Sistema de Automação Logística de Equipamentos em Estante Inteligente RFID

Gustavo Martins Pacheco[1], Rodolfo Francisco de Oliveira[2]

[1]Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas – Instituto Federal de São Paulo (IFSP) – Campus Hortolândia – Hortolândia, SP – Brasil

[2]Área de Informática – Campus Hortolândia – Instituto Federal de São Paulo (IFSP) – Hortolândia, SP – Brasil

pacheco.gu@gmail.com, rodolfo_foliveira@hotmail.com

Abstract. *The purpose of this work is to build a logistic automation system to control the location, in real time, of objects stored in a smart rack. The system will consist by RFID sensors and an IOT Server to reconcile and persist the identification data with physical positions. The objective is to simplify, streamline and reduce human fail in the operational process, transferring the user's responsibility to the automation system, which will be in charge to update the equipment location data whenever there is a change. Therefore, the employee will no longer have the burden of maintaining the equipment location in the work order, since the work order system will be powered by the automation system through an API, that will be created for this purpose.*

Resumo. *A proposta deste trabalho é construir um sistema de automação logística para o controle da localização, em tempo real, de objetos armazenados em estante inteligente. O sistema será composto por sensores RFID e um IOT Server para conciliar e persistir as informações de identificação com as posições físicas. O objetivo é de simplificar, agilizar e reduzir a falha humana no processo operacional, transferindo a responsabilidade do usuário para o sistema da automação, que ficará a cargo de atualizar os dados sobre a localização dos equipamentos sempre que houver uma alteração. Assim o funcionário não terá mais o encargo de manter atualizada a localização dos equipamentos na ordem de serviço, já que o sistema de ordens de serviço será alimentado pelo sistema de automação através de uma API, que será criada para este propósito.*

1. Introdução

A popularidade da Internet das Coisas (IoT) e a automação logística estão se tornando cada vez mais presentes no mundo conectado. A ideia de interligar dispositivos inteligentes à Internet para oferecer serviços tem transformado a forma como as pessoas se comunicam e como as operações logísticas são realizadas.

Neste cenário, Kevin Ashton propôs o uso de etiquetas eletrônicas, conhecidas como RFID (*Radio Frequency Identification)*, para melhorar a logística da cadeia de produção. Essas etiquetas permitem o controle e o acesso a dados de localização de objetos de alto valor agregado [Ashton 2011].

Considerando o uso do RFID como uma solução para questões logísticas, aproveitando-se da IoT, este recurso pode ser totalmente aproveitado ao integrá-lo aos sistemas de ordem de serviço. Sistemas de ordem de serviço são importantes para organizar e gerenciar as atividades em empresas, ajudando na coordenação, acompanhamento e trazendo vantagens como o aumento da produtividade [Krajewski e Malhotra 2007].

Apesar das vantagens dos sistemas de ordem de serviço no controle de dados, históricos e tarefas, ainda existem desafios como erros humanos na atualização das informações de localização de equipamentos. Para resolver esse problema, um sistema de automação logística pode fornecer dados em tempo real sobre a localização dos equipamentos, atualizando o sistema de ordem de serviço através de uma API (*Application Programming Interface*).

Este trabalho apresenta o desenvolvimento de um sistema de automação logística para controlar a localização de objetos em uma estante inteligente. O sistema utiliza identificação por radiofrequência (RFID), *firmware* e um servidor IoT para conciliar informações de localização física e disponibilizá-las por meio de API.

O objetivo é simplificar e agilizar o processo operacional, transferindo a responsabilidade de atualização de localização para o sistema de automação. Além disso, o sistema permite ao usuário escolher o espaço de armazenamento adequado com base no volume e no peso do equipamento.

O trabalho é estruturado em capítulos que abordam diferentes aspectos do sistema proposto. No Capítulo 2, é apresentada uma revisão bibliográfica dos principais temas relacionados ao projeto, tais como a Internet das Coisas e a tecnologia RFID. O Capítulo 3 aborda trabalhos correlatos que se aproximam do sistema proposto, fornecendo uma visão geral das soluções existentes. Já o Capítulo 4 especifica a metodologia aplicada no desenvolvimento do projeto, incluindo seus processos, ferramentas e padrões utilizados para sistematizar o planejamento e sua execução. No Capítulo 5, é apresentada a proposta do projeto e como ela pode ser uma solução para atender às necessidades identificadas em sistemas de ordem de serviço. No Capítulo 6, são descritos os materiais e métodos utilizados durante o desenvolvimento do projeto, incluindo os equipamentos e dispositivos utilizados, bem como os *softwares* e linguagens de programação adotados. No Capítulo 7, é apresentado os resultados dos testes realizados com o protótipo e as conclusões finais sobre o sistema desenvolvido, incluindo suas limitações e possíveis direções para trabalhos futuros.

2. Revisão Bibliográfica

A Internet das Coisas e a tecnologia RFID são áreas de grande interesse na atualidade, pois oferecem uma variedade de benefícios e possibilidades em diferentes setores. Este capítulo abordará sobre os conceitos fundamentais dessas tecnologias, como a comunicação entre dispositivos, sensores e *tags*, e como esses dispositivos podem ser utilizados em diversas aplicações. Por fim, também serão explorados os modelos de arquitetura da IoT e sobre o protocolo MQTT que é uma das tecnologias de comunicação que permitem a transmissão de informações entre dispositivos.

2.1. RFID

RFID (*Radio-Frequency Identification*) é uma tecnologia de identificação automática por meio de radiofrequência que permite a identificação e rastreamento de objetos, animais ou pessoas por meio de etiquetas eletrônicas (*tags*) que contêm informações armazenadas em microchips.

Essas *tags* podem ser lidas remotamente por leitores que utilizam ondas de rádio para se comunicar com as *tags* e obter informações sobre elas, como sua identificação, localização e outras informações relevantes. O RFID é amplamente utilizado em diversos setores, como logística, varejo, saúde e segurança, entre outros [Benedetti e Maselli 2022].

2.2. Internet das Coisas

A Internet das Coisas (*Internet of Things* - IoT) consiste em uma gama de tecnologias que permitem a conexão de dispositivos inteligentes para coletar, transmitir e processar dados em tempo real. É um conceito que tem recebido grande atenção nos últimos anos, e tem o potencial de revolucionar a forma como vivemos e trabalhamos.

O termo IoT não é algo novo e vem sendo discutido por empresas e acadêmicos especialistas no assunto por muitos anos. O primeiro dispositivo IoT foi uma torradeira conectada à Internet, apresentada na *INTEROP '89 Conference* por John Romkey, que podia ser ligada e desligada através de um computador com rede TCP/IP [Bouhaï e Saleh 2017].

Uma solução IoT é composta por várias camadas, cada uma com sua função específica. A especificação adequada de cada camada é crucial para o sucesso de um projeto de IoT. A Figura 1 apresenta um modelo de 4 camadas que definem a IoT [IEEE 2016]. No entanto, é relevante mencionar que alguns autores propõem modelos com 5 camadas [Buyya E Dastjerdi 2016] [Al-Fuqaha et al. 2015].

A seguir será apresentado o modelo de 4 camadas, por ser o mais simples e o que possui maior relação com a proposta deste trabalho:

Figura 1. Camadas da IoT.

1. Camada de Percepção: Esta é a primeira camada de uma rede IoT e é responsável por coletar informações do ambiente físico através de sensores e dispositivos conectados à rede. Esses sensores podem coletar diferentes tipos de dados, como temperatura, umidade, luminosidade, movimento e outros. Depois de serem coletados, os dados são enviados para a próxima camada de rede.
2. Camada de Rede: A principal função da camada de rede é de enviar os dados coletados pelos sensores para as aplicações, através das técnicas de roteamento para garantir que eles cheguem ao seu destino. A camada de rede também é responsável por gerenciar a comunicação entre os dispositivos IoT, permitindo que eles troquem informações entre si.
3. Camada de Processamento (*Middleware*): É uma parte do sistema que processa e interpreta os dados coletados pela camada de percepção. Ela utiliza diversas ferramentas como algoritmos de aprendizado de máquina e análise de dados a fim de transformar esses dados em informações úteis e compreensíveis para serem usadas em outras partes do sistema. Ou seja, essa camada tem a finalidade de transformar os dados brutos em informações úteis para aplicações interessadas.
4. Camada de Aplicação: Esta camada é responsável pela entrega dos serviços e aplicações IoT aos usuários finais. Ela inclui as aplicações específicas IoT, como sistemas de monitoramento remoto, controle de acesso, rastreamento de ativos, entre outros.

No contexto de uma estante inteligente, a camada de aplicação é responsável por fornecer serviços e aplicações especializadas que ajudam a monitorar e controlar a estante de uma forma útil e eficiente. Por exemplo, sensores instalados na estante

podem coletar dados sobre a localização, o peso e a disponibilidade dos produtos armazenados e enviar esses dados para a camada de aplicação. A partir desses dados, as aplicações podem fornecer aos usuários dados precisos em tempo real sobre a localização e disponibilidade dos produtos, permitindo que eles otimizem a gestão de estoque, aumentem a eficiência na reposição de produtos e evitem perdas.

2.3. Protocolo MQTT

Na arquitetura da IoT, o protocolo MQTT (*Message Queuing Telemetry Transport*) faz parte da camada de rede. O mesmo define um conjunto de regras para formatação e processamento dos dados de forma que possam ser lidos e interpretados por diferentes destinatários, independente do sistema operacional ou do *hardware* escolhido.

O MQTT foi desenvolvido para ser bastante leve quando se fala em uso da largura de banda e recursos computacionais. Isso faz dele o protocolo mais indicado para dispositivos com recursos bem limitados, como sensores e microcontroladores [Cosmi e Mota 2019].

Este protocolo permite que a comunicação ocorra de maneira assíncrona, ou seja, de forma que os dispositivos possam se comunicar de forma independente, de acordo com a demanda, sem que para isso seja necessário estabelecer uma comunicação contínua.

Outra característica do protocolo MQTT é que ele trabalha com o conceito de mensagens em forma de tópicos que são criadas por um publicador (*Publisher*) e disponibilizadas a um assinante (*Subscriber*), como pode ser observado na Figura 2.

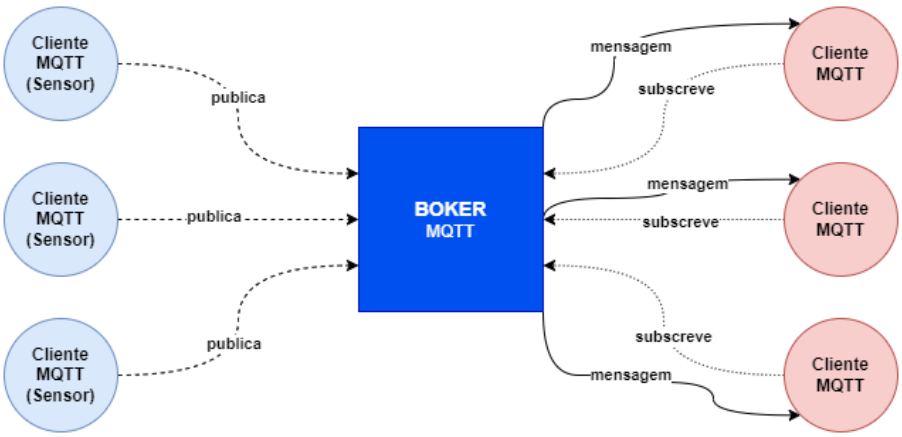

Figura 2. Protocolo MQTT.

O tópico é um identificador de mensagens que permite categorizá-las e determinar quais assinantes receberão essas mensagens. Os publicadores enviam mensagens para um servidor *broker*, identificadas através de um tópico, e os assinantes se inscrevem em tópicos específicos no *broker* recebem as mensagens desejadas. Já um publicador é um dispositivo ou aplicativo que envia mensagens em um tópico

específico. Essas mensagens são enviadas a um servidor centralizado chamado *broker*. Já o assinante é um dispositivo ou aplicativo que se conecta ao *broker* e recebe mensagens de um ou mais tópicos específicos.

3. Trabalhos Correlatos

Para realizar o levantamento dos trabalhos correlatos, foram utilizadas as plataformas Google Acadêmico (Google Scholar) e o IEEE. Os termos empregados na pesquisa foram "Internet das Coisas (IoT)", "RFID", "Automatização", "Armazenamento", "Eficiência", "Gerenciamento", "Armário inteligente" e "Localização de livros". O período de investigação compreendeu de 24 de abril de 2023 a 11 de maio de 2023.

No estudo "*ServicesLocker*: uma solução de armário inteligente baseado em Internet das Coisas", os autores tiveram por objetivo desenvolver um armário inteligente que utiliza o conceito de IoT para oferecer serviços intermediários para lavanderias (entrega e retirada de roupas) e a entrega de mercadorias. O trabalho apresenta uma arquitetura de IoT em cinco camadas e utiliza tecnologias como Angular, Node.js, Thingsboard e Raspberry Pi. Essa solução visa facilitar a entrega de encomendas em residências, superando o desafio da ausência do destinatário durante a entrega. [Bortoli e Costa 2020].

Já no trabalho "*Intelligent Refrigerator based on Internet of Things*" (Geladeira Inteligente baseada em Internet das Coisas, é apresentado o desenvolvimento de uma geladeira inteligente baseada na Internet das Coisas (IoT) e que utiliza a tecnologia RFID. A proposta deste trabalho correlato é de criar um sistema de gerenciamento de alimentos dentro do refrigerador, onde os dados gerados são automaticamente registrados e enviados para uma plataforma de nuvem. Isso permite que os usuários visualizem facilmente as informações sobre os alimentos armazenados e recebam recomendações de receitas com base nos itens disponíveis. O refrigerador também emite avisos notificando quando os alimentos estão próximos da data de validade ou se já estão vencidos. O módulo de controle sem fio permite a conexão do refrigerador com dispositivos inteligentes, como um *smartphone* ou *tablet*, facilitando a verificação do conteúdo interno do refrigerador. A proposta tem como objetivo tornar o gerenciamento dos alimentos mais eficiente, reduzir desperdícios e oferecer uma vida mais conveniente aos usuários [Qiao et al. 2017].

Em outro trabalho correlato, intitulado "Sistema RFID de Baixo Custo para Localização de Acervo Bibliográfico", os pesquisadores desenvolveram um sistema que utiliza a tecnologia RFID para otimizar a gestão e organização de bibliotecas. O objetivo principal do sistema é de disponibilizar uma solução mais acessível e eficiente para localizar livros em acervos bibliográficos. Nessa pesquisa, foi proposto um sistema que utiliza *tags* (etiquetas) RFID fixadas nos livros, permitindo que sejam rastreados de forma automatizada. Assim, é possível obter informações precisas sobre a localização de cada livro na biblioteca, simplificando o processo de busca e empréstimo. O sistema foi projetado para ser uma solução de baixo custo, de forma a tornar a tecnologia RFID mais acessível a instituições com limite de recursos [Torres et al. 2017].

Pode-se concluir que os trabalhos correlatos supracitados podem ser relacionados ao presente projeto por ambos compartilharem do uso da tecnologia RFID aplicada ao conceito de Internet das Coisas (IoT) para fornecer soluções inteligentes,

mesmo que em contextos diferentes. Todas as principais funcionalidades descritas, nos trabalhos correlatos, estão resumidas na tabela 1.

Por fim, é importante ressaltar que a principal vantagem deste trabalho em relação aos trabalhos correlatos mencionados é a presença de uma API que possibilita a integração com outros sistemas em diversos contextos de negócios. Ao integrar a estante com esses sistemas, torna-se possível aproveitar todas as funcionalidades oferecidas para o armazenamento de qualquer tipo de objeto, em qualquer contexto que faça uso de uma prateleira. Essa flexibilidade e adaptabilidade tornam esse trabalho uma solução versátil e escalável para atender diferentes necessidades.

Tabela 1. Comparativo de funcionalidades dos trabalhos correlatos.

Funcionalidade	***Estante Inteligente RFID***	***Intelligent Refrigerator***	***ServicesLocker***	***Sistema de Acervo Bibliográfico***
Uso de Tecnologia RFID	✓	✓	✓	✓
Monitoramento e Localização	✓	✓	✓	✓
Integração via API	✓	✓	✓	✗
Registro de Movimentações	✓	✓	✓	✓
Consulta de Espaços Disponíveis	✓	✗	✗	✗
Aplicação Principal no IoT Server	✓	✓	✓	✗
Uso do protocolo MQTT	✓	✓	✓	✗
Possui fins Didáticos	✓	✗	✓	✓

✓ = Possui.

✗ = Não possui.

4. Metodologia

Para o desenvolvimento deste projeto, foi necessária a criação de um fluxo de trabalho de fácil visualização e acompanhamento, que permitisse agregar novas funcionalidades

que pudessem surgir no decorrer do projeto, além de possibilitar correções e adequações em partes já implementadas. Para atender a essa necessidade, foi utilizado o método de desenvolvimento incremental em conjunto com o quadro *Kanban*. No entanto, é importante esclarecer que o quadro Kanban foi utilizado como uma ferramenta de gerenciamento e não como uma metodologia Kanban em si.

O desenvolvimento incremental é uma abordagem em que uma implementação inicial de um sistema é criada e aprimorada por meio de várias versões, com base nos *feedbacks* dos usuários. As atividades de especificação, desenvolvimento e validação são intercaladas, permitindo um rápido progresso e ajustes mais fáceis. Cada incremento incorpora funcionalidades importantes, possibilitando entregas rápidas e obtenção de *feedbacks* dos clientes ao longo do processo, facilitando mudanças e permitindo a entrega de *software* útil. [Sommerville 2011].

Com base no conceito de desenvolvimento incremental, mencionado acima, o projeto foi dividido em partes, chamadas de demandas, e representadas como cartões, ou *cards* (em inglês), no quadro Kanban. As demandas são implementadas uma por vez, seguindo uma sequência até que seja atingido o pleno funcionamento de um módulo do projeto. Após o módulo ser concluído, ele passa por análise e testes para garantir a qualidade e o cumprimento dos requisitos levantados inicialmente. Caso algo não esteja de acordo, ou seja, considerado insuficiente, uma demanda de aprimoramento é criada. Esse processo de desenvolvimento é ilustrado na Figura 3.

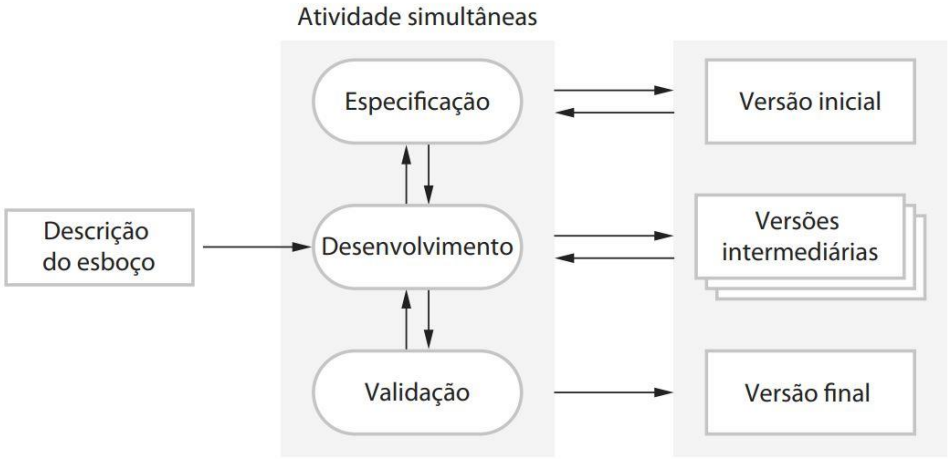

Figura 3. Fluxo do desenvolvimento incremental [Sommerville 2011].

O uso do quadro Kanban permite aproveitar os benefícios visuais e organizacionais, pois fornece uma representação visual do fluxo de trabalho e do status das demandas, ajudando na tomada de decisões para otimizar o processo de desenvolvimento. Os principais tipos de status são: "*To Do*" (A fazer), representando as tarefas pendentes; "*In Progress*" (Em andamento), indicando as tarefas em execução; e "*Complete*" (Concluído), que representa as tarefas finalizadas [Aguiar e Peinado 2007]. A Figura 4 apresenta um exemplo de quadro Kanban utilizado no gerenciamento do projeto.

Figura 4. Quadro Kanban.

5. Apresentação da Proposta

Este trabalho tem como proposta criar uma solução de automação logística para sistemas de ordem de serviço, para identificar a localização de objetos em uma estante inteligente.

A ideia é criar uma prateleira inteligente que possua um sensor capaz de realizar a leitura de objetos por meio de identificadores presos a eles e que disponibilize essas informações a um sistema terceiro, para que possa vincular a posição do objeto a uma ordem de serviço ou realizar qualquer outro tipo de controle em que seja necessário saber a localização de um objeto em uma prateleira ou nicho.

A Figura 5 apresenta uma visão geral da proposta, bem como os módulos que constituem a mesma. Nela, é possível observar que o sistema é composto por um conjunto de sensores que se comunicam com o IoT Server, sendo este último responsável por disponibilizar os dados para outras aplicações que vão consumir os dados disponibilizados via API. É no IoT Server onde fica a aplicação principal do sistema, com diferentes módulos responsáveis por funções distintas, sendo eles: Broker, Módulo Gerenciador, Banco de Dados e API.

O projeto foi desenvolvido com o objetivo de ser um protótipo experimental de caráter didático, destinado a testar o funcionamento do sistema e das tecnologias de *hardware* e *software* escolhidas.

Nos próximos capítulos, serão abordados os requisitos funcionais do sistema, bem como a utilização dos sensores para identificação de objetos. Será discutido o papel

do IoT Server e como seus módulos contribuirão para o funcionamento do sistema, incluindo a captura e gerenciamento dos dados pelo *broker*, a persistência desses dados no banco de dados e sua disponibilização para sistemas terceiros por meio de uma API.

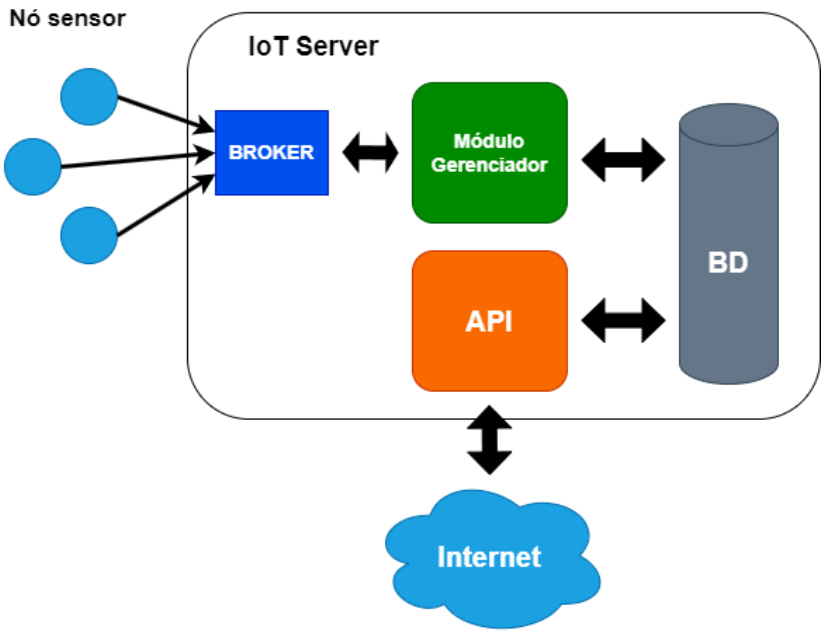

Figura 5. IoT Server

5.1. Requisitos Funcionais

Requisitos funcionais são descrições detalhadas das funcionalidades esperadas e do comportamento esperado de um sistema de *software*, especificando as ações que o sistema deve executar e as saídas esperadas que estejam de acordo com as respostas a entradas específicas. Esses requisitos são importantes para o desenvolvimento do sistema, pois garantem que o sistema atenderá as necessidades e expectativas dos usuários [Sommerville 2011].

A Tabela 2 apresenta os requisitos funcionais e a Tabela 3 os requisitos não funcionais, que foram levantados para este projeto.

Tabela 2. Requisitos Funcionais.

Código	Nome	Descrição
RF-01	Identificação por RFID.	O sensor deve ser capaz de identificar os equipamentos por meio de *tags* RFID, possibilitando a leitura das informações contidas nas *tags*.
RF-02	Registro de equipamentos.	O sistema deve permitir registrar em banco de dados a localização do objeto com base no código

		da *tag* e o endereço MAC do sensor.
RF-03	Atualização automática da localização.	O sistema deve atualizar automaticamente na base de dados a localização dos equipamentos sempre que houver uma mudança de posição na estante.
RF-04	Registrar o histórico de movimentações.	Sempre que o objeto for movido para outra prateleira, o sistema deve registrar no banco de dados, data, hora e código da *tag* e da prateleira para manter o histórico do objeto.
RF-05	Integrar com outros sistemas.	O sistema de automação logística deve permitir ser integrado a outros sistemas, alimentando-os com informações atualizadas sobre a localização dos objetos, por meio de uma API.
RF-06	Cadastrar novas prateleiras.	O sistema deverá permitir o cadastro de novas prateleiras, associando a elas o endereço MAC do sensor, por meio de uma API.
RF-07	Localizar objetos.	O sistema deve permitir localizar um equipamento específico por meio da *tag*, fornecendo a posição da estante inteligente.
RF-08	Consulta de espaços livres na prateleira.	O sistema deve permitir o gerenciamento dos espaços disponíveis na estante, fornecendo informações sobre espaços livres nas prateleiras.

Tabela 3. Requisitos Não Funcionais.

Código	Nome	Descrição
RNF-01	Escalabilidade.	O sistema deverá permitir se adequar a diferentes tamanhos de negócios, com diferentes quantidades de prateleiras.
RNF-02	Protocolo de Comunicação.	O sistema deverá utilizar protocolo MQTT entre o sensor e o IoT Server.
RNF-03	Integração com diferentes sistemas.	O sistema deverá trabalhar com API para possibilitar a integração com diferentes sistemas terceiros de ordem de serviço.

Não foram elencados requisitos não funcionais de segurança e de desempenho devido a restrições de tempo e escopo definidas inicialmente, já que este projeto possui finalidade acadêmica e não comercial.

5.2. Sensores RFID

Neste projeto os sensores desempenham um papel crucial de identificar os equipamentos armazenados nas prateleiras. Cada prateleira é equipada com um único leitor, que pode ser fixado na base ou nas laterais, com o objetivo de identificar em qual prateleira o equipamento está localizado.

A Figura 6 ilustra o funcionamento de um sensor RFID, que envolve uma série de processos de inicialização para permitir que o sensor desempenhe sua função. Primeiramente, o sensor inicializa a porta serial para permitir a comunicação entre o *hardware* de processamento e o *hardware* de leitura. Em seguida, as dependências responsáveis pelo funcionamento do *hardware* de leitura são inicializadas, seguidas pelas dependências para a comunicação entre o sensor e o *broker* através de uma conexão Wi-Fi (*Wireless Fidelity*). Após isso, as dependências para a comunicação com o *broker* através do protocolo MQTT são inicializadas. Após essa sequência de inicializações, o sensor entra em modo de escuta e, assim que uma *tag* é aproximada, a leitura é realizada. Em seguida, uma *string* contendo o valor lido, juntamente com o endereço MAC, é criada e publicada no *broker*.

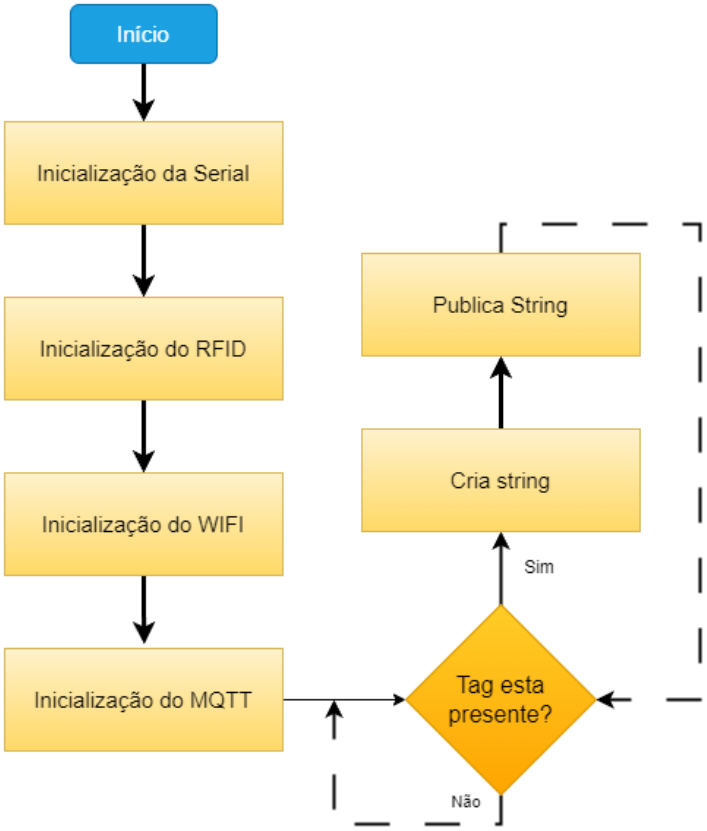

Figura 6. Fluxograma Sensor RFID.

5.3. IoT Server

O IoT Server é o responsável por realizar o intermédio entre os sensores e o sistema de ordem de serviço através de um *broker*, um módulo gerenciador, um banco de dados e um *webserver*. Este último possui uma API que poderá ser consumida por um sistema de ordens de serviço de terceiros.

5.3.1. Broker

O *broker* é um intermediário de mensagens em um sistema de comunicação distribuído. Ele recebe as mensagens de um ou mais dispositivos conectados e as roteia para o(s) dispositivo(s) destinatário(s), com base nas informações de destino contidas nas mensagens [Marzolla e Mottola 2016].

Na arquitetura MQTT, o *broker* é responsável por manter a lista de tópicos que os clientes (dispositivos) podem assinar ou publicar, bem como gerenciar as conexões dos clientes e a segurança do sistema. Ele é o ponto central de comunicação para o tráfego de mensagens entre os dispositivos conectados e, portanto, desempenha um papel crítico na arquitetura da Internet das Coisas.

5.3.2. Módulo Gerenciador

O módulo gerenciador é responsável por subscrever no *broker* e realizar a coleta de dados referente as leituras realizadas pela rede de sensores. Estes dados são validados e tratados de forma a registrar no banco de dados a última movimentação ocorrida.

5.3.3. Banco de Dados

Um banco de dados relacional é um sistema de gerenciamento de banco de dados que organiza os dados em tabelas relacionadas entre si por meio de chaves primárias e estrangeiras, seguindo os princípios da teoria dos conjuntos [Date 2004].

As vantagens de um banco de dados relacional incluem a estrutura organizada, integridade dos dados, flexibilidade, escalabilidade e segurança. Um banco de dados relacional permite a criação de consultas complexas para recuperar informações específicas de várias tabelas e atualização dos dados sem afetar outras partes do sistema.

Devido as vantagens apresentadas, este projeto será composto por um banco de dados relacional intitulado “smartshelf”.

A base de dados "smartshelf" foi criada para armazenar informações relacionadas ao monitoramento dos objetos nas prateleiras e o registro de movimentações desses objetos entre as prateleiras. A base de dados é composta por três tabelas principais: "NoSensor", "Prateleira" e "Registro", conforme ilustrado na figura 7.

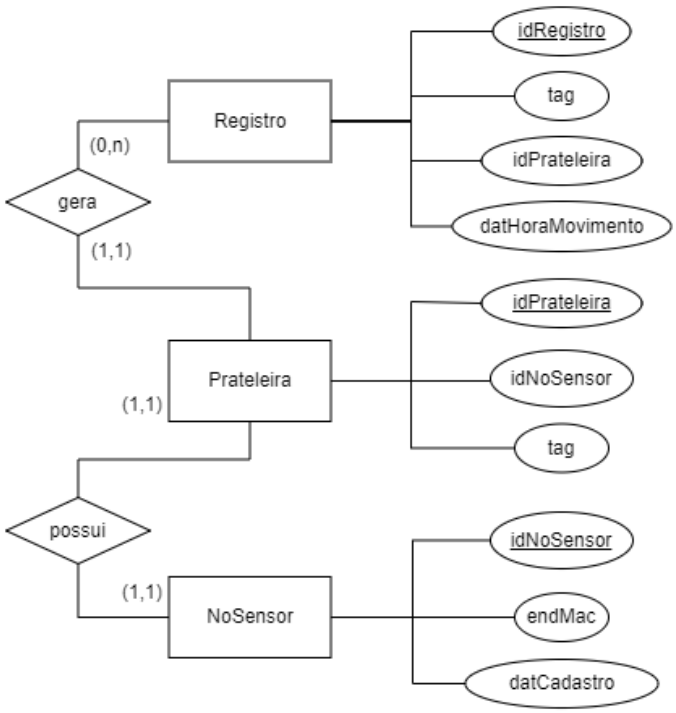

Figura 7. Diagrama Entidade Relacionamento (DER).

A tabela "NoSensor" é responsável por armazenar informações sobre os sensores instalados nas prateleiras. Ela possui os atributos "idNoSensor", que é uma chave primária artificial, segundo Elmasri et. al, criada para identificar o sensor, "endMac", que armazena o endereço MAC do sensor que também é único, porém é uma chave secundária natural, também segundo Elmasri et. al, por ser definida pela interface de rede do sensor, e "datCadastro", que registra a data de cadastro do sensor [ELMASRI et al. 2018].

A tabela "Prateleira" registra as informações sobre cada prateleira. Ela possui os atributos: "idPrateleira", que é uma chave primária artificial criada para identificar a prateleira, "*tag*", que armazena o valor da *tag* RFID do objeto armazenado e "idNoSensor", que é uma chave estrangeira que relaciona os dados da tabela "Prateleira" com a tabela "NoSensor". Isso permite que cada prateleira esteja vinculada a um sensor específico.

A tabela "Registro" armazena os registros de movimentações nas prateleiras, como forma de criar um histórico. Ela possui os atributos "idRegistro", uma chave primária artificial que identifica o registro gerado, "*tag*", que armazena o identificador do objeto movimentado, "datHoraMovimento", que registra a data e hora em que a movimentação ocorreu, e "idPrateleira", uma chave estrangeira que relaciona o registro à prateleira que sofreu a ação de inclusão do objeto (*tag*). Dessa forma, é possível rastrear em qual prateleira ocorreu cada movimentação, com base nos registros gerados.

Além das tabelas, a base de dados também inclui um gatilho (*trigger)* chamado "MovimentacaoPrateleiraTrigger" que é acionado após a inclusão ou alteração de registros na tabela "Prateleira". Quando acionado, o *trigger* chama o procedimento

(*procedure*) "RegistrarMovimentacaoPrateleira", que realiza a inserção de um novo registro na tabela "Registro", para se criar um histórico de movimentações de objetos entre prateleiras.

Essa estrutura de base de dados permite o monitoramento e histórico dos objetos nas prateleiras, fornecendo informações importantes para o controle logístico, através das APIs.

5.3.4. API

API é a sigla para "*Application Programming Interface*", que em português significa "Interface de Programação de Aplicativos". É um conjunto de rotinas, protocolos e ferramentas para a criação de *software* e integração de sistemas. A API define um conjunto de regras e padrões que permitem a comunicação entre diferentes aplicações, permitindo a troca de informações e funcionalidades. [Red Hat 2023].

Este projeto possui uma API que define vários *endpoints*, conforme apresentado na Tabela 3. Cada *endpoint* é responsável por lidar com uma funcionalidade específica da API.

Tabela 3. *Endpoints* da API.

Endpoint	**Método**	**Descrição**
/prateleira	GET	Consulta todas as prateleiras no banco de dados.
/prateleira_livre	GET	Consulta todas as prateleiras livres no banco de dados.
/registro	GET	Retorna todos os registros. Ou seja, retorna o histórico de movimentações dos objetos que ocorreram entre as prateleiras.
/localiza_objeto	POST	Localiza um objeto específico com base em sua *tag*.
/prateleira	POST	Cadastra uma nova prateleira no banco de dados.

6. Ferramentas

Neste capítulo, serão discutidas as ferramentas utilizadas na realização deste projeto. Cada ferramenta foi selecionada cuidadosamente para viabilizar a solução proposta.

6.1. Sensores RFID

Para a construção do sensor foram utilizados um leitor RFID-RC522 da NXP [Semiconductors 2014] e uma plataforma de prototipagem NodeMCU modelo AMICA [NodeMcu 2020]. Ambos são equipamentos de baixo custo e possuem os requisitos mínimos de *hardware* necessários para a elaboração do protótipo.

O leitor RFID-RC522 possui um circuito integrado MFRC522 que se conecta ao NodeMCU através de uma conexão física serial padrão SPI - *Serial Peripheral Interface* (Interface Periférica Serial, em português), que realiza a comunicação com o NodeMCU através de um algoritmo do tipo fila chamado FIFO (primeiro a entrar, primeiro a sair em português). [Tanenbaum 2015]. A leitura da *tag* é realizada pelo módulo leitor através de um campo eletromagnético gerado pelo mesmo, que faz retornar as informações da *tag* através de rádio frequência [Cunha 2006].

O NodeMCU modelo AMICA, representado na Figura 8 e montado conforme a Figura 9, é a plataforma de prototipagem utilizada, possui um processador de 32 bits sob o chip ESP8266. O mesmo pode ser programado com linguagem Lua ou através da IDE do Arduíno [Arduino 2020], suporta comunicação Wi-Fi, possui porta serial USB para comunicação e alimentação do circuito além de 10 pinos para comunicação digital e 1 pino para comunicação analógica, que podem ser utilizados com diferentes módulos desenvolvidos para diversas finalidades [NodeMcu 2020]. Neste projeto, o NodeMCU foi utilizado em conjunto com o módulo leitor RFID, conforme mencionado acima.

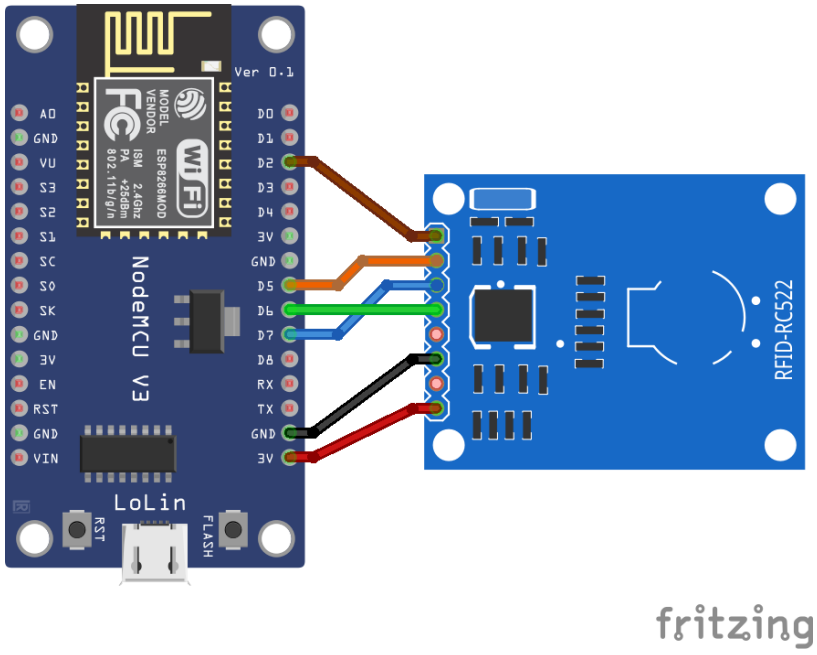

Figura 8. Esquema de hardware (NodeMCU + RFID-RC522)

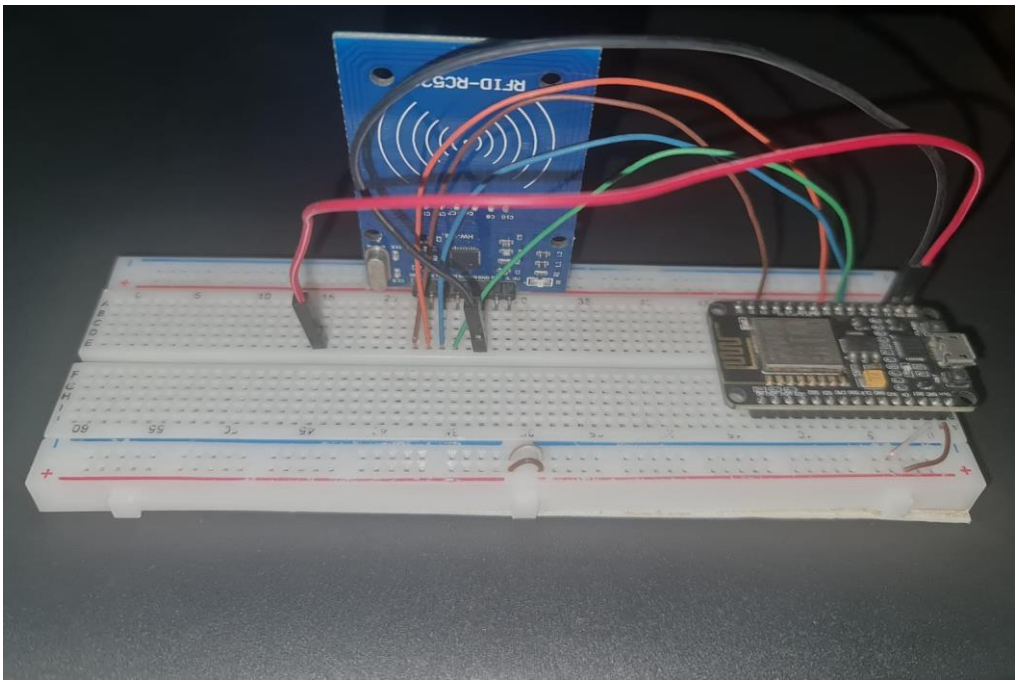

Figura 9. Protótipo em *protoboard* (matriz de contato) com NodeMCU + RFID-RC522

6.2. Mosquitto Broker

O Eclipse Mosquitto é um *broker* de mensagens de código aberto (licenciado sob EPL/EDL) que implementa as versões 5.0, 3.1.1 e 3.1 do protocolo MQTT. O Mosquitto é leve e adequado para uso em todos os dispositivos, desde computadores de placa única de baixa potência até servidores completos [Mosquitto 2020].

Neste projeto, o Mosquitto desempenha um papel importante ao possibilitar a comunicação entre os sensores RFID e o IoT Server, especialmente por ser adequado para o NodeMCU, que é o *hardware* escolhido para o protótipo do sensor.

6.3. Módulo Gerenciador

O módulo gerenciador foi desenvolvido utilizando a linguagem de programação Python.

Python é uma linguagem de programação de alto nível, interpretada e com tipagem dinâmica [Matthes 2016]. A escolha dessa linguagem neste projeto se deve ao seu benefício de possuir uma sintaxe simples e legível, fato este que a tornou em uma escolha popular entre os desenvolvedores em diversos tipos de aplicações, que vão desde desenvolvimento *web* e análise de dados até automação e inteligência artificial.

6.4. Banco de Dados

Neste projeto, o MySQL foi escolhido como o sistema de gerenciamento de banco de dados. Ele é um sistema de gerenciamento de banco de dados relacional de código aberto que utiliza a linguagem SQL (Structured Query Language), ou Linguagem de Consulta Estruturada, em português, para gerenciar dados em tabelas.

O MySQL possui uma licença dual, oferecendo uma opção gratuita para finalidade não comercial, como no caso deste projeto que possui fins acadêmicos, e opera sob o modelo cliente-servidor. Assim como é muito flexível e pode ser expandido ou personalizado de muitas maneiras para atender necessidades específicas de um aplicativo ou ambiente de banco de dados. Possui uma comunidade ativa e é bastante utilizado em diferentes tipos de aplicativos. É compatível com várias plataformas e é mantido pela Oracle, em conjunto com a comunidade MySQL [Mysql 2023].

6.5. API

A API deste projeto também foi desenvolvida utilizando a linguagem de programação Python, porém foi utilizado o Flask. O Flask é um *framework* para uso em linguagem Python e tem a finalidade de trazer consigo toda uma estrutura para facilitar a implementação de APIs [Pallets Team 2022].

7. Desenvolvimento e Validações

Nesta seção, é abordado o processo de desenvolvimento do sistema, desde a criação do protótipo do sensor RFID até a implementação do IoT Server e da API. Além disso, serão discutidas as etapas de validação que foram conduzidas para garantir que o sistema esteja operacional.

7.1. Desenvolvimento do Sensor RFID

O desenvolvimento do sensor RFID foi o ponto de partida para a concretização do projeto. No processo de construção do protótipo do sensor, um leitor RFID-RC522 da NXP e uma plataforma NodeMCU modelo AMICA foram utilizados.

A conexão entre o leitor RFID-RC522 e o NodeMCU foi estabelecida por meio da interface serial padrão SPI. Esse leitor desempenhou o papel de identificar os equipamentos armazenados nas prateleiras da estante inteligente. Sempre que uma *tag* RFID se encontrava dentro do campo eletromagnético gerado pelo leitor, a informação de identificação contida na *tag* era lida e transmitida ao NodeMCU, que em seguida encaminhava esses dados ao IoT Server.

7.2. Configuração do Mosquitto Broker

O próximo passo consistiu na configuração do *broker* Mosquitto. O *broker* tem o importante papel de possibilitar a comunicação entre os sensores RFID e o IoT Server.

O Mosquitto foi selecionado devido à sua compatibilidade com o NodeMCU, e sua capacidade de implementar o protocolo MQTT. Essa escolha permitiu que a comunicação eficiente entre o sensor RFID e o servidor ocorresse.

7.3. Implementação do IoT Server

O módulo gerenciador possui métodos implementados, conforme a Figura 10, para que o IoT Server possa identificar dados de localização de um equipamento sempre que o mesmo for movido de posição.

```
52
53  # Criação do objeto cliente MQTT
54  cliente = mqtt.Client("manager")
55
56  # Associa a função de callback ao cliente
57  cliente.on_message = onMessage
58
59  # Estabelece conexão com o broker MQTT
60  cliente.connect(enderecoBroker)
61
62  # Assina o tópico '/topico/sensor'
63  cliente.subscribe("/topico/sensor")
64
65  # Inicia o loop de processamento de mensagens
66  cliente.loop_forever()
67
68
```

Figura 10. Código principal do módulo gerenciador.

Um objeto do tipo mqtt.Client é criado, com o identificador "manager", representando o cliente MQTT. A função onMessage é associada como *callback* para lidar com mensagens recebidas. O cliente estabelece conexão com o *broker* MQTT usando o endereço fornecido. Ao se inscrever no tópico "/topico/sensor", o cliente passa a receber mensagens publicadas pelo *broker* nesse tópico. Para processar essas mensagens de forma contínua, o cliente inicia um *loop* infinito, utilizando o método loop_forever(), que chama a função de *callback* onMessage sempre que uma nova mensagem é recebida.

```python

#LOCALIZAR OBJETO ESPECÍFICO
@app.route('/localiza_objeto', methods=['POST'])
def getObjeto():

    tag = request.json
    number = int(tag['tag'])

    myCursor = mydb.cursor()
    sql = (f"SELECT * FROM Prateleira WHERE tag = {number}")
    myCursor.execute(sql)
    prateleira = myCursor.fetchall()

    local = list()

    for prat in prateleira:
        local.append(
            {
                'idPrateleira' : prat[0],
                'tag' : prat[1],
                'idNoSensor' : prat[2]
            }
        )

    return make_response(
        jsonify(
            mensagem='Local do objeto:',
            dados=local
        )
    )

```

Figura 11. Trecho do código da API para consulta de prateleiras específicas.

O trecho de código da API, conforme a Figura 11, é responsável por retornar os dados da localização de um objeto específico. Ele espera receber uma solicitação POST com dados JSON contendo uma chave chamada '*tag*'. Em seguida, ele consulta o banco de dados para encontrar registros na tabela 'Prateleira' onde a coluna '*tag*' é igual ao valor fornecido. Como resposta, a API retorna os detalhes dos objetos encontrados, incluindo 'idPrateleira', 'tag' e 'idNoSensor', em formato JSON.

7.4. Validações do Sistema

Com o intuito de garantir que o sistema operasse conforme o escopo previsto, algumas etapas de validação foram conduzidas:

1. Testes de leitura: Não foi possível submeter o sensor a testes de leitura da *tag,* visto que no decorrer do projeto a placa de prototipagem foi danificada, impossibilitando que a validação pudesse ocorrer. Portanto para realizar as validações necessárias do código principal do sistema e do banco de dados foi necessário realizar o *mock* (objetos que simulam o comportamento de objetos

reais de forma controlada) na tabela “Prateleira” e “NoSensor”, visando simular que o sistema já possui esses cadastros. Conforme a Figura 12, através da linha de comando é possível simular o comportamento do sensor, ao enviar os dados do equipamento e de sua localização ao tópico do Broker.

O código do sensor não representa a parte fundamental do projeto visto que o sistema principal opera no IoT server e independe do tipo de sensor a ser utilizado.

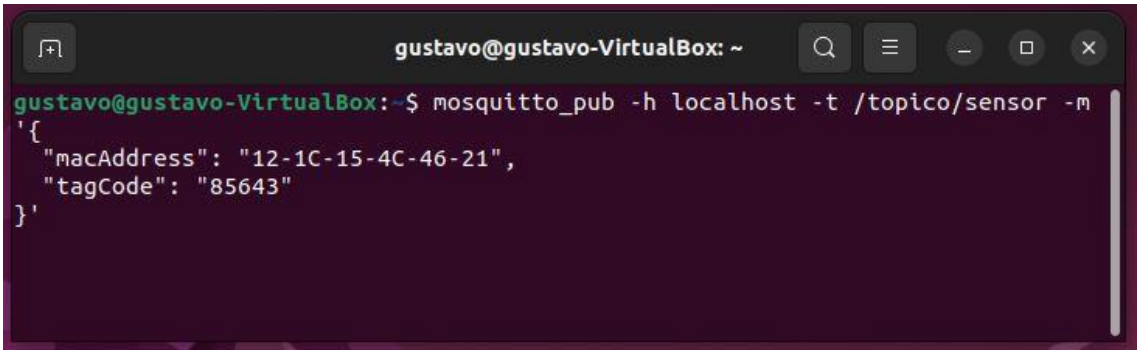

Figura 12. Comando para publicar os dados no tópico do Broker.

1. Validação do banco de dados: Após cada mudança na localização dos equipamentos na prateleira, foram efetuadas validações para verificar se os dados eram corretamente atualizados e relacionados nas tabelas do banco de dados.
2. Testes da API: A API foi testada para confirmar se os *endpoints* operavam conforme o planejado e que as consultas retornavam os resultados esperados. Para testar os *endpoints* da API foi utilizado o *software PostMan*, que é uma ferramenta muito útil para auxiliar no desenvolvimento e testes de API’s [Postman 2023]. Um exemplo a ser citado é o *endpoint* da API que, que é consumido através da seguinte URI, http://localhost:5000/localiza_objeto através do método de requisição, do tipo POST, do protocolo HTTP.

Ao longo do processo de validação, algumas áreas que precisavam de ajustes e aprimoramentos foram identificadas. Isso resultou em correções de *bugs* garantindo que o sistema se tornasse operacional.

8. Conclusão

Este projeto apresentou o desenvolvimento de uma solução de automação logística baseada em uma estante inteligente RFID, demonstrando a viabilidade da identificação e rastreamento de objetos em prateleiras com o uso de sensores RFID, um servidor IoT e uma API.

Para futuros trabalhos, há várias áreas de melhoria e expansão possíveis:

- Sensores de Médio e Longo Alcance: Uma melhoria importante seria explorar o uso de sensores de RFID de médio e longo alcance. Isso

poderia permitir o monitoramento de objetos em uma área maior, sem que haja a necessidade de aproximar a *tag* do leitor, bastando apenas que o objeto estivesse dentro do espaço destinado na prateleira, tornando a solução ainda mais versátil em ambientes logísticos.

- Segurança e Autenticação: Embora a aplicação atual tenha fins didáticos, a segurança e autenticação são aspectos críticos em sistemas de automação logística do mundo real. Trabalhos futuros poderiam abordar a implementação de camadas de segurança, como autenticação de sensores, do acesso aos recursos da API e da criptografia de dados, para tornar o sistema apropriado para uso comercial.
- Integração com Sistemas Existentes: Uma expansão adicional envolveria a integração com sistemas de ordens de serviço ou outros sistemas logísticos já existentes em organizações. Isso permitiria que o sistema se encaixasse em ambientes empresariais.
- Melhorias de *Hardware* e *Software*: À medida que novas tecnologias surgem, melhorias contínuas no *hardware* do sensor e no *software* do sistema principal podem aprimorar o desempenho e a eficiência da solução.
- A implementação e testes de dispositivos físicos do nó sensor.

As sugestões para trabalhos futuros podem ajudar a levar essa solução acadêmica a um estágio em que ela seja aplicável em cenários do mundo real, proporcionando maior eficiência e segurança no rastreamento de objetos em ambientes logísticos.

É importante ressaltar que o sistema principal independe da tecnologia de sensor utilizada, portanto fica como sugestão a reutilização deste trabalho para implementação futura de melhorias no *hardware* e *software* do sensor, assim como do sistema principal, no que se refere a tecnologias de segurança da informação. Devido ao seu caráter didático, o sistema apresentado não oferece garantias para fins comerciais ou para qualquer outra finalidade que envolva a exposição de dados importantes.

Os conhecimentos adquiridos no curso de Tecnologia em Análise e Desenvolvimento de Sistemas foram fundamentais para o desenvolvimento do presente projeto, em especial as disciplinas relacionadas de Lógica de Programação, Banco de Dados, Engenharia de Requisitos, Redes de Computadores e Serviços de Rede. Conhecimentos novos foram adquiridos para lidar com o NodeMCU, o desenvolvimento de *endpoints* em Python e o servidor *broker* Mosquitto.

Referências

Aguiar, G. F.; Peinado, J. (2007) “Compreendendo o Kanban: Um Ensino Interativo Ilustrado”, In: da Vinci, Curitiba, v. 4, n. 1, p. 133-146, 2007.

Al-fuqaha, A. et al. (2015) “Internet of Things: A Survey on Enabling Technologies, Protocols, and Applications”, IEEE Communication Surveys & Tutorials, v. 17, n. 4, p. 2347-2376.

Arduino. Getting Started with NodeMCU ESP8266 on Arduino IDE. 2020. Disponível em: https://create.arduino.cc/projecthub/electropeak/getting-started-w-nodemcu-esp8266-on-arduino-ide-28184f. Acesso em: 10 dez. 2020.

Ashton, K. (2009) “That 'Internet of Things' thing”, RFID Journal, v. 22, n. 7, p. 97-114.

Benedetti, D.; Maselli, G. (2022) “Robust RFID Tag Identification”, Sensors, v. 21, n. 13, p. 4384.

Bouhaï, N.; Saleh, I. (2017) “Internet of Things: Evolutions and Innovations”, 1ª ed., Wiley, 240 p., ISBN: 9781786301512.

Bortoli, A. D.; Silveira, J. G.; Costa, R. (2020) “ServicesLocker: uma solução de armário inteligente baseado em Internet das Coisas”, Instituto Federal de Santa Catarina (IFSC), Lages, SC.

Buyya, R.; Dastjerdi, A. V.; Venugopal, S. (2016) “Internet of Things: Principles and Paradigms”, Elsevier.

Cosmi, A. B.; Mota, V. F. S. (2019) “Uma Análise dos Protocolos de Comunicação para Internet das Coisas”, Departamento de Informática, Universidade Federal do Espírito Santo (UFES), Vitória, ES, Brasil.

Date, C. J. (2004) “An Introduction to Database Systems”, 8ª edição, Addison-Wesley Professional.

ELMASRI, Ramez; NAVATHE, Shamkant B.; VIEIRA, Daniel (Tradutor); SERAPHIM, Enzo (Revisor). Sistemas de Banco de Dados. 7ª edição. São Paulo: Pearson, 2018.

IEEE. (2016) “IEEE P2413/D0.5. Standard for an Architectural Framework for the Internet of Things (IoT)”.

Krajewski, L. J.; Malhotra, M. K. (2007) “Operations Management: Processes and Supply Chains”, 13. ed. Global Edition, Harlow, England: Pearson.

Marzolla, M.; Mottola, L. A. (2016) “Comprehensive survey of MQTT: Features, applications, advantages and challenges”, IEEE Communications Surveys & Tutorials, v. 18, n. 3, p. 1767-1804, 2016.

Matthes, Eric (2016). Python Crash Course: A Hands-On, Project-Based Introduction to Programming. San Francisco: No Starch Press. 525p. ISBN: 9781593276034.

Mosquitto. Eclipse Mosquitto™ An open source MQTT broker. 2020. Disponível em: https://mosquitto.org/. Acesso em: 10 dez. 2020.

MySQL. Supporting Documentation. 2023. Disponível em: https://dev.mysql.com/doc. Acesso em: 29 ago. 2023.

NodeMCU. (2020) “NodeMCU Documentation”. Disponível em: https://nodemcu.readthedocs.io/en/release/. Acesso em: 11 dez. 2020.

Pallets Team. Flask Documentation. 2022. Disponível em: https://flask.palletsprojects.com. Acesso em: 14 nov. 2023.

Postman. “What is Postman?”. Disponível em: <https://www.postman.com/home\>. Acesso em: 18 de dezembro de 2023.

Qiao, S. et al. (2017) “Intelligent Refrigerator based on Internet of Things”, In: IEEE International Conference on Computational Science and Engineering (CSE) and IEEE International Conference on Embedded and Ubiquitous Computing (EUC), 2017, pp. 406-407. IEEE.

Red Hat. (2023) “O que são Application Programming Interfaces (APIs) e como funcionam”. Red Hat. Disponível em: https://www.redhat.com/pt-br/topics/api/what-are-application-programming-interfaces#como-as-apis-funcionam. Acesso em: 28 jun. 2023.

Semiconductors, N. (2014) “Mfrc522 standard 3v mifare reader solution”. Product datasheet Company public.

Sommerville, I. (2011) “Engenharia de Software”, 9. ed., São Paulo: Pearson Prentice Hall, 2011. ISBN 978-85-7936-108-1.

Tanenbaum, A. S. (2015) “Sistemas Operacionais Modernos”, p. 145-146, Ed. Pearson.

Cunha, A. F. (2006) "Etiquetas com eletrônica de ponta", Saber Eletrônica, São Paulo, v. 20, n. 401, p. 20-25, junho.

Torres, T. C. et al. (2017) “Sistema RFID de baixo custo para localização de acervo bibliográfico”, Sinergia, v. 19, n. 1, p. 81-85.

Documento Digitalizado Público

Artigo Final - Gustavo Martins Pacheco

Assunto: Artigo Final - Gustavo Martins Pacheco
Assinado por: Rodolfo Oliveira
Tipo do Documento: Anexo
Situação: Finalizado
Nível de Acesso: Público
Tipo do Conferência: Documento Digital

Documento assinado eletronicamente por:

- **Rodolfo Francisco de Oliveira**, **PROFESSOR ENS BASICO TECN TECNOLOGICO**, em 22/12/2023 16:32:29.

Este documento foi armazenado no SUAP em 22/12/2023. Para comprovar sua integridade, faça a leitura do QRCode ao lado ou acesse https://suap.ifsp.edu.br/verificar-documento-externo/ e forneça os dados abaixo:

Código Verificador: 1535099
Código de Autenticação: 4c98f93d4b