# Introdução

A Internet das Coisas (Internet Of Things – IOT) tem como essência a interligação de diferentes tecnologias de rede agregadas a objetos, coisas, como smartphones, sensores pessoais, braços robóticos na automação, dentre outros. Essa interligação se torna uma única malha de dispositivos conectados. Mesmo que inicialmente tivessem propósitos únicos, ao se interligar passam a ter uma nova gama mais extensa de funcionalidades que se complementam. O uso de braços robóticos está presente na realidade do IOT na escala industrial relacionado a automação, porém não está limitado a ela. Seu uso se expande a diversas funções chegando até mesmo a ser utilizado nas sondas espaciais Perseverance, Spirit, Opportunity e Curiosity. O controle desses braços robóticos pode ser feito de diversas formas, sendo uma delas o seu controle através de uma interface. Com o aumento da popularidade de smartphones e seus aplicativos, a capacidade de controlar dispositivos à distância através dos dispositivos mobile se torna mais almejado. Para o desenvolvimento de uma interface que permita essa função, busca-se um bom aproveitamento de código para as diversas plataformas disponíveis atualmente, com destaque para os dispositivos iOS e Android, sem que suas funcionalidades sejam consideravelmente comprometidas.

# Objetivos

Este projeto tem como objetivos principais a montagem de um braço mecânico com peças de acrílico e a programação de um app que permita utilizar um smartphone Android para controlar, através de tecnologias de comunicação remota, o braço mecânico montado.

# Metodologia

As metodologias utilizadas foram a programação no Arduino IDE que é um ambiente de desenvolvimento integrado que disponibiliza um editor de texto para a escrita do código, um console de texto, barra de ferramentas com diversas funções e permite a conexão com vários microcontroladores para o envio do programa desenvolvido e se comunicar com o dispositivo e também no Visual Studio Code que é um editor de código simplificado com suporte para operações de desenvolvimento como debug e controle de versões. O VS Code foi o ambiente utilizado para o desenvolvimento de aplicativos mobile. Posteriormente foi feita a montagem do braço robótico utilizando uma chave Philips para montar um novo braço robótico com 5 graus de liberdade, sendo este novo braço feito de acrílico. Em seguida é feita a junção dos códigos com o braço, assim permitindo a transmissão de dados. Posteriormente com a montagem do braço robótico, a transmissão de dados, já funcional, foi utilizada para movimentar o braço robótico.

# Resultados e Discussão

Como resultado vê-se de forma sucinta a montagem do braço robótico e do aprendizado sobre tecnologias de comunicação remota adquirido para o desenvolvimento do app. Após a pesquisa, foram encontradas diversas tecnologias de comunicação remota, muitas delas utilizadas no contexto do IOT. Abaixo estão listadas algumas das que foram pesquisadas e o parecer quanto a quão apropriado é o seu uso no atual projeto, não descartando a possibilidade e seu uso em outros projetos envolvendo o IOT: RFid (Radio-Frequency Identification – Identificação por Radiofrequência) – Não apropriado. Protocolo CAN – Viável, porém não apropriado. ESP32 LoRaWan – Viável, porém não apropriado. ZigBee – Viável, porém não apropriado. Wi-Fi (Wireless Fidelity) – Apropriado: Também chamado de WLAN (Wireless Local Area Network – Rede Sem Fio de Área Local) é uma tecnologia presente na maioria dos Smartphones atuais, geralmente utilizado para se conectar à internet. Apresenta diversos módulos para o uso com o Arduíno. Estão disponíveis vários microcontroladores, como o ESP32 e o NodeMCU ESP8266, que já vêm com o módulo WiFi embutido. Como o NodeMCU ESP8266 está disponível para uso sem a necessidade de aquisição, ele foi escolhido para ser testado. Bluetooth (Classic) – Apropriado: O Bluetooth, também chamado de WPAN (Wireless Personal Area Network – Rede Sem Fio de Área Pessoal), assim como o WiFi, também é uma tecnologia presente na maioria dos Smartphones atuais, geralmente utilizado para se conectar dispositivos sem fio, principalmente fones de ouvido, teclados e mouses. Apresenta diversos módulos para o uso com o Arduíno, podendo ser citados os módulos HC-05 (utilizado nesse projeto) e o HC-06. Essa tecnologia também está disponível no módulo ESP32 citado anteriormente que embora não disponível, pode ser adquirido para projetos futuros. Com isso entende-se que as tecnologias apropriadas e escolhidas para o projeto atualmente são Wi-Fi e Bluetooth. Foram testados os microcontroladores NodeMCU Amica com ESP8266 e NodeMCU Lolin com ESP8266 para a conexão Wi-Fi e foi utilizado o módulo HC-05 para a conexão Bluetooth com o Arduíno e também com ambos os microcontroladores Amica e Lolin. Também foram feitas pesquisas quanto às plataformas microcontroladas e foram encontradas diversas opções como: Placas que têm o microchip ESP8266: ESP01, ESP12E, ESP12F, ESP201, NODEMCU-ESP12 e CP2102 e placas que têm o chip ESP32: ESP32-WROVER, ESP32-WROOM-32U, ESP32S-CP2102 e NODEMCU V3 WIFI 802.11. De início optou-se por manter o desenvolvimento com o Arduíno, visto que a quantidade de falhas poderia ser reduzida devido à familiaridade, e por isso o desenvolvimento pode ser mais dinâmico. Assim que o desenvolvimento fosse concluído, novos testes poderiam ser realizados com as outras placas para analisar se vale ou não a pena a mudança. As placas disponíveis para testes no laboratório e que foram definidas como viáveis para o projeto através de pesquisa temos o NodeMCU Amica e NodeMCU Lolin ambos com o microchip ESP8266 com acesso à conexão Wi-Fi. Com o objetivo final de controlar o braço robótico através de tecnologias de comunicação remota, foram desenvolvidos programas no Arduino IDE para se alcançar este resultado. Sendo o primeiro a representação dos dados no Monitor e no Plotter Serial que são duas formas de análise dos valores recebidos. Um deles é o uso do monitor Serial que mostra o valor dos ângulos de cada um dos servomotores pontualmente e separadamente. Como segunda temos o Plotter serial com o qual podemos ver em forma de gráficos a variação de angulação dos servomotores que são valores recebidos pelo Arduino. Outro código é desenvolvido para testar o recebimento de dados via Bluetooth que acaba por ser bem simples, sendo necessário apenas que na função *setup ()* ele inicie a comunicação serial na velocidade 9600 bauds, ou seja, 9600 bits por segundo e que na função *loop ()* se o valor estiver disponível na entrada serial, ele irá imprimir o valor lido. Já o código para recebimento de dados via Wi-Fi apresenta uma complexidade muito superior à do envio de dados via Bluetooth. Sendo comparado ao Arduíno, o NodeMCU Amica além de demorar mais tempo para ser configurado, também demora mais para fazer o upload dos códigos para o microcontrolador. O código também acaba não sendo muito legível pois há a junção de linguagens de programação, o que pode gerar certa confusão ao se misturar comandos em C com as tags do HTML. Mas mesmo com esses contratempos, ainda assim é possível obter a transmissão dos dados. O código permite a transmissão de dados após a mudança de posição de um slider. Entretanto, a informação só é enviada após soltar o slider na nova posição, o que reduz a quantidade de dados que é transmitida. Diante da vasta gama de linguagens de programação disponíveis foi feita uma pesquisa sobre algumas linguagem com foco em desenvolvimento de apps e os seus benefícios, sendo classificadas quanto à sua aplicabilidade no atual projeto: Swift – Não apropriado, Java – Viável e apropriado, Kotlin – Viável e apropriado: Possui as mesmas desvantagens do Java, Ionic e Flutter – Viáveis e apropriados, JavaScript – Viável, apropriado e utilizado: É a linguagem utilizada no back-end do app. É a linguagem que em conjunto com o framework React Native, torna o app funcional. React Native –Viável, apropriado e utilizado: Permite que um app utilizando da linguagem JavaScript seja convertido para a linguagem nativa tanto do iOS quanto do Android. Também podendo ser considerado que a lógica presente no desenvolvimento se assemelha (e utiliza de conceitos) de linguagens mais consolidadas, sendo elas HTML e CSS (mais utilizadas em desenvolvimento web) e também de React (uma biblioteca de JavaScript para desenvolvimento de interfaces) e JavaScript. Além disso pode ser desenvolvido utilizando o Expo. Que torna possível programar utilizando React Native e quase que instantaneamente ver o app resultante em um Smartphone conectado à mesma rede Wi-Fi que tenha o app do Expo. Mesmo com a velocidade o Expo tem suas limitações: ao utilizá-lo não é possível importar bibliotecas (conjunto de código que permitem o acesso a novas funcionalidades) foram necessárias para o projeto. Também tem seu uso consolidado em diversos apps conhecidos como Facebook, Instagram, Discord, dentre outros. Com isso fica decidido o uso do React Native em conjunto com o JavaScript para o desenvolvimento do App do atual projeto. Para entender melhor o uso e a sintaxe da linguagem escolhida, foram desenvolvidos diversos apps, porém primeiramente foi necessário aprender sobre a linguagem que seria utilizada. Primeiro aprendendo a montar o ambiente de trabalho com os softwares necessários, sendo eles VS Code para programar o app, Android Studio para poder emular um dispositivo Android em que o desenvolvimento do app seria observado e também o Node.js que disponibiliza algumas ferramentas de desenvolvedor e também o Expo que permite uma rápida atualização através de seu app. Durante o aprendizado utilizando o Expo, foram desenvolvidos diversos apps com funcionalidades diferentes para se explorar as capacidades da linguagem e entender o seu funcionamento. No primeiro aplicativo foram testados componentes como Bloco de Texto, Imagem, Relógio e Caixa de Texto. Após diversos testes com esses componentes e variáveis, percebeu-se uma limitação presente no Expo: quando a biblioteca para o uso do Bluetooth foi adicionada, o Expo não a reconheceu, com isso, foi necessário dispensar o Expo para que o desenvolvimento continuasse. Então foi necessário mudar a abordagem, passando a conectar fisicamente o smartphone que receberia o aplicativo ao computador onde as programações estavam sendo desenvolvidas e assim, utilizando apenas o módulo Bluetooth HC-05 conectado ao Arduíno pode-se desenvolver o app para que ele enviasse os dados devidamente. Devido à forma como o Arduíno recebe os dados, foi necessário primeiramente converter os dados que seriam enviados utilizando a base 64 que é uma forma de compactar os dados enviados em caracteres mais comuns a fim de reduzir a quantidade de bytes necessários para enviar texto através de dispositivos, assim reduzindo o risco de perda de dados por incompatibilidade de caracteres entre diferentes plataformas. Após desenvolver um app com conexão Bluetooth, foi possível se conectar ao módulo Bluetooth HC-05 conectado ao Arduíno, fato confirmado pela mudança no padrão luminoso emitido pelo módulo. Após pesquisas na documentação de bibliotecas de comunicação Bluetooth, pôde-se entender e utilizar as funções necessárias para o envio de dados via Bluetooth. Com isso foi possível alterar algumas funções já existentes para poder enviar dos dados utilizando a biblioteca *react-native-base64* para converter os dados enviados para a base 64 já citada anteriormente. O app permite enfim ligar e desligar o Bluetooth no dispositivo, se conectar a um dispositivo disponível e enviar para ele os valores dos sliders quando eles eram movimentados. Com isso foi possível se conectar ao módulo HC-05 e enviar os dados apropriadamente, ou seja, o aplicativo é capaz de se conectar através de uma tecnologia de comunicação remota à um dispositivo microcontrolado capaz de movimentar um novo braço robótico. Finalizando os resultados com a montagem do braço robótico optou-se por um modelo de braço robótico de acrílico com 5 graus de liberdade que tem 5 servomotores “*Servo Motor MG996R Tower Pro*” que são mais potentes e trazem maior resistência ao peso da estrutura do que os servomotores “*Micro Servo 9g SG90 TowerPro*”, que eram usados anteriormente. Assim, com o desenvolvimento do app que enviava os valores dos sliders apropriadamente para o HC-05 e da conclusão da montagem do braço robótico, os diferentes segmentos do projeto foram conectados, resultando na movimentação esperada do braço robótico, necessitando apenas de alguns ajustes em relação à angulação máxima de cada servo motor. Após esses ajustes, o braço robótico deveria estar completamente funcional. Porém após algumas tentativas percebeu-se que após determinada angulação, os servos não eram capazes de suportar o peso da estrutura e acabavam deixando a estrutura descer lentamente. Essa situação inesperada não se encontra resolvida e espera-se que uma solução seja encontrada em projetos posteriores. Com a conclusão do objetivo do projeto utilizando o Arduíno UNO com o módulo Bluetooth HC-05 e o Sensor Shield v5.0. Passou-se a testar os outros microcontroladores utilizando o mesmo app. Após alguns ajustes nas configurações do Arduino IDE o HC-05 pôde ser conectado ao NodeMCU Amica sem maiores problemas e os valores recebidos puderam ser lidos adequadamente.

# Conclusões

Como conclusão podemos observar o aprendizado quanto à diferentes tecnologias de comunicação remota das quais foram escolhidas o Bluetooth e o Wi-Fi, módulos e microcontroladores disponíveis no mercado que possuam tais tecnologias, sendo escolhidos o HC-05 e o NodeMCU Amica, e também quanto à diversas linguagens de programação utilizáveis para o desenvolvimento de aplicativos, sendo escolhidos o JavaScript e o React Native com os quais foi desenvolvido um aplicativo capaz de controlar o braço robótico através da do módulo Bluetooth HC-05 integrado ao Arduino UNO, o desenvolvimento de um servidor web com o uso do microcontrolador NodeMCU para envio de dados através do Wi-Fi e a montagem de um braço robótico com peças de acrílico que possui 5 graus de liberdade.

[XX ACHO QUE MISTUREI CONCLUSÃO COM RESULTADO XX]