UNIVERSIDADE DO ESTADO DO AMAZONAS

ESCOLA SUPERIOR DE TECNOLOGIA

MATHEUS COELHO CARDOSO

DESENVOLVIMENTO DE UM PROTÓTIPO DE BICICLETA INTELIGENTE E MONITORADA APLICANDO A TECNOLOGIA IOT

MANAUS

2023

MATHEUS COELHO CARDOSO

DESENVOLVIMENTO UM PROTÓTIPO DE BICICLETA INTELIGENTE E MONITORADA APLICANDO A TECNOLOGIA IOT

Projeto de pesquisa desenvolvido durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia de Controle e Automação da Universidade do Estado do Amazonas, como pré-requisito para obtenção do título Engenheiro de Controle e Automação.

Orientador: Prof. Me. Cleto Cavalcante de Souza Leal.

MANAUS

2023

MATHEUS COELHO CARDOSO

DESENVOLVIMENTO DE UM PROTÓTIPO DE BICICLETA INTELIGENTE E MONITORADA APLICANDO A TECNOLOGIA IOT

Projeto de pesquisa desenvolvido durante a disciplina de Trabalho de Conclusão de Curso II e apresentada à banca avaliadora do Curso de Engenharia de Controle e Automação da Universidade do Estado do Amazonas, como pré-requisito para obtenção do título Engenheiro de Controle e Automação.

Manaus, 14 de Setembro de 23.

Nota obtida: 9 (Nove Pontos)

Aprovado em 14 / 07 / 2023. Área de concentração: Sistemas Embarcados

BANCA EXAMINADORA

.....

Israel Mazaira Morales

.....

Moisés Pereira Bastos

.....

Marivan Silva Gomes

MANAUS

2023

RESUMO

Esse trabalho pretende de desenvolver um protótipo de bicicleta inteligente, também conhecido por SmartBikes, que monitora os dados em tempo real como localização, velocidade e inclinação da bicicleta. O projeto é focado como objetivo de desenvolver um meio visual para monitoramento dos dados para o ciclista, por meio de uma plataforma mobile para visualização de dados, de uma forma simples e intuitiva. É proposto realizar a comunicação dos dados para o aplicativo em forma de computação em nuvem, conhecido Internet das Coisas (Internet of Things - IoT). O microcontrolador ESP32 possibilita o envio de dados coletados pelos sensores, através da conexão Bluetooth usado no projeto. Para comunicação e criação do modelo de tela para funcionar como dashboard, foi utilizado o software BlynkIoT, configurando os dados e criando o modelo do projeto para visualização dos dados do ciclista em tempo real. Por fim, o desenvolvimento do protótipo promoveu um ambiente de sistema supervisório para aquisição de dados, podendo melhorar a experiência do ciclista no treino e ajudar em tomada de decisões.

Palavras-chave: Sistema de controle, Internet das Coisas, Bicicletas Inteligentes.

ABSTRACT

This paper intends to develop a prototype of a smart bicycle, also known as SmartBikes, which monitors real-time data such as location, speed and inclination of the bicycle. The project is focused on the objective of developing a visual means for monitoring data for the cyclist, through a mobile platform for data visualization, in a simple and intuitive way. It is proposed to carry out data communication to the application in the form of cloud computing, known as the Internet of Things (IoT). The ESP32 microcontroller makes it possible to send data collected by the sensors through the Bluetooth connection used in the project. For communication and creation of the screen model to function as a dashboard, the BlynkIoT software was used, configuring the data and creating the project model for viewing the cyclist's data in real time. Finally, the development of the prototype promoted a supervisory system environment for data acquisition, which could improve the cyclist's training experience and help him in decision making.

Key Words: Control Systems, Internet of Things, SmartBikes.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

DC – *Direct Current;*

GPS – *Global Positioning System;*

I2C – *Inter-Integrated Circuit;*

I2S – *Inter-IC Sound*;

IHM – *Interface Homem Máquina;*

IDE – *Integrated Development Environment;*

IoT – *Internet Of Things*;

RFID – *Radio Frequency Identification*;

RTLS – *Real-Time Locating System;*

RX – *Receiver;*

SAP – *System Analysis Program Development;*

SPI – *Serial Peripheral Interface;*

TX – *Transceiver;*

UART – *Universal Asynchronous Receiver-Transmitter;*

USB – *Universal Serial Bus*;

UWB – *Ultra Wideband.*

SUMÁRIO

1 INTRODUÇÃO

A mobilidade urbana em um contexto geral tem enfrentado problemas de transporte e logística para a população, ocorrendo uma falta de mobilidade entre os modais de transporte, como, por exemplo, bicicletas, podendo ocasionar acidentes e danos materiais. Com isso, existem políticas de incentivos públicos ou privados para criação de programas de uso de bicicletas inteligentes, que melhoram a locomoção dos indivíduos e dos ciclistas, envolvendo projetos também sustentáveis (SALES DE AZEVEDO, MARQUES DE CASTRO, *et al.*, 2020).

Com a implementação das tecnologias nos usos de bicicletas para a melhoria de sistemas de transporte, deu o início do conceito de *SmartBikes*, que são bicicletas tecnológicas que melhoram a experiência do ciclista, por meio de aplicativos, podendo propor uma segurança por meio da locomoção e implementação dos meios sustentáveis para a sociedade. (OLIVEIRA, NERY, *et al.*, 2021).

Propõe-se também o uso de tecnologias que monitoram os dados em tempo real do ciclista, focando na sua saúde e nas condições do ambiente, como também na sua posição geográfica em tempo real. Com isso, o conceito de bicicletas inteligentes se abrangem, podendo ter aplicações em saúde e em transporte (MUNDADA, MUKKAMALA, 2020).

A Internet das Coisas (*Internet of Things*) é uma das tecnologias que buscam solucionar o gerenciamento, monitoramento e processamento dos dados de bicicletas inteligentes, conectando os elementos de físicos como sensores e uma aplicação digital e visual para o usuário.

Em geral, a IoT é uma rede de dispositivos físicos (coisas) com software e sensores especiais incorporados, que permitem conectar e compartilhar dados por meio de conexão na nuvem e um protocolo de comunicação, sendo que esses *devices* podem ser *smartphones* e *wearables*, que são dispositivos vestíveis (MAIO, AFONSO, 2015).

Com base na ampla aplicação de tecnologias que buscam inovar a experiência dos ciclistas com o uso de *SmartBikes*, o objetivo geral do projeto é propor o desenvolvimento de um protótipo de bicicleta inteligente com uso de sensores, aplicando a tecnologia IoT para aquisição de dados de localização geográfica e monitoramento de queda, usando um sistema *mobile* para virtualização desses dados em tempo real.

Visando atingir o objetivo principal, são necessários alguns objetivos específicos que servem como requisito para o sucesso do projeto, que são eles:

- Monitorar os dados em tempo real como posição geográfica e as variáveis de aceleração, podendo ser em x, y ou z.
- Desenvolver um código para o sistema embarcado interagir com os sensores.
- Desenvolver em um aplicativo a tela de interface para o usuário.

O levantamento de dados e referências para o trabalho, foram pesquisados em artigos publicados em revistas, artigos publicados em conferências, outros trabalhos de conclusão de curso e sites. Um dos projetos pesquisados que são relevantes a base de estudo e prática da metodologia desse trabalho é de um autor cujo artigo foi publicado no evento do IEEE sobre tecnologias de telecomunicações. Nesse projeto ele propõe construir um sistema de monitoramento de dados de localização em tempo real do ciclista, dados de sinais vitais de saúde como eletrocardiograma, oximetria e temperatura corporal, usando Arduino e um módulo ESP8266 para enviar esses dados em tempo real para um aplicativo usando o *Blynk* (MUHAMAD, RAZALI, *et al.*, 2020).

2 REFERENCIAL TEÓRICO

2.1 BICICLETAS INTELIGENTES: UMA SOLUÇÃO PARA MOBILIDADE URBANA

O avanço das tecnologias inovadoras de rádio frequência e da computação em nuvem, permitiu a aplicabilidade de projetos em diversas áreas como, por exemplo, as cidades inteligentes.

O termo *Smart Cities* (Cidades Inteligentes) consiste em usar a tecnologia para melhorar a infraestrutura urbana e tornar os meios modais mais organizados e melhores em se viver, promovendo a viabilidade e a prosperidade de áreas urbanas (NEIROTTI, DE MARCO, *et al.*, 2014). Esse conceito surgiu no Fórum Mundial de 1997 sobre Cidades Inteligentes, em que reuniram os principais países da Europa, com o intuito de criar uma rede inteligentes nessas cidades, traçando objetivo estratégico para atingir as metas estipuladas nesse fórum (HOLLANDS, 2008).

Dentre dos meios de transportes que mais se destacam em projetos, que envolve a mobilidade urbana nas cidades inteligentes, são as bicicletas. As *Smart Bikes* (bicicletas inteligentes) são um meio de mobilidade urbana inteligente, com o transporte de dados em tempo real à internet, usando sensores para coletar dados, analisar e interagir com o usuário, por meio de *smartphones* usando um aplicativo como interface. As bicicletas inteligentes também contam com GPS para localização do ciclista e sua velocidade (DE ARAUJO, MOREIRA, *et al.*, 2020)

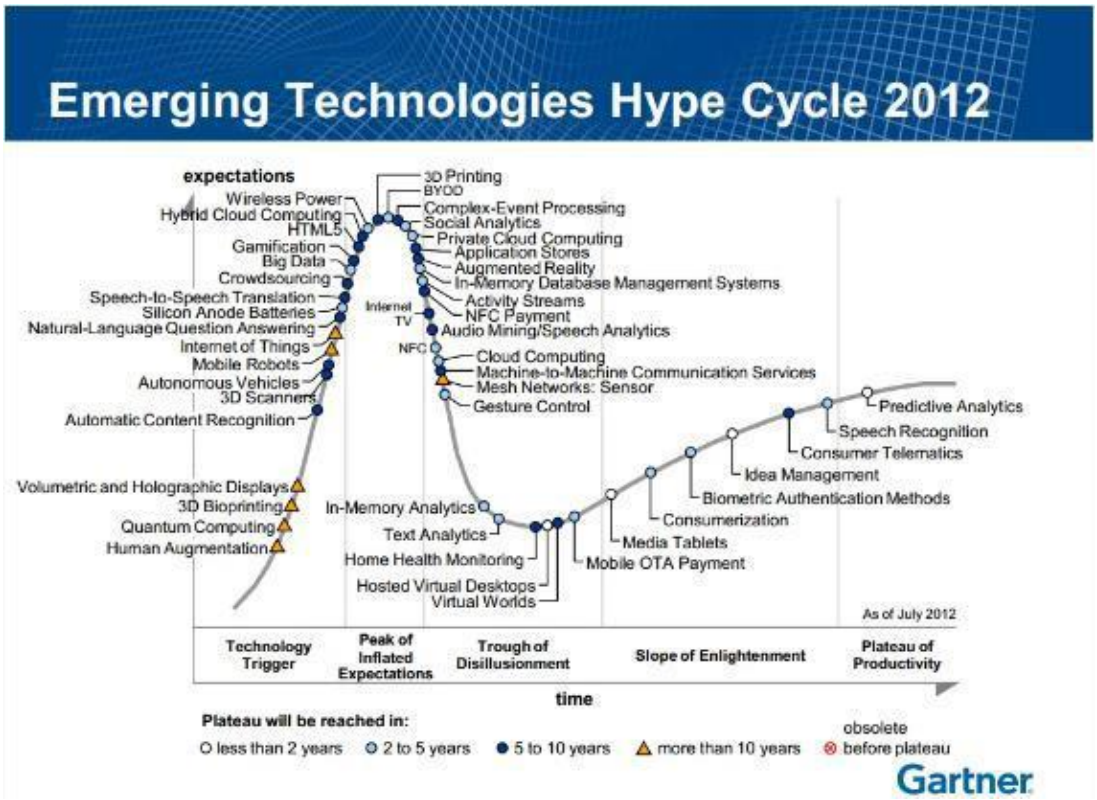
2.2 INTERNET OF THINGS

2.2.1 A TECNOLOGIA DA INTERNET DAS COISAS

Uma das primeiras menções sobre esse termo *Internet of Things* surgiu em 1999 por Kevin Ashton pelo trabalho "*I made at Procter & Gamble"*, em que os projetos propostos na comunicação entre cadeia de suprimentos fossem conectados a uma rede, que nesse trabalho era por Identificação de Rádio Frequência (KEVIN ASTHON, 2010).

Ao decorrer dos anos, a tecnologia IoT foi ganhando popularidade no âmbito acadêmico e industrial. Segundo os especialistas das áreas de tecnologia emergentes, a empresa de consulta de tecnologia, a Gartner, publicou no seu artigo anual chamado *Emerging Technologies Hype Cycle 2012*, um gráfico de tecnologias do quais os mesmos estão no âmbito das ideias, sendo desenvolvidos ou em patamar de produção.

Figura 1: Ciclo de *hype* para tecnologias emergentes no ano de 2012.

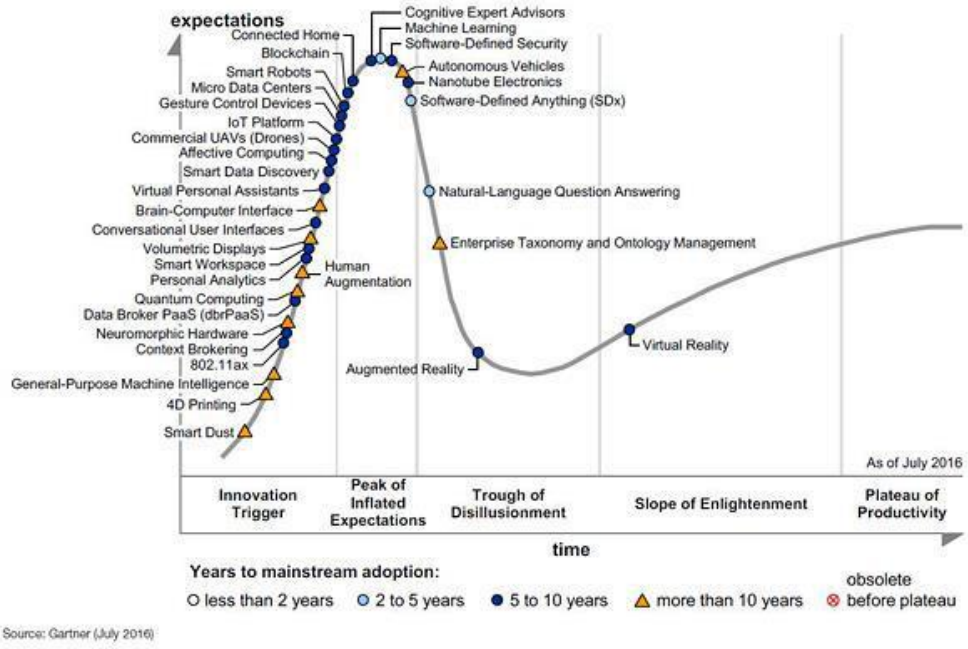


Fonte: Gartner, 2012.

Nesse ano, a Internet das Coisas estava com previsão com mais de 10 anos para ser adotado pelo mercado, entretanto, ganhou uma grande notoriedade e investimento nos projetos de pesquisa que diminui esse tempo para 3 anos, e também como surgimento das primeiras plataformas de IoT para atender a demanda dessa tecnologia (SANTOS, SILVA, *et al.*, 2016).

Da figura 2, que mostra o ciclo de *hype* das tecnologias emergentes de 2016, segundo do *Gartner*, se pode notar que a tecnologia em questão não aparece no gráfico e que essas plataformas estão sendo estudadas para serem utilizadas, e essas plataformas são, por exemplo, AWS*Cloud*, *SAP*, *ORACLE* e dentre outros.

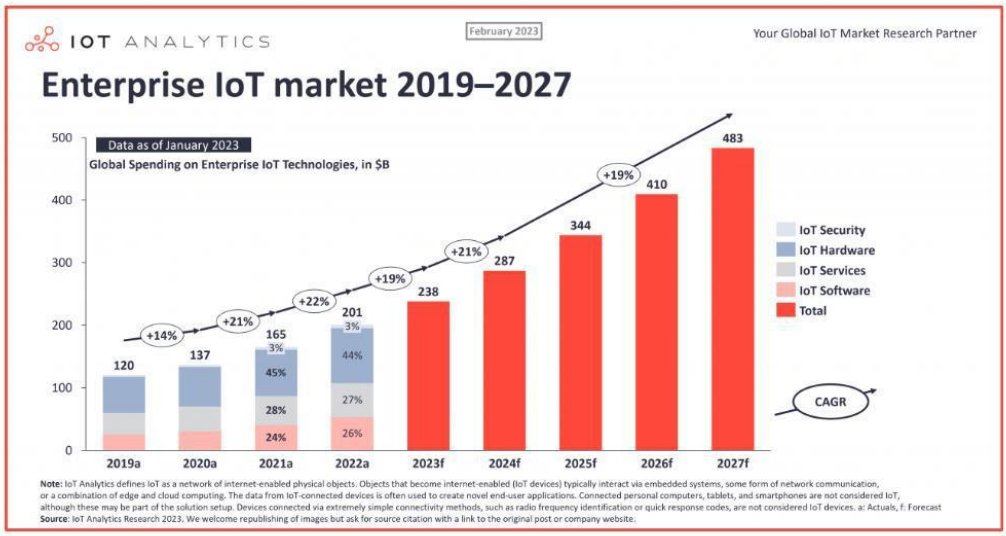
Figura 2: Ciclo de *hype* para tecnologias emergentes no ano de 2016.



Fonte: Gartner, 2016.

Com o avanço dos estudos e das diversas aplicações com a tecnologia IoT, diversas empresas de tecnologia estão investindo cada vez mais pesado em internet das coisas. Segundo o artigo da revista *IoT Analytics* (WEGNER, 2023), a estimativa é que até 2027 o mercado da tecnologia IoT cresça em 19% em projetos, gastando cerca de 400 bilhões de dólares em diversos projetos. O gráfico abaixo mostra o gasto das empresas em bilhões de dólares com o mercado da tecnologia da Internet das Coisas.

Figura 3: Crescimento de investimento global em projetos que envolvem a tecnologia IoT.



Fonte: IoT Analytics, 2023.

2.2.2 A DEFINIÇÃO DA INTERNET DAS COISAS

Por definição, a Internet das Coisas é uma rede que proporciona diversos dispositivos se conectando à internet, podendo ser sensores, celulares ou até mesmo *wearables* – que são dispositivos inteligentes vestíveis. Dessa forma, os usuários são capazes de interagir com esses dispositivos conectados à rede, com a possibilidade de realizar o monitoramento de dados em tempo real, ou realizar ações sobre o sistema mediante uma aplicação de *software* (SANTOS, SILVA, *et al.*, 2016).

O fluxo de dados em projetos de IoT é baseado em alguns objetos inteligentes, que no final tem um consumidor para consumir esses dados, que são listados a seguir (KIM, MOON, 2018):

- **Sensores Inteligentes**: Dispositivos conjuntos que coletam dados do ambiente e transmite os fluxos de dados. Ex: Smartphones, roteadores, *wearables*, termômetros;
- **Centro de Dados**: Os dados transportados são armazenados por uma rede de computação em nuvem;
- ***Software***: Aplicações que mostram os dados coletados e fornecem serviços de controle para usuários;
- **Consumidor:** É o usuário final que interage com o software.

Figura 4: Aplicações da Internet das Coisas.

Fonte: Hackr.io, 2022.

Com essa nova forma de utilizar a internet para conectar os objetos no cotidiano, houve uma grande pesquisa e projetos aplicando uma automação no dia a dia por meio de uso de celulares ou computadores, podendo até ser incluidos sistemas de interface homem maquina (IHM) em casos de projetos industriais, incluindo diversos setores como *SmartGrids* usado no setor elétrico, *SmartHome* ou em setores hospitalares como *SmartHealth* (LEITE, MARTINS, *et al.*, 2017).

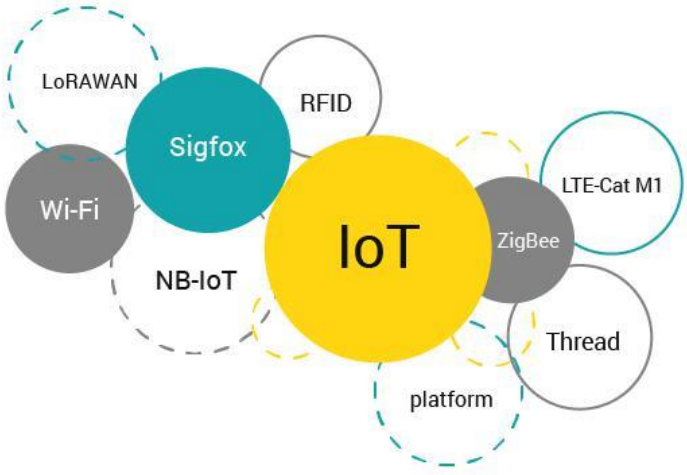
2.3 SISTEMAS DE LOCALIZAÇÃO EM TEMPO REAL (RTLS)

Com o crescimento do uso da Internet das Coisas ligados aos objetos no dia a dia na sociedade, houve-se a necessidade de conectar aos dispositivos fora do ambiente local, sendo esses chamados de localização *indoor*. Para solucionar esse problema, desenvolveu dispositivos com o Sistema de Posicionamento Global (GPS, do inglês *Global Positioning System*), usados para localizar objetos (DOS, LEANDRO, *et al.*, 2020).

Os projetos que evolvem o sistema RTLS (do inglês *Real-Time Locating System*), utilizam tecnologias de rádio-frequência para realizar a detecção do posicionamento de objetos em tempo real (MENEGOTTO, 2015), dos quais são: Wi-Fi, Bluetooth, ZigBee, UWB e RFID.

Como soluções futuras de projetos para cidades inteligentes, são destacados projetos que envolvem *SmartBikes* que usam sistemas de GPS para sua localização em sistemas outdoors, do qual o ambiente tende a ocorrer certas interferências externas. Então são usadas as tecnologias de rádio-frequência para atenuar os ruídos externos ou até mesma fazer que o sistema funcione através da frequência da tecnologia usada para transmissão de dados.

Figura 5: Tecnologias de transmissão usadas em projetos de IoT.

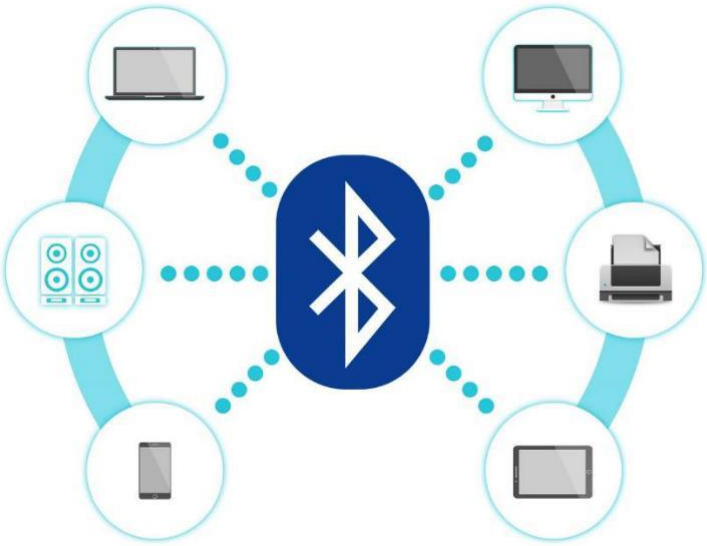


Fonte: AvSystem, 2019.

2.4 A TECNOLOGIA BLUETOOTH

No ano de 2010 surgiu uma tecnologia capaz de emitir sinais de rádio de curtas distâncias com objetivo de substituir redes de dispositivos a fio. Esse sistema também possui características que quando emitem ou recebem sinais, que consome uma baixa quantidade de energia, denominados de *Bluetooth Low Energy* (BLE). Desde então, esses vêm ocupando espaço em projetos envolvendo sistemas de localização em tempo real (MENEGOTTO, 2015).

Figura 6: Conectividade do Bluetooth.



Fonte: Codingninjas, 2023.

Ao decorrer do surgimento de novas plataformas de IoT, a tecnologia Bluetooth foi aplicada massivamente em diversos projetos, fazendo que essa tecnologia de rádio frequência fossem bastante utilizados, tanto em ambientes acadêmicos quanto em diversos trabalhos de *SmartCities*.

2.5 O MICROCONTROLADOR ESP32

O microcontrolador da família ESP iniciou-se por uma empresa chinesa chamada *Espressif* e seu primeiro chip foi o ESP8266. Ao passar dos anos, a empresa evoluiu os seus recursos e em 2016 criou o primeiro *chip* do ESP32 com preço mais baixo que o chip anterior. Esse novo módulo tem canais de conversores Analógico-Digital e portas com protocolo de comunicação I2C, como também alocação de sensores nos pinos e alimentação de 3,3 Volts em corrente contínua (SYSTEMS, 2023).

Existem também módulos da família desse microcontrolador com a possibilidade de trabalhar com transmissões em rádio frequência, podendo ser de curto alcance como *Bluetooth* e de longo alcance como a rede LoRa (*Long Range*).

Figura 7: Microcontrolador ESP-WROOM-32.

Fonte: Usinainfo, 2019.

A seguir, temos uma tabela comparativa entre esses dois microcontroladores dessa empresa, com suas características técnicas (RIOS, ANDRADE, *et al.*, 2020):

Tabela 1 - Características técnicas entre ESP32 e ESP8266.

Microcontroladores		
Chip	ESP32	ESP8266
N.º Processadores	2	1
Arquitetura	32 Bit	32 Bit
CPU Freq. (Max)	240MHz	160MHz
Bluetooth	Sim	Não
SRAM	520Kb	50Kb
FLASH	4MB	4MB
GPIO	36	17
ADC Pinos	16	1
Sensor Touch	10	0
Sensor Hall	1	0
Sensor Temperatura	1	0

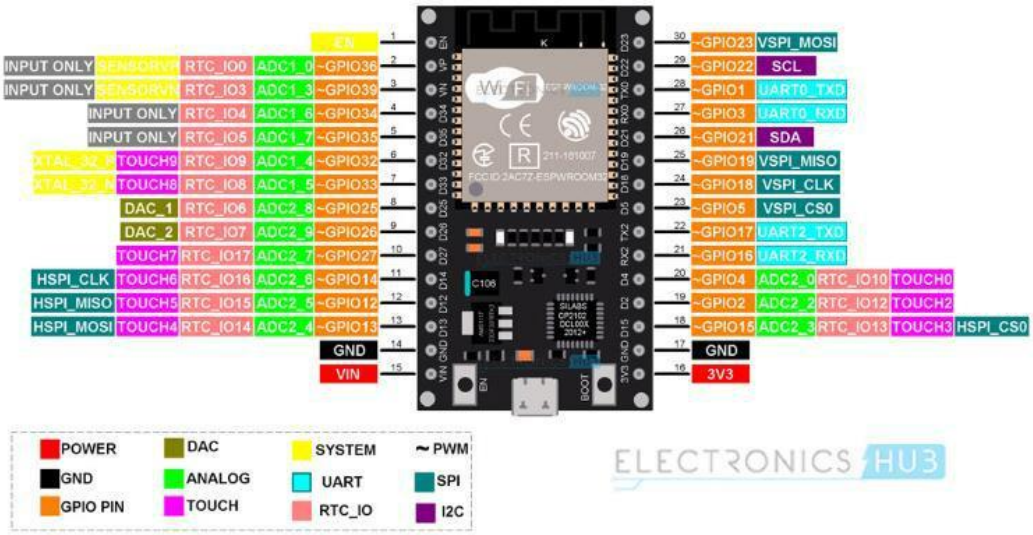
Fonte: Próprio Autor.

O microcontrolador ESP32 tem suas principais características de *hardware* específicas que podem variar de acordo com módulo, dos quais são (PRAVALIKA, PRASAD, 2019):

- Tensão de Operação: 3.3V DC.
- Corrente máxima: 12mA.
- Microprocessador: *Dual core Tensilica Xtensa 32-bit LX6.*
- Tecnologia *Bluetooth* v4.2 (*Bluetooth Low Energy*).
- Tecnologia *WiFi* padrão 802.11 B/G/N, na faixa de 2.4 a 2.5GHz.
- Interfaces *Ethernet*, I2C, I2S, SPI, UART.

Dentro das famílias do microcontrolador ESP32, existe o módulo chamado ESP-WROOM-32, no qual contém 4 MB SPI Flash, frequência do *clock* de 40MHz.Como todos outros módulos, na placa tem porta USB, botões *RESET* e *BOOT*.

Figura 8: Pinagem da ESP32.



Fonte: Eletronics Hub, 2021.

2.6 AS PLATAFORMAS IOT

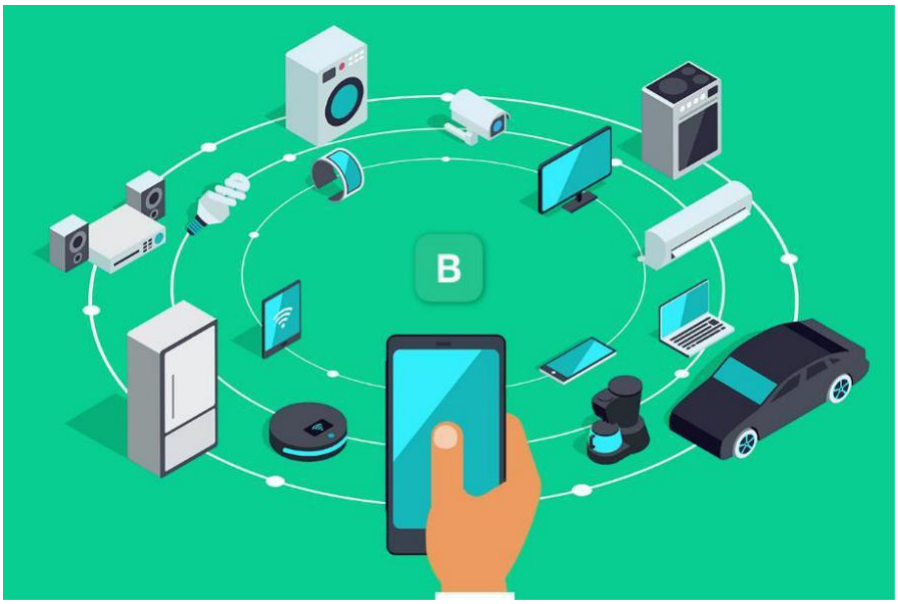
Com a crescente demanda de projetos envolvendo a área da Internet das Coisas, houve uma preocupação em desenvolver ferramentas focados em auxiliar a vida dos usuários, sejam em questão de visualização de dados ou até mesmo na interação dos dispositivos conectados a nuvem (CAROLINA, 2020). Com isso, as plataformas IoT tem um papel importante em fornecer as ferramentas para coletar, monitorar e analisar os dados dos dispositivos conectados em tempo real.

Logo, é possível criar projetos usando essas plataformas, pois cada um tem suas limitações, funcionalidades e se é pago ou código aberto (ISMAIL, HAMZA, *et al.*, 2019). Segundo a (WEGNER, 2023), as plataformas pagas mais usadas são AWS IoT, Oracle IoT Cloud e Google Cloud IoT. Os que são *openssource* (código aberto) são: ThingSpeak, Macchina, TagoIo e o Blynk, do qual este é uma das plataformas mais usadas que atendem aos usuários, com telas podendo ser *mobiles* ou *desktops*.

2.6.1 A FERRAMENTA BLYNK

Um das plataformas IoT open source disponível tanto para modelo *Desktop* ou mobile é o Blynk. O Aplicativo Blynk fornece um ambiente simples e didático para uma visualização de dados e implementação de funções de controle, para que o usuário possa interagir com o dispositivo, como botões, chaves e entre outros.

Figura 9: A internet das coisas com uso do BlynkIoT.



Fonte: BlynkIoT, 2018.

Esse aplicativo é um dos mais usados em projetos de IoT em menor escala, usando alguns sensores e com limitações de dados e transmissões. O protocolo de comunicação do aplicativo para o transporte das mensagens é baseado através do *cloud Blynk*. O servidor é responsável por armazenar os dados do aplicativo e da leitura dos sensores, podendo ser lidos até mesmo se o aplicativo estiver fechado.

2.7 OS SENSORES

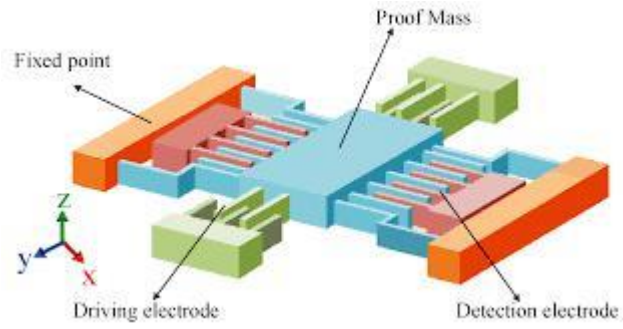
Em projetos que envolvem automação com uso de computação à nuvem, é necessário determinar as condições das variáveis do sistema, obtendo os seus dados em tempo real para o seu monitoramento. Os sensores são dispositivos eletrônicos sensíveis cujo objetivo é reagir a uma grandeza física, transformando em um valor mensurável, como temperatura, pressão, velocidade, posição e entre outras variáveis (WENDLING, 2013).

Em projetos em que usam um sistema de localização em tempo real aplicados em caso de *SmartBikes*, tem suas principais variáveis, dos quais são consideradas as mais avaliadas e estudadas são: posicionamento e a inclinação em relação a um dos seus eixos. Para leitura e monitoramento desses dados são utilizados os sensores como giroscópio e um sensor GPS com uma antena para transmissão da localização.

2.7.1 O SENSOR GIROSCÓPIO

Os giroscópios usam seu acelerômetro para extrair a aceleração linear do objeto em estudo, sendo esta variável usada para distinção com a aceleração da gravidade. Essa variável usado nesses sensores são medidas em três dimensões, com seus eixos x, y e z, podendo calcular a velocidade e deslocamento do objeto (MENEGUZZI, CENDRON, *et al.*, 2016).

Figura 10: Movimento do sensor giroscópio.



Fonte: PNGWing, 2018.

Nesses sensores, ao trabalharem com a aceleração linear, utilizam seus eixos diferente dos eixos em relação à Terra, por esses não trabalharem com a aceleração da gravidade. Para isso, os sensores precisam ser bastantes precisos, filtrando seus ruídos utilizando algoritmos que estimam o real valor lido pelo sensor (NORTH, FUX, *et al.*, 2008). Em projetos que envolvem a área de embarcados, um dos principais sensores mais utilizados é o MPU-6050, onde há bastante conteúdo de material nos setores acadêmicos e aplicados à IoT.

2.7.2 O SENSOR GPS

Os sensores GPS tem seu receptor com intuito de descobrir a distância entre os satélites, que revelam as informações de posição e velocidade quando conectados. Em geral, o sensor tem como referência três satélites e sua posição no espaço, onde o receptor pode identificar sua localização no espaço (DOS, LEANDRO, *et al.*, 2020).

Figura 11: Método de detecção do posicionamento do sensor.

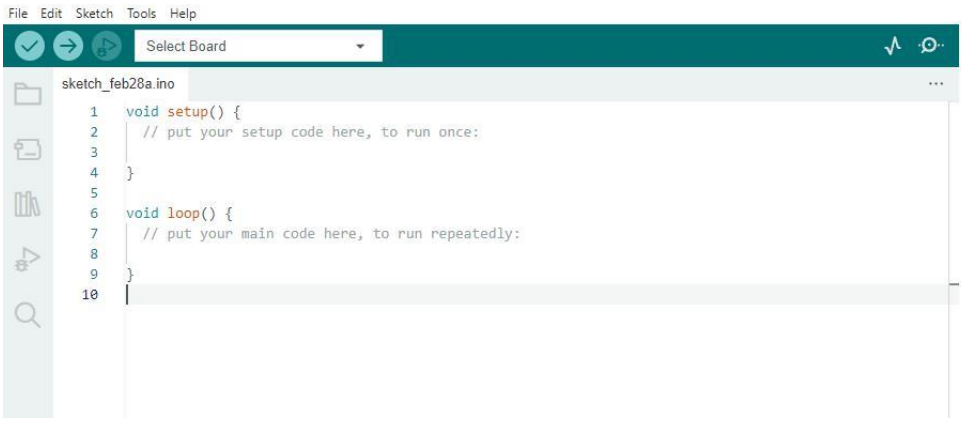
Fonte: MasterWalker, 2018.

Em projetos envolvendo IoT, existem alguns sensores mais utilizados do mercado, podendo ser eles o NB-IoT, GPS NEO-6M e o SIM800, onde cada um tem apenas a diferença ou no método de transmissão, ou na velocidade de transmissão de dados.

2.8 AMBIENTE DE DESENVOLVIMENTO DO CÓDIGO

O *IDE* (*Integrated Development Environment)* é um *software* que usa uma linguagem de programação para realizar o desenvolvimento do programa. O microcontrolador ESP32 tem seu próprio ambiente de programação, sendo o *Espressif IoT Development Framework* usando-se linguagem C ou Python. Com base na familiaridade de desenvolvimento de projetos, uma IDE, bastante usado, é da própria *Arduino*, no qual tem uma diversa gama de bibliotecas de vários tipos de microcontroladores, que acaba atendendo as demandas de projetos com os microcontroladores ESP. Esse programa é gratuito e pode baixar no site da própria *Arduino*, e consiste em uma linguagem baseada C (ARDUINO, 2019).

Figura 12: Ambiente de desenvolvimento de código do Arduino.



Fonte: Próprio Autor.

Nessa IDE tem como adicionar o tipo de microcontrolador clicando em “Ferramentas”, depois “Placa” e o modulo da placa, que no caso é o *ESP32 Dev Module*, no qual é o mais usado para desenvolvimento padrão das placas *ESP32*. Para trabalhar e uma placa específica, basta procurar a série e o módulo da placa pelo fabricante.

3 METODOLOGIA

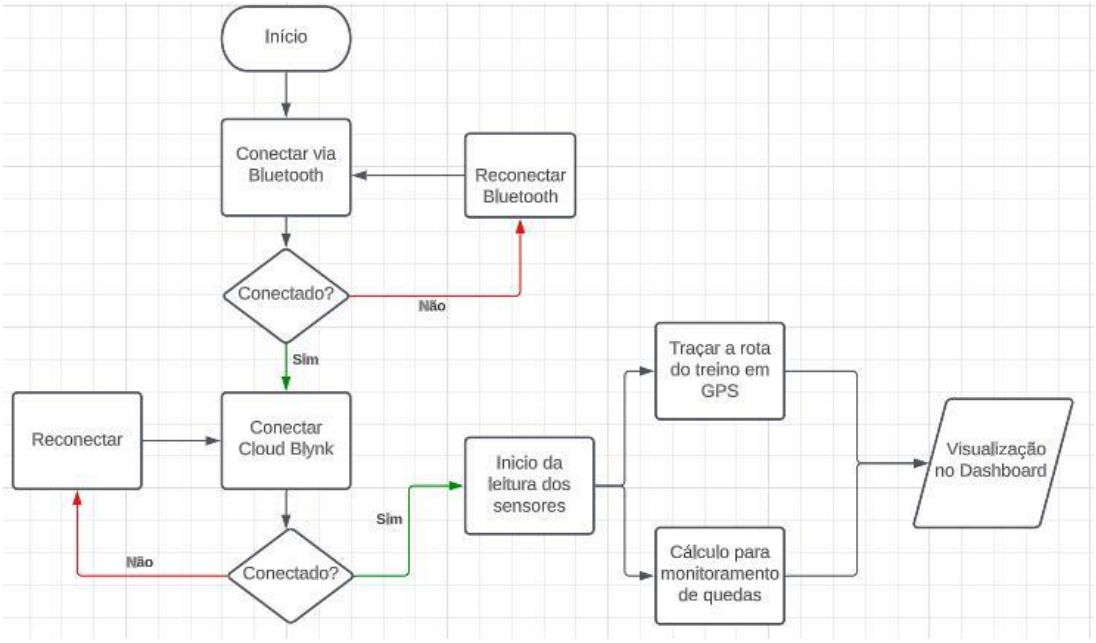
O processo da metodologia consiste em captar e analisar as características dos métodos, das capacidades, eficiência, limitações e as consequências da aplicação do processo (GIL, 2002). A metodologia deste trabalho consiste no método do hipotético-dedutivo a fim de comprovar o objetivo do projeto proposto.

O protótipo desenvolvido pretende de validar a aplicação da tecnologia da IoT com a construção de uma bicicleta inteligente, monitorando sua posição e quedas durante o percurso ciclista. Para isso, buscou-se realizar o levantamento dos dados focando prioritariamente em artigos, coletando-os em desenvolvimento de sistemas embarcados aplicado às tecnologias que envolvem IoT, usando o microcontrolador ESP32, focando em monitoramento dos dados em tempo real. Também buscou-se estudar e projetar um sistema para visualização dos dados coletados pelos sistemas embarcados, usando uma plataforma *mobile* ou *desktop*.

A conectividade Bluetooth foi pesquisada para ser utilizado pelo Blynk, como mais uma funcionalidade ao projeto. O *framework* foi estudado nesses artigos para ser utilizado como ferramenta de programação, onde o Arduino IDE atendia todos os requisitos de prototipação do programa para o desenvolvimento do projeto.

3.1 O FLUXOGRAMA DO PROJETO

Figura 13: Fluxograma do projeto.



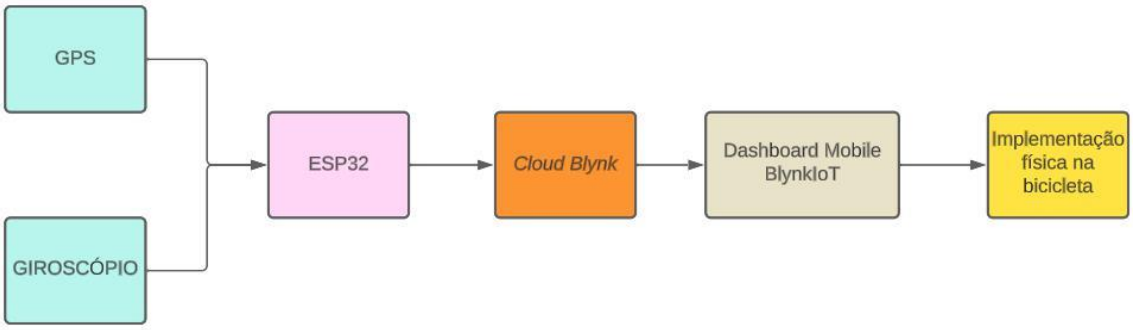
Fonte: Próprio Autor.

Este fluxograma representa o processo do teste com o software embarcado, onde pelo aplicativo o ciclista iniciará o treino, conectando via *bluetooth* a transmissão dos dados dos sensores para o armazenamento na nuvem. Para a primeira parte do trabalho, foram analisadas as variáveis de entrada tendo como objeto o ciclista, buscando-se os sensores que coletam os do posicionamento do ciclista e da rotação de um dos eixos da bicicleta, para prever a sua queda.

Realizado as coletas, projetou-se um modelo de projeto de comunicação sem fio que se dá com o ESP32 recebendo os dados provenientes dos sensores usando a tecnologia de transmissão *Bluetooth*. Para a transmissão dos dados em tempo real em uma rede local, foi proposto usar o microcontrolador para receber as informações dos sensores, como posicionamento e detecção de quedas, publicando esses dados para o servidor na nuvem, usando o *Cloud Blynk*. O servidor de rede local fornecerá dados do ciclista no seu percurso em um *dashboard*, para ser a fonte de visualização para o usuário. Para isso, o aplicativo Blynk 2.0 servirá como esse propósito para interação com usuário, mostrando os dados em tempo real e gráficos para futuras análises.

E a última etapa é a implementação do protótipo na bicicleta, organizando os materiais a serem usados, a posição dos sensores e o local de teste. Foi criado um diagrama em blocos do protótipo, resumindo cada etapa do seu desenvolvimento.

Figura 14: Arquitetura geral do projeto.



Fonte: Próprio Autor.

3.3 MATERIAIS

- Um *protoboard*.
- Fios de cabos e *jumpers*.
- 1 resistor de 1kOhms e 2 resistores de 4.7kOhms.
- 1 sensor módulo acelerômetro/giroscópio MPU-6050.
- 1 sensor módulo GPS NEO-6M com antena.

- 1 microcontrolador módulo ESP32-WROOM.
- Uma bateria recarregável *power bank*.
- Cabo USB.
- API Blynk.

3.4 O DIAGRAMA ESQUEMÁTICO DO CIRCUITO

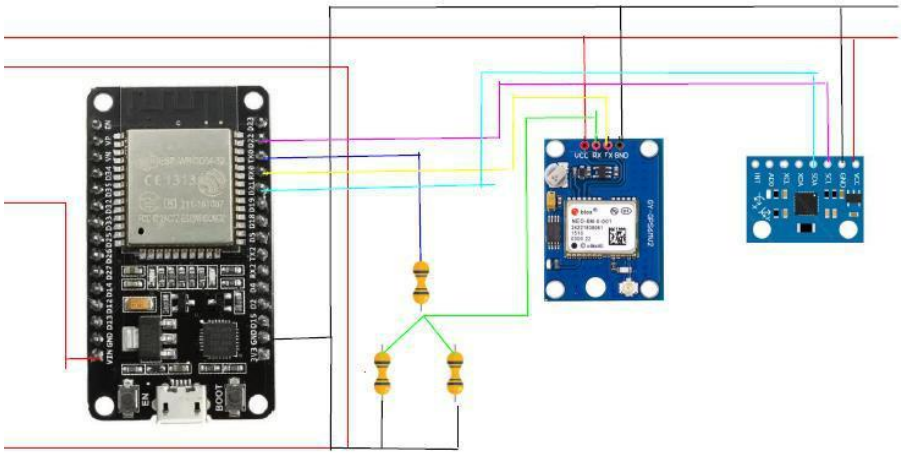
O modelo do esquemático do circuito foi feito pelo programa *Google Drawings*, uma plataforma da *Google* livre para criação de desenhos ou esquemas para projetos. A tabela mostra a pinagem dos sensores ligando ao microcontrolador ESP32, que serve como a passo-passo da montagem, indicando qual porta do sensor é conectado em tal porta do microcontrolador. No caso do módulo GPS, o sinal RX vai para divisor de tensão para limitar a saída da corrente do pino TX do ESP32.

Tabela 2 – Configurações da pinagem dos sensores com microcontroladores ESP32.

MPU-6050 -→ ESP32	GPS NEO-6M -→ ESP32
VCC -→ VIN	VCC -→ VIN
GND -→ GND	GND --> GND
SCL --> D22	TX --> RX0
SDA --> D11	RX --> DIV --> TX0

Fonte: Próprio Autor.

Figura 15: Esquemático do circuito do projeto.



Fonte: Próprio Autor.

4 IMPLEMENTAÇÃO

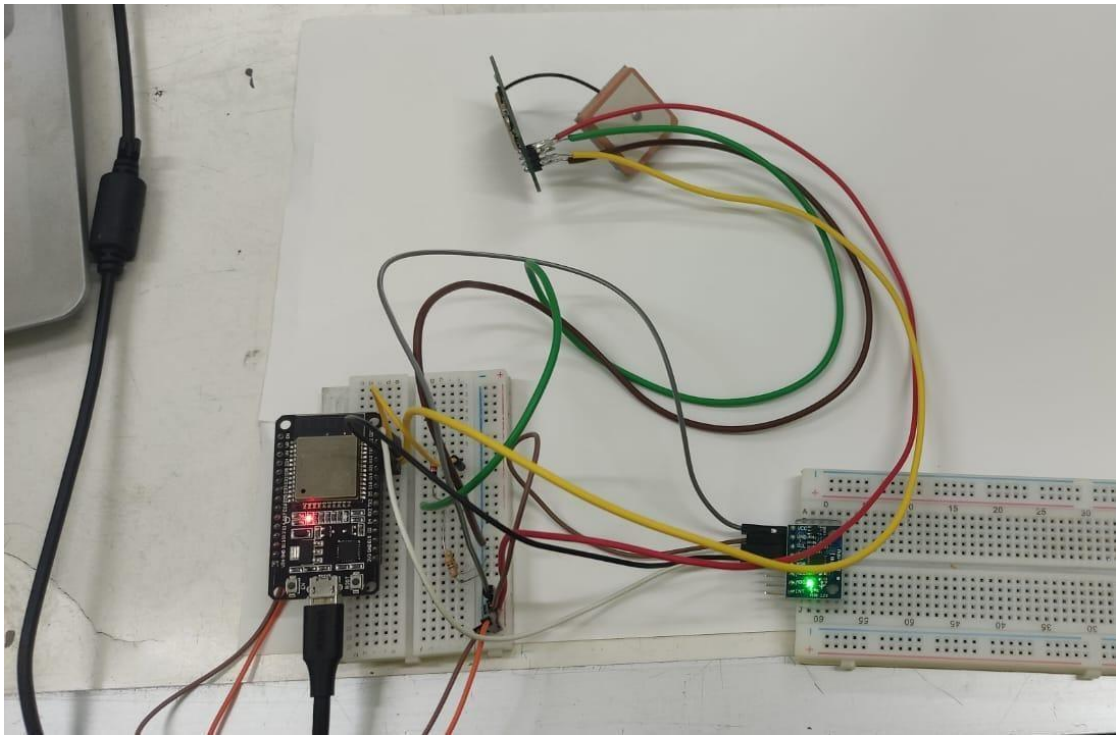
Nesta sessão serão descritos de uma forma detalhada a construção do protótipo inteligente em uma bicicleta. Foi dividido em duas etapas: a primeira é a com testes na bancada, com a ligação dos sensores e seu funcionamento, desenvolvendo o seu *software* de programação e criação do *widget*, que é a tela de monitoramento dos dados usando o aplicativo *Blynk*. Segunda parte é a implementação do sistema na bicicleta, transmitindo os dados via bluetooth para a nuvem *Blynk*.

4.1 TESTES NA BANCADA

4.1.1 MONTAGEM DO SISTEMA

Foi construído o circuito com base no esquemático do projeto descrito na sessão anterior, colocando os sensores ligados ao microcontrolador com o uso do protoboard.

Figura 16: Teste na bancada com os sensores.



Fonte: Próprio Autor.

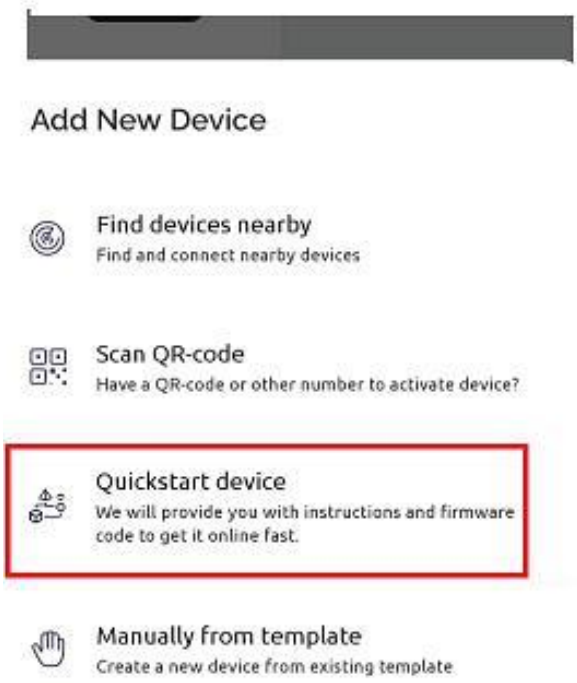
O módulo GPS, quando ligado a uma fonte de 5V, demora cerca de 1 a 5 minutos para estabelecer sua comunicação com um satélite mais próximo. Ao se conectar, o sensor vai piscar um sinal vermelho em seu LED, indicando que está pronto para a comunicação. O módulo giroscópio acende o LED verde ao ser alimentado corretamente.

4.1.2 CRIAÇÃO DO WIDGET NO APLICATIVO BLYNKIOT

Para o uso desse aplicativo, é necessário fazer uma conta para realizar os seus projetos. Ao entrar, a conta do usuário é caracterizado como básico, possibilitando apenas no máximo três projetos e dois dispositivos conectados em cada, contando com apenas as funcionalidades básicas para o uso no dashboard. Para este projeto específico, foi necessário fazer um upgrade na conta para ter funcionalidade como trajeto do mapa em tempo real e criação de alerta no aplicativo.

Ao entrar na conta do aplicativo, é necessário adicionar um novo dispositivo e criar um *modelo*.

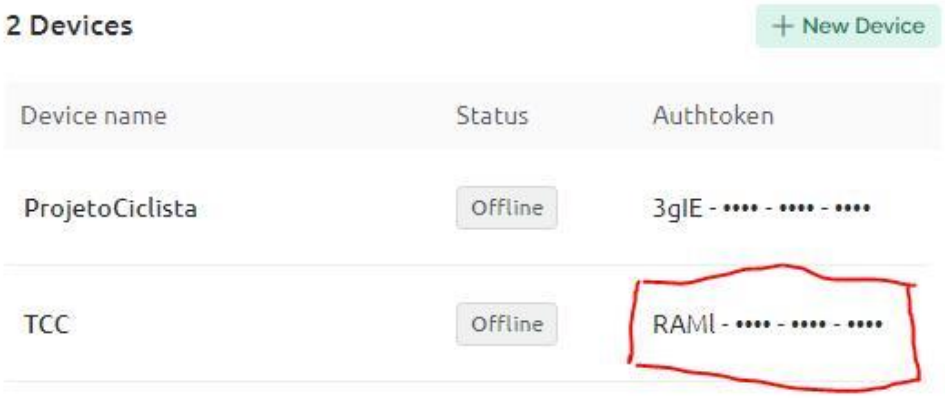
Figura 17: Criar modelo no BlynkIoT.



Fonte: Próprio Autor.

Logo depois, foi inserido nos campos o nome do projeto, o microcontrolador utilizado e o tipo de transmissão de rádio frequência. Para esse trabalho, são usados o ESP32 e o modo bluetooth para transmissão. Após ter criado o modelo, foi enviado por e-mail um código de configuração do projeto, para colocar-lo no ambiente de desenvolvimento do projeto. Esse código contém um *token* que é uma senha criptografada para ser lincado com o sistema, basta copiar e colocar no código.

Figura 18: Senha de configuração.



Fonte: Próprio Autor.

Antes de configurar o dashboard, é necessário criar as variáveis do projeto relacionando com os modelos da tela, que são chamados de *datastreams*. São usados pinos virtuais para referenciar o dado da variável no dashboard com o do programa realizado no IDE, do qual foram usados os pinos V0 a V4, contendo as variáveis de latitude, longitude, inclinação para queda, mapa e velocidade.

Figura 19: Tela de datastreams.

Fonte: Próprio Autor.

Terminado as configurações do projeto, basta criar o visual da tela do sistema supervisório colocando as funcionalidades de forma que fique melhor para visualização de dados. Para cada data, é criado uma funcionalidade, criando ao todo cinco caixas de informação.

Figura 20: Tela inicial com as funcionalidades.

Fonte: Próprio Autor.

Foi criado o modelo da tela para visualizar os dados do projeto, e o próximo passo é a configuração do *software* embarcado para leitura de sensores e envio de dados para o modelo do BlynkIoT.

4.1.3 O DESENVOLVIMENTO DO CÓDIGO NA IDE ARDUINO

No início do desenvolvimento, foram necessários instalar algumas bibliotecas na IDE Arduino para o funcionamento do projeto, trabalhando especificamente com os sensores definidos no trabalho. Foram instaladas as bibliotecas *TinyGPS++*, *AdafruitSensor*, ambos utilizados para os sensores e o *BlynkSimple32*, usada para transmitir os dados para a nuvem Blynk. A lista de bibliotecas usados no projeto pode ser puxadas nas configurações da IDE.

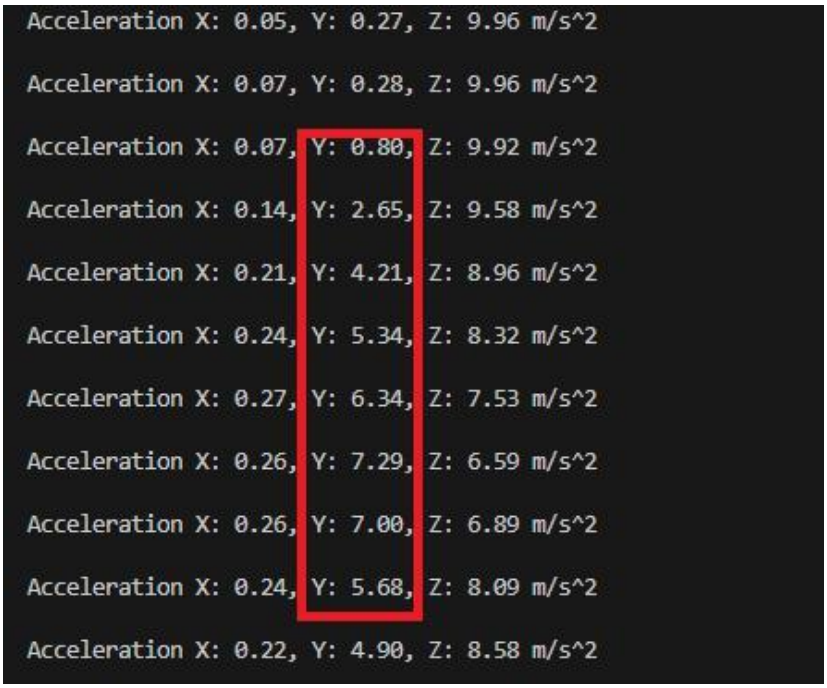
Figura 21: Bibliotecas instaladas no software.

Fonte: Próprio Autor.

Primeiramente, criou-se um código funcional para cada sensor separadamente, lendo os dados e mostrando os seus valores na saída do terminal do ambiente de desenvolvimento. Para o GPS, foram postados os valores de latitude, longitude e velocidade, sendo transmitidos a cada 1 segundo, e também segue o mesmo procedimento de teste para o sensor do giroscópio, para inclinação do ciclista.

Para determinar o eixo do sensor MPU para ser usado em caso de inclinação do cilista, foi realizado uns testes de leitura desse sensor, este preso em uma estrutura reta e móvel para movimentação de um dos seus eixos. Essa estrutura foi movimentada em direções X, Y e Z para determinar qual desses seria usado com um intuito de posicionar o sensor da bicicleta. Usando a biblioteca *AdafruitSensor*, foi feito um código para leitura de valores e pelos testes foi determinado que o eixo Y é o melhor em questão de posicionamento.

Figura 22: Leitura dos dados do sensor MPU-6050.



Fonte: Próprio Autor.

Após trabalhar com os dois sensores de forma individual, juntou os programas dos sensores. No software do programa, foram feitas funções para otimizar a programação, contendo um loop principal que visa rodar a cada 2 segundos, toda a programação transmitindo os dados para o *Blynk*. Essa transmissão se dá a uma função que tem na biblioteca *BlynkSimple32* que é o *Blynk.virtualWrite* onde tem dois parâmetros nessa função, que um é o pino virtual (que no caso pode ser V0 a V4) e o outro é a variável ligada ao pino virtual.

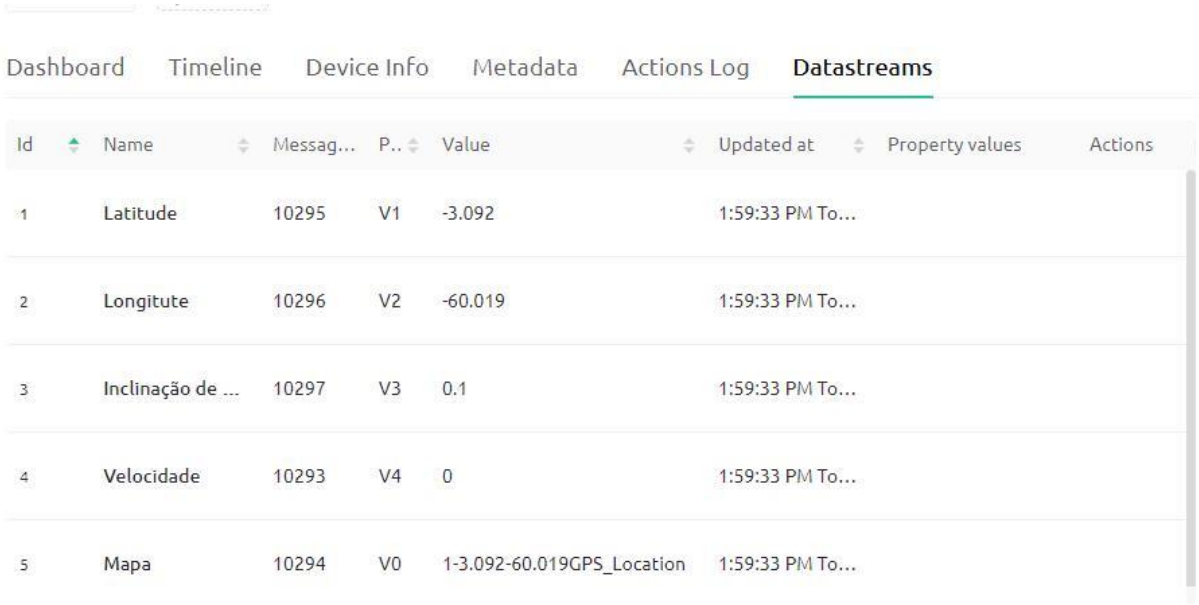
Figura 23: Funções que transmitem dados das variáveis.

```
float aceel = mpuTest();
Blynk.virtualWrite(V1, String(latitude, 6));
Blynk.virtualWrite(V2, String(longitude, 6));
Blynk.virtualWrite(V3,aceel);
Blynk.virtualWrite(V4, velocity);
```

Fonte: Próprio Autor.

Após a implementação dos dois sensores no código com as bibliotecas para transmissão de dados, é necessário compilar o programa. Se as conexões e o dispositivo ESP32 estiverem energizados, o sistema se conectará a um ponto de transmissão bluetooth e transmitir as informações para dashboard. A figura abaixo mostra um painel do *datastreams* com os dados das variáveis, atualizando a cada segundo.

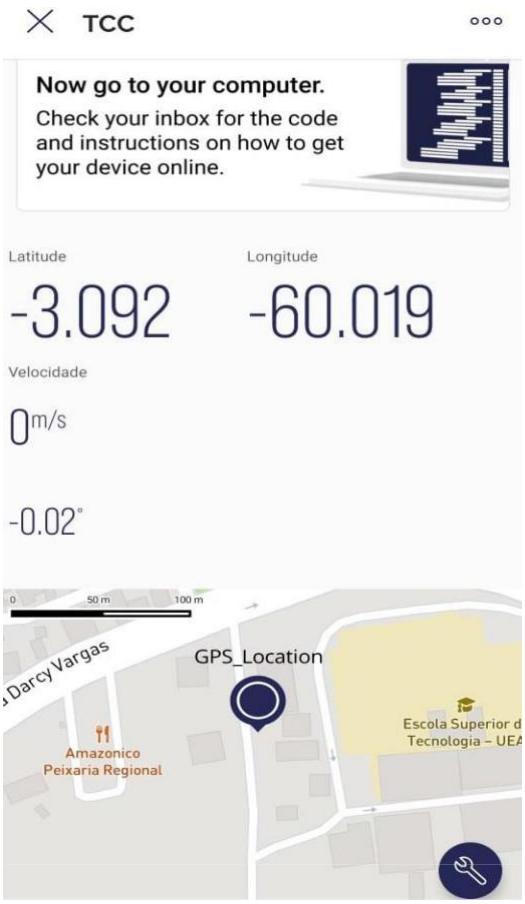
Figura 24: Dados sendo transmitidos no sistema.



Fonte: Próprio Autor.

Observando que os dados estão sendo transmitidos para o aplicativo, agora é possível visualizar os dados em tempo real com as interações visuais das funcionalidades. A figura abaixo mostra a tela que o usuário vê durante o percurso do ciclista.

Figura 25: Dashboard de teste.



Fonte: Próprio Autor.

Através deste teste na bancada, obtendo o funcionamento da leitura dos sensores e sua transmissão de dados para o aplicativo com o uso da conexão bluetooth, foi possível realizar, com base nesse ensaio, o experimento com a bicicleta no ambiente ao ar livre.

4.2 TESTES NO AMBIENTE OUTDOOR

Para os testes no ambiente externo, o local de teste foi na Universidade Federal do Amazonas, no campus da faculdade de Fisioterapia. Definiram-se modelos de teste para algumas diferentes situações, dos quais foram: movimento em baixa velocidade sem inclinação, movimentos de velocidade de média e alta intensidade, e simulações de queda da bicicleta em movimento leve.

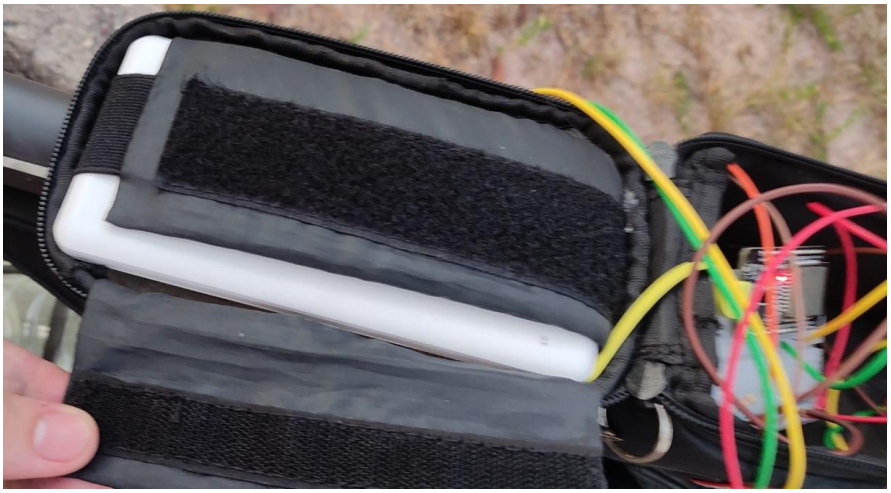
Para implementação, foi usada uma bolsa para colocar o circuito eletrônico e o *power bank* no lugar, para a alimentação do circuito. Colocou-se o sensor MPU-6050 na frente da bicicleta e o módulo GPS. Para execução do programa, foi compilado o código definitivo do projeto, pressionando o botão *boot* para ser carregado.

Depois disso, abriu-se o aplicativo do BlynkIoT com *template* TCC construído conforme descrito na sessão anterior.

Figura 26: Implementação do protótipo na bicicleta.

Fonte: Próprio Autor.

Figura 27: Implementação da alimentação do circuito.



Fonte: Próprio Autor.

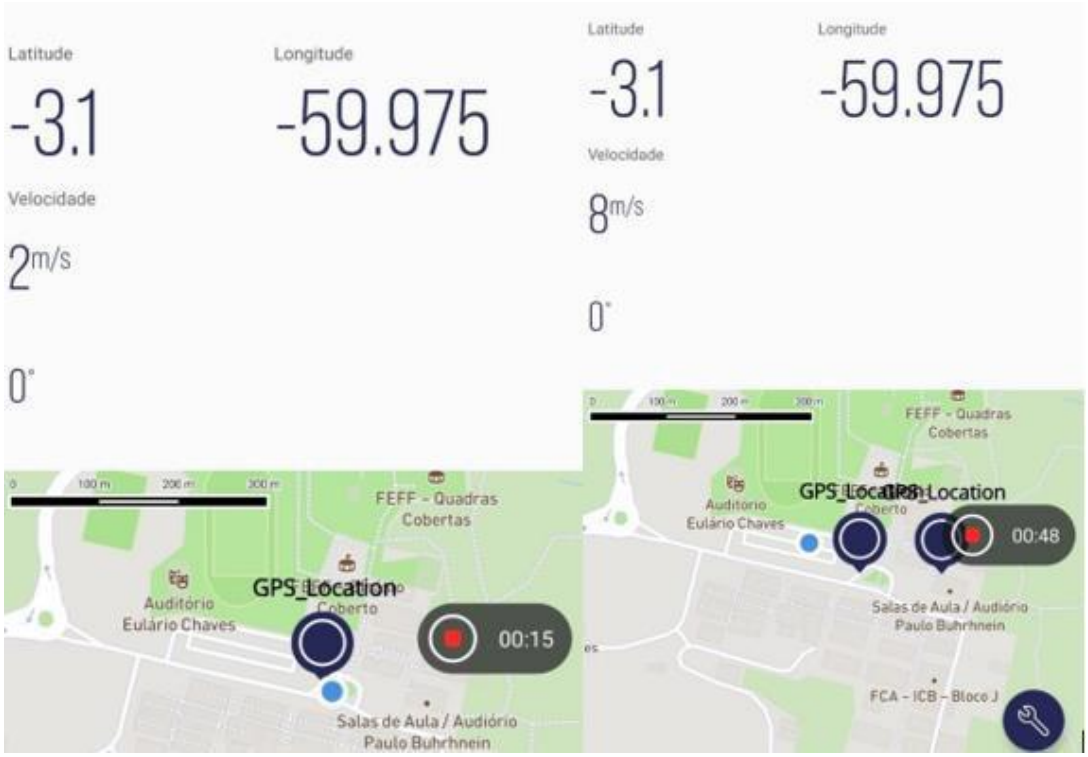
Figura 28: O sensor GPS na bicicleta.

Fonte: Próprio Autor

5 ANÁLISE DE RESULTADOS E DISCUSSÕES

A análise da construção do protótipo de bicicleta inteligente se dá ao analisar os dados em tempo real com os parâmetros propostos no projeto, que se dá através dos dados de cada modelo de teste no ambiente aberto, com dados do treino de velocidade leve, velocidade moderada e testes de inclinação da bicicleta.

Figura 29: Testes do modelo com velocidade leve (a esqueda) e velocidade moderada.



Fonte: Próprio Autor.

Para a aplicação, pode-se ver o sistema em pleno funcionamento, trazendo os dados da localização e da sua velocidade em tempo real. Podemos ver que na figura acima, os dados do lado esquerdo mostram o resultado de uma velocidade baixa, que nesse caso é entre um a três metros por segundo, já o do lado direito traz uma velocidade moderada para intensa, chegando entre 7 à 10 metros por segundo. Podemos ver também um ponto azul no mapa, indicando o lugar atual do ciclista.

Para o teste de inclinação com a bicicleta, foram realizados alguns treinos em movimento leve, para não ocasionar acidentes para o ciclista do treino realizado em questão. Podemos ver que, em alguns treinos, ocorreu uma leve variação de inclinação e outros uma grande variação de inclinação, mas ocorrendo de uma forma não padronizada, mas com alguns casos o sensor giroscópio detectava essa inclinação.

Figura 30: Teste de velocidade moderada com inclinação na bicicleta.

Fonte: Próprio Autor.

Para uma análise geral de modo a demonstrar os objetivos atingidos no trabalho, o resultado foi bastante satisfatório, podendo visualizar os dados em tempo real da localização do ciclista no mapa, a sua velocidade durante o percurso e inclinação, com o auxílio de uma plataforma de fácil acesso e com um bom *design* para o usuário, mesmo com limitações de recursos com o plano básico. Para projetos futuros relacionado o desenvolvimento de um projeto melhor de *SmartBike* propõe-se:

- Melhorar a precisão do sensor de inclinação, podendo ser ajustável no software ou posicionamento.
- Melhoria do local físico dos sensores, como uma construção de um *case* em impressão 3D.
- Possibilidade de ampliar a transmissão de dados a longa rede, usando o sistema LoRa.
- Criação de uma placa prototipada para otimização do circuito eletrônico.

CONCLUSÃO

Este projeto propôs a elaboração de um protótipo de uma bicicleta inteligente capaz de monitorar os dados de posição geográfica e inclinação do ciclista em um percurso, criando-se um sistema *mobile* para visualização de dados em um *dashboard*. Aplicaram-se os conhecimentos adquiridos durantes as pesquisas executadas, juntamente com o conteúdo ministrado na graduação.

Apresentaram-se no conteúdo deste trabalho, o microcontrolador ESP32, bem com suas características e se percebe quão robusta é esta tecnologia, que apesar de nova, vem ganhando mercado mundo afora com soluções em IoT. Construiu-se o protótipo com os sensores de localização em tempo real, que é o GPS, e o sensor giroscópio para medir a variação de aceleração em um dos seus eixos. Implementou-se o código-fonte na IDE do Arduino e foi feito a compilação e envio do código. Pelo sistema BlynkIoT consegue-se ver a conexão com Bluetooth e com o servidor são bem-sucedidas. Começa então o envio de dados por meio dos coletados pelos sensores, e é visto na aplicação.

Pode-se verificar o funcionamento da comunicação entre o servidor Blynk e os sensores, onde a informação na nuvem esteve presente na aplicação, ao usuário de saída. A resposta ao usuário foi relativamente satisfatória, onde o aplicativo utilizado suportou parcialmente o recebimento de informações do servidor e o seu *design* e facilidade em criação de modelos de *dashboard*, proporciona uma melhor experiência para o ciclista.

Para desenvolvimento deste trabalho futuramente, propõe-se melhorar a precisão do sensor de inclinação, como também criar um espaço físico na bicicleta para o encaixo do circuito eletrônico. Propõe-se também como melhorias aplicando uma tecnologia de longa distância, como LoRa, para aplicação de desenvolvimento à longa distâncias.

REFERÊNCIAS BIBLIOGRÁFICAS

CAROLINA, L. "Análise comparativa de plataformas baseadas em Cloud para IoT e o desenvolvimento de aplicações.", 2020.

DE ARAUJO, J. H., MOREIRA, E. M. M., DE FREITAS, C. F., *et al.* "Smart Cities: um estudo prospectivo sobre Internet das Coisas (IoT) aplicada ao setor de mobilidade urbana", **Cadernos de Prospecção**, v. 13, n. 1, p. 138, 2020. DOI: 10.9771/cp.v13i1.32691. .

DOS, A. S., LEANDRO, S., MATHEUS, A., *et al.* **Tecnologias Emergentes Em Iot: Rssf, Rtls, Rfid Conceitos E Aplicações Para Cidades Inteligentes E Indústria 4.0 Autores**, 2020. Disponível em: www.ipt.br.

GIL. **Antônio Carlos Gil Projetos de pesquisa**, 2002.

HOLLANDS, R. G. "Will the real smart city please stand up? Intelligent, progressive or entrepreneurial?", **City**, v. 12, n. 3, p. 303–320, 2008. DOI: 10.1080/13604810802479126. .

ISMAIL, A. A., HAMZA, H. S., KOTB, A. M. "Performance Evaluation of Open Source IoT Platforms", **2018 IEEE Global Conference on Internet of Things, GCIoT 2018**, p. 1–5, 2019. DOI: 10.1109/GCIoT.2018.8620130. .

KEVIN ASTHON. "That ' Internet of Things ' Thing", **RFID Journal**, p. 4986, 2010. Disponível em: http://www.rfidjournal.com/article/print/4986.

KIM, S. H., MOON, H. J. "Case study of an advanced integrated comfort control algorithm with cooling, ventilation, and humidification systems based on occupancy status", **Building and Environment**, v. 133, p. 246–264, 2018. DOI: 10.1016/j.buildenv.2017.12.010. Disponível em: https://doi.org/10.1016/j.buildenv.2017.12.010.

LEITE, J. R. E., MARTINS, P. S., URSINI, E. L. "A INTERNET das COISAS (IoT) : Tecnologias e Aplicações", **Brazilian Technology Symposium**, p. 6, 2017. .

MAIO, A. F., AFONSO, J. A. "Wireless cycling posture monitoring based on smartphones and bluetooth low energy", **Lecture Notes in Engineering and Computer Science**, v. 2217, p. 653–657, 2015. .

MENEGOTTO, J. L. "Sensoriamento Da Edificação: Um Sistema De Localização Baseado Em Beacons Ble", p. 264–274, 2015. DOI: 10.5151/engpro-tic2015-024. .

MENEGUZZI, L., CENDRON, M. M., FRITZEN, R., *et al.* "Utilização de giroscópio e acelerômetro para identificação de movimentação em ambientes tridimensionais", 2016. .

MUHAMAD, W. N. W., RAZALI, S. A. Bin, WAHAB, N. A., *et al.* "Smart Bike Monitoring System for Cyclist via Internet of Things (IoT)", **2020 IEEE 5th International Symposium on Telecommunication Technologies, ISTT 2020 - Proceedings**, p. 168–173, 2020. DOI: 10.1109/ISTT50966.2020.9279377. .

MUNDADA, M., MUKKAMALA, R. R. "Smart cities for sustainability-an analytical perspective", **Proceedings of the World Conference on Smart Trends in Systems, Security and Sustainability, WS4 2020**, p. 770–775, 2020. DOI: 10.1109/WorldS450073.2020.9210379. .

NEIROTTI, P., DE MARCO, A., CAGLIANO, A. C., *et al.* "Current trends in smart city initiatives: Some stylised facts", **Cities**, v. 38, p. 25–36, 2014. DOI: 10.1016/j.cities.2013.12.010. Disponível em: http://dx.doi.org/10.1016/j.cities.2013.12.010.

NORTH, A., FUX, S., BOUABDALLAH, S. "Development of a planar low cost Inertial Measurement Unit for UAVs and MAVs", **Spring**, n. January, p. 91, 2008. .

OLIVEIRA, F., NERY, D., COSTA, D. G., *et al.* "A survey of technologies and recent developments for sustainable smart cycling", **Sustainability (Switzerland)**, v. 13, n. 6, 2021. DOI: 10.3390/su13063422. .

PRAVALIKA, V., PRASAD, C. R. "Internet of Things Based Home Monitoring and Device Control Using Esp32", n. 1, p. 58–62, 2019. .

RIOS, A. D. A., ANDRADE, D. A. De, GOMES, L. C., *et al.* "INTERNET DAS COISAS", p. 1–6, 2018

SALES DE AZEVEDO, R., MARQUES DE CASTRO, V., DE ANDRADE BARROS, V., *et al.* **FaSCi-Tech Smart Bike: Plataforma Aberta para Monitoramento e Gestão de Transporte Urbano baseado em Bicicletas Elétricas e IoT**, 2020.

SANTOS, B. P., SILVA, L. A. M., CELES, C. S. F. S., *et al.* "Internet das Coisas: da Teoria à Prática", **XXXIV Simp´osio Brasileiro de Redes de Computadores e Sistemas Distribu´ıdos (SBRC 2016)**, p. 1–50, 2016. .

SYSTEMS, E. **ESP32-WROOM-32 Datasheet**., 2023.

WEGNER, P. "Global IoT market size to grow 19% in 2023 - IoT shows resilience despite economic downturn", **IOT Analytics**, v. 49, n. 0, 2023. Disponível em: https://iot-analytics.com/iot-market-size/.

APÊNDICE – CÓDIGO FONTE

```
// Matheus Coelho Cardoso - DESENVOLVIMENTO UM PROTÓTIPO DE
BICICLETA INTELIGENTE
// E MONITORADA APLICANDO A TECNOLOGIA IOT

// INCLUSÃO DE BIBLIOTECAS E DEFINIÇÃO DE CERTAS VARIAVEIS
#include <Arduino.h>
#include <TinyGPSPlus.h>
#include <TinyGPS++.h>
#include <Adafruit_MPU6050.h>
#include <Adafruit_Sensor.h>
#define BLYNK_TEMPLATE_ID "TMPL2W0PzJynz"
#define BLYNK_TEMPLATE_NAME "Quickstart Template"
#define BLYNK_AUTH_TOKEN "RAMlxKLLEvH6hdy-Uv9zRSQd5fXm3O37"
#include <BlynkSimpleEsp32.h>
#include <SoftwareSerial.h>
#include <WiFi.h>
#define BLYNK_PRINT Serial
// CONFIGURAÇÕES DO SENSOR GPS
static const int RXPin = 3, TXPin = 1;
static const uint32_t GPSBaud = 9600;
TinyGPSPlus gps;
WidgetMap myMap(V0); // MOSTRAR O MAPA COM O PINO VIRTUAL
V0 SoftwareSerial mygps(RXPin, TXPin);
BlynkTimer timer;

// VARIAVEIS PARA O SISTEMA
GPS float latitude;
float longitude;
float velocity;
float sats;
String bearing;

// CONFIGURAÇÃO PADRÃO DO BLYNKIOT
char auth[] = BLYNK_AUTH_TOKEN;            //Blynk Authentication Token

unsigned int move_index = 1;
// VARIÁVEL DO SENSOR GIROSCÓPIO
Adafruit_MPU6050 mpu;

// FUNÇÃO DO GIROSCOPIO QUE RETORNA A INCLINAÇÃO NO EIXO
Y float mpuTest(){
  sensors_event_t a, g, temp;
  mpu.getEvent(&a, &g, &temp);
  return a.acceleration.y;
}

// FUNÇÃO VERIFICAR LIGAÇÃO DO GPS
void checkGPS()
{
  if (gps.charsProcessed() < 10)
  {
    Serial.println(F("No GPS detected: check
    wiring.")); Blynk.virtualWrite(V3, "GPS ERROR");
  }
}


// FUNÇÃO LEITURA DOS SENSORES
```

```
void displayInfo()
{

  if (gps.location.isValid() )
  {

    latitude = (gps.location.lat());
    longitude = (gps.location.lng());
    velocity = gps.speed.mps();
    bearing = TinyGPSPlus::cardinal(gps.course.value());
    float aceel = mpuTest();
    Blynk.virtualWrite(V1, String(latitude, 6));
    Blynk.virtualWrite(V2, String(longitude, 6));
    Blynk.virtualWrite(V3,aceel);
    Blynk.virtualWrite(V4, velocity);
    myMap.location(move_index, latitude, longitude, "GPS_Location");

  }
  Serial.println();
}

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

  while (!Serial)
    delay(10); // will pause Zero, Leonardo, etc until serial console
  opens Serial.println("Adafruit MPU6050 test!");
  // Try to initialize!
  if (!mpu.begin()) {
    Serial.println("Failed to find MPU6050
    chip"); while (1) {
      delay(10);
    }
  }
  // CALIBRAÇÃO DO ACELERÔMETRO DO MPU6050
  Serial.println("MPU6050 Found!");
  mpu.setAccelerometerRange(MPU6050_RANGE_8_G);
  Serial.print("Accelerometer range set to: ");
  switch (mpu.getAccelerometerRange()) { case
  MPU6050_RANGE_2_G:
    Serial.println("+-2G");
    break;
  case MPU6050_RANGE_4_G:
    Serial.println("+-4G");
    break;
  case MPU6050_RANGE_8_G:
    Serial.println("+-8G");
    break;
  case MPU6050_RANGE_16_G:
    Serial.println("+-16G");
    break;
  }
  // CALIBRAÇÃO DA ROTAÇÃO DO MPU6050
  mpu.setGyroRange(MPU6050_RANGE_500_DEG);
  Serial.print("Gyro range set to: ");
  switch (mpu.getGyroRange()) { case
  MPU6050_RANGE_250_DEG:
    Serial.println("+- 250 deg/s");
    break;
  case MPU6050_RANGE_500_DEG:
```

```
    Serial.println("+- 500 deg/s");
    break;
  case MPU6050_RANGE_1000_DEG:
    Serial.println("+- 1000 deg/s");
    break;
  case MPU6050_RANGE_2000_DEG:
    Serial.println("+- 2000 deg/s");
    break;
  }
  // CALIBRAÇÃO DO FILTRO DE ATENUAÇÃO DO MPU6050
  mpu.setFilterBandwidth(MPU6050_BAND_5_HZ);
  Serial.print("Filter bandwidth set to: "); switch
  (mpu.getFilterBandwidth()) {
  case MPU6050_BAND_260_HZ:
    Serial.println("260
    Hz"); break;
  case MPU6050_BAND_184_HZ:
    Serial.println("184
    Hz"); break;
  case MPU6050_BAND_94_HZ:
    Serial.println("94
    Hz"); break;
  case MPU6050_BAND_44_HZ:
    Serial.println("44
    Hz"); break;
  case MPU6050_BAND_21_HZ:
    Serial.println("21
    Hz"); break;
  case MPU6050_BAND_10_HZ:
    Serial.println("10
    Hz"); break;
  case MPU6050_BAND_5_HZ:
    Serial.println("5
    Hz"); break;
  }
  mygps.begin(GPSBaud);
  Blynk.begin(auth, ssid, pass);
  timer.setInterval(2000L, checkGPS); // CHECAR A CADA 2 SEGUNDOS NOVOS
DADOS DO MODULO GPS
}

void loop()
{
  // AQUI É A FUNÇAO PRINCIPAL

  while (mygps.available() > 0) // SE A ANTENA DO GPS FOR DETECTADA
  {
    if(gps.encode(mygps.read()))
      displayInfo();
  }
  Blynk.run(); // RODAR AS CONFIGURAÇÕS DO
  BLYNK timer.run();

}
```