial
    // String strLocation = getCommand();
    //Transforma em inteiro
    // int location = strLocation.toInt();
    int location = 1;
    //Verifica se a posição é válida ou não
    if(location < 1 || location > 149)
    {
        //Se chegou aqui a posição digitada é inválida, então abortamos os
        próximos passos
        Serial.println(F("Posição inválida"));
        return;
    }
    Serial.println(F("\nEncoste o dedo no sensor\n"));
    //Espera até pegar uma imagem válida da digital
    while (fingerprintSensor.getImage() != FINGERPRINT_OK);
}

```

```

//Converte a imagem para o primeiro padrão
if (fingerprintSensor.image2Tz(1) != FINGERPRINT_OK)
{
    //Se chegou aqui deu erro, então abortamos os próximos passos
    Serial.println(F("Erro image2Tz 1"));
    return;
}
Serial.println(F("\nTire o dedo do sensor\n"));
delay(2000);
//Espera até tirar o dedo
while (fingerprintSensor.getImage() != FINGERPRINT_NOFINGER);
//Antes de guardar precisamos de outra imagem da mesma digital
Serial.println(F("\nEncoste o mesmo dedo no sensor\n"));
//Espera até pegar uma imagem válida da digital
while (fingerprintSensor.getImage() != FINGERPRINT_OK);

//Converte a imagem para o segundo padrão
if(fingerprintSensor.image2Tz(2) != FINGERPRINT_OK)
{
    //Se chegou aqui deu erro, então abortamos os próximos passos
    Serial.println(F("Erro image2Tz 2"));
    return;
}
//Cria um modelo da digital a partir dos dois padrões
if(fingerprintSensor.createModel() != FINGERPRINT_OK)
{
    //Se chegou aqui deu erro, então abortamos os próximos passos
    Serial.println(F("Erro createModel"));
    return;
}
//Guarda o modelo da digital no sensor
if(fingerprintSensor.storeModel(location) != FINGERPRINT_OK)
{
    //Se chegou aqui deu erro, então abortamos os próximos passos
    Serial.println(F("Erro storeModel"));
    return;
}
//Se chegou aqui significa que todos os passos foram bem sucedidos

```



```

    Serial.println(F("\nSucesso!!!\n"));
    aux1 = 1;
}
//Verifica se a digital está cadastrada
void checkFingerprint()
{
    Serial.println(F("\nAguardando sinal do proprietário...\n"));
    //Espera até pegar uma imagem válida da digital
    while (fingerprintSensor.getImage() != FINGERPRINT_OK);
    //Converte a imagem para o padrão que será utilizado para verificar com o
    banco de digitais
    if (fingerprintSensor.image2Tz() != FINGERPRINT_OK)
    {
        //Se chegou aqui deu erro, então abortamos os próximos passos
        Serial.println(F("Erro image2Tz"));
        return;
    }
    //Procura por este padrão no banco de digitais
    if (fingerprintSensor.fingerFastSearch() != FINGERPRINT_OK)
    {
        //Se chegou aqui significa que a digital não foi encontrada
        Serial.println(F("\nAcesso negado!\n"));
        aux2 = 0;
        return;
    }
    //Se chegou aqui a digital foi encontrada
    //Mostramos a posição onde a digital estava salva e a confiança
    //Quanto mais alta a confiança melhor
    // Serial.print(F("Digital encontrada com confiança de "));
    // Serial.print(fingerprintSensor.confidence);
    // Serial.print(F(" na posição "));
    // Serial.println(fingerprintSensor.fingerID);
    Serial.print(F("\nVerificado com sucesso!\n"));
    aux2 = 1;
    digitalWrite(4, LOW);
    delay(1500);
    digitalWrite(4, HIGH);
}
void printStoredFingerprintsCount()

```

```

{
    //Manda o sensor colocar em "templateCount" a quantidade de digitais salvas
    fingerprintSensor.getTemplateCount();
    //Exibe a quantidade salva
    Serial.print(F("Digitais cadastradas: "));
    Serial.println(fingerprintSensor.templateCount);
}

void deleteFingerprint()
{
    // Serial.println(F("Qual a posição para apagar a digital? (1 a 149)"));
    //Lê o que foi digitado no monitor serial
    // String strLocation = getCommand();
    //Transforma em inteiro
    // int location = strLocation.toInt();
    int location = 1;
    //Verifica se a posição é válida ou não
    if(location < 1 || location > 149)
    {
        //Se chegou aqui a posição digitada é inválida, então abortamos os
        //próximos passos
        Serial.println(F("Posição inválida"));
        return;
    }
    //Apaga a digital nesta posição
    if(fingerprintSensor.deleteModel(location) != FINGERPRINT_OK)
    {
        Serial.println(F("Erro ao apagar digital"));
    }
    else
    {
        // Serial.println(F("Digital apagada com sucesso!!!"));
        aux1 = 0;
        aux2 = 0;
    }
}

void emptyDatabase()
{
    Serial.println(F("Tem certeza? (s/N)"));
    //Lê o que foi digitado no monitor serial

```

```

String command = getCommand();
//Coloca tudo em maiúsculo para facilitar a comparação
command.toUpperCase();
//Verifica se foi digitado "S" ou "SIM"
if(command == "S" || command == "SIM")
{
    Serial.println(F("Apagando banco de digitais..."));
    //Apaga todas as digitais
    if(fingerprintSensor.emptyDatabase() != FINGERPRINT_OK)
    {
        Serial.println(F("Erro ao apagar banco de digitais"));
    }
    else
    {
        Serial.println(F("Banco de digitais apagado com sucesso!!!"));
    }
}
else
{
    Serial.println(F("Cancelado"));
}

balanca.set_scale(calibration_factor);
/* a balança está em função do fator de calibração */

Serial.print("Peso: ");
/* Printa "Peso:" na COM */

peso = balanca.get_units(), 10;
/* imprime peso */

if (peso < 0)
/* se a unidade for menor que 0 será considerado 0 */
{
    peso = 0.00;
/* Para o caso do peso ser negativo, o valor apresentado será 0 */
}

Serial.print(peso);
/* Printa o peso na serial */

Serial.print(" kg");
/* Printa "kg" na serial */

Serial.print(" Fator de calibração: ");
/* Printa "Fator de calibração:" na serial */

Serial.print(calibration_factor);
/* Printa o fator de calibração na serial */

Serial.println();
/* Pula linha no serial */

```

```

    delay(500);
    /* atraso de 500ms = 0.5s*/

    if(Serial.available())
    /* caso sejam inseridos caracteres no serial */

    {

        char temp = Serial.read();

        if(temp == '+')
        /* Se o + for pressionado */

            calibration_factor += 1;
        /* incrementa 1 no fator de calibração */

        else if(temp == '-')
        /* Caso o - seja pressionado */

            calibration_factor -= 1;
        /* Decrementa 1 do fator de calibração */

    }

    float force = readCells();

    Serial.print("Forca total: ");

    Serial.println(force);

    sendData(force);

    delay(5000);

}

float readCells() {

    int cellReading1 = analogRead(CELL_PIN_1);

    int cellReading2 = analogRead(CELL_PIN_2)

    float peso1 = map(cellReading1, 0, 4095, 0, 3300) / 1000.0;

    float peso2 = map(cellReading2, 0, 4095, 0, 3300) / 1000.0;

    float force1 = peso1 / 0.5; // substitua o valor 0.5 pelo valor de
calibração da célula

    float force2 = peso2 / 0.5; // substitua o valor 0.5 pelo valor de
calibração da célula

    return force1 + force2; // retorna a força total em unidades de medida

}

void sendData(float force) {

    if (WiFi.status() == WL_CONNECTED) {

        HTTPClient http

        http.begin(serverName);

        http.addHeader("Content-Type", "application/json");

        String requestBody = "{\"force\": " + String(force) + "}";

        int httpResponseCode = http.POST(requestBody);

    }

}

```

```

if (gps.location.isValid()) {
    float latitude = gps.location.lat();
    float longitude = gps.location.lng();
    Serial.print("Latitude: ");
    Serial.println(latitude, 6);
    Serial.print("Longitude: ");
    Serial.println(longitude, 6);
    sendData(latitude, longitude);
}
while (GPS_SERIAL.available() > 0) {
    gps.encode(GPS_SERIAL.read());
}
}

void sendData(float latitude, float longitude) {
    if (WiFi.status() == WL_CONNECTED) {
        HTTPClient http;
        http.begin(serverName);
        http.addHeader("Content-Type", "application/json");
        String requestBody = "{\"latitude\": " + String(latitude, 6) + ",
        \"longitude\": " + String(longitude, 6) + "}";
        int httpResponseCode = http.POST(requestBody);
        if (httpResponseCode > 0) {
            Serial.print("Resposta do servidor: ");
            Serial.println(httpResponseCode);
        } else {
            Serial.println("Erro na requisicao");
        }
        http.end();
    } else {
        Serial.println("Erro na conexao WiFi");
    }
}
}

```