



Solução baseada em Web para Visualização de Informações no Contexto da IoT

Roger Neves de Mattos¹
Vinícius Tavares Guimaraes²
Tiago Baptista Noronha³

RESUMO

A Internet of Things (IoT) traz consigo o conceito de conectividade para uma vasta gama de dispositivos e objetos, permitindo assim que estes possam ser remotamente controlados e monitorados. Nos últimos anos, a IoT se consolidou como um dos tópicos que mais cresce na área da computação, seja no âmbito acadêmico ou na indústria. Dentre os diferentes tópicos investigados na IoT está a construção de técnicas de visualização de informações a partir da massa de dados gerada pelos diferentes dispositivos que constituem a rede. No âmbito da IoT, o emprego de técnicas de visualização da informação se tornam fundamentais, uma vez que são um dos caminhos para análise e raciocínio sobre os dados aferidos pelos dispositivos permitindo, por exemplo, identificar gargalos em cadeias de suprimento ou predizer falhas em equipamentos. Por outro lado, um número significativo de iniciativas e experimentações em IoT nascem com o desenvolvimento de protótipos por pequenos grupos de pesquisa acadêmicos ou entusiastas da área. Dessa forma, é comum que a etapa de visualização das informações oriundas das aplicações desenvolvidas seja um fator de dificuldade por três motivos principais: (i) implementar técnicas de visualização do zero pode ser complexo e, conseqüentemente, demorado; (ii) muitas vezes as equipes que estão propondo uma determinada solução em IoT não possuem humanos habilitados para o desenvolvimento de visualizações; e (iii) as soluções disponíveis ou são pagas, ou exigem uma curva de aprendizado considerável, o que também aumenta a complexidade no desenvolvimento. Considerando o cenário exposto, o presente trabalho propõe o desenvolvimento de uma ferramenta Web simples e aberta para visualização de informações no contexto da IoT. De forma geral, essa ferramenta deverá ter a capacidade de obter dados de dispositivos IoT por meio de protocolos padronizados como, por exemplo, o Message Queuing Telemetry Transport (MQTT), Constrained Application Protocol (CoAP) e Serviços Web. A partir daí disponibilizar ao usuário uma interface amigável e intuitiva para a configuração e exibição de visualizações a partir dos dados coletados dos dispositivos.

Palavras-chave: Visualização de Informações, Internet of Things, Ferramentas Web

1. INTRODUÇÃO

A *Internet of Things* (IoT) traz consigo o conceito de conectividade para uma vasta gama de dispositivos e objetos, permitindo assim que estes possam ser remotamente controlados e monitorados. Nos últimos anos, a IoT se consolidou como um dos tópicos que mais cresce na área da computação, seja no âmbito acadêmico ou na indústria [1]. Fundamentalmente, a IoT estabelece como premissa a capacidade de comunicação em praticamente tudo que envolva componentes eletrônicos (de dispositivos de comunicação a eletrodomésticos) e isso abre espaço para que seja

¹ IFSUL - Instituto Federal Sul-rio-grandense - Campus Charqueadas - Brasil.

² IFSUL - Instituto Federal Sul-rio-grandense - Campus Charqueadas - Brasil.

³ IFSUL - Instituto Federal Sul-rio-grandense - Campus Charqueadas - Brasil.

utilizada para as mais diversas funções como: medir temperatura em uma estufa que cultiva hortaliças, relógios inteligentes que monitoram as atividades físicas ou ainda monitorar sinais vitais de pessoas em tratamento de saúde.

Dentre os diferentes tópicos investigados na IoT está a construção de técnicas de visualização de informações a partir da grande massa de dados geradas pelos diferentes dispositivos que constituem a rede [2] [3]. Visualização de Informações se consolida como uma das áreas de pesquisa em Ciência da Computação e pode ser definida (em tradução livre) como o uso, apoiado por computador, de representações visuais interativas de dados abstratos com o objetivo de ampliar a cognição [4]. No âmbito da IoT, o emprego de técnicas de visualização da informação se torna fundamental, uma vez que são um dos caminhos para análise e raciocínio sobre os dados coletados dos dispositivos permitindo, por exemplo, identificar gargalos em cadeias de suprimento ou predizer falhas em equipamentos.

Nesse contexto, existem atualmente soluções proprietárias tais como o AWS da Amazon, Azure IoT da Microsoft e Tableau. Tais soluções disponibilizam diversas funcionalidades e técnicas, mas se caracterizam por modelos de negócios em que o usuário paga pelo serviço. De outro lado, tem-se as ferramentas *open source* onde destacam-se o ThingsBoard, Grafana e Kibana. No caso do ThingsBoard, tem-se uma solução com grande variabilidade de módulos agregados que vão desde o gerenciamento de dispositivos, processamento de dados e visualizações. Isso faz com que o seu uso seja mais complexo pois, em diferentes casos, é necessário que o usuário tenha alguma habilidade para, por exemplo, programar utilizando tecnologias Web, de forma a obter os dados do dispositivo e apresentá-los. No caso do Grafana e Kibana, estes necessitam de camadas de softwares intermediários que salve os dados coletados dos dispositivos em um banco de dados, tornando o processo como um todo mais oneroso para o desenvolvedor.

Essencialmente, um número significativo de iniciativas e experimentações em IoT nascem com o desenvolvimento de protótipos por pequenos grupos de pesquisa acadêmicos ou entusiastas da área. Dessa forma, é comum que a etapa de visualização das informações oriundas das aplicações desenvolvidas seja um fator de dificuldade por três motivos principais: (i) implementar técnicas de visualização do zero pode ser complexo e, conseqüentemente, demorado; (ii) muitas vezes as equipes que estão propondo uma determinada solução em IoT não possuem humanos habilitados para o desenvolvimento de visualizações; e (iii) conforme apresentado anteriormente, as soluções disponíveis ou são pagas, ou exigem uma curva de aprendizado considerável (seja pelo funcionamento da ferramenta ou pela necessidade de softwares intermediários), o que também aumenta a complexidade no desenvolvimento.

Considerando esse cenário, o presente trabalho se edifica com base na seguinte hipótese: facilitar e agilizar a construção de técnicas de visualização de informações para aplicações IoT, pode potencializar o desenvolvimento de novas propostas por acadêmicos, entusiastas e a comunidade *maker* em geral. Sendo assim, o projeto foca no desenvolvimento de uma solução Web simples e aberta para visualização de informações no contexto da IoT. De forma geral, essa ferramenta possui a capacidade de obter dados de dispositivos IoT por meio de protocolos padronizados como, por exemplo, o *Message Queuing Telemetry Transport* (MQTT), *Constrained Application Protocol* (CoAP) e Serviços Web. A partir daí, disponibilizar ao usuário uma interface amigável e intuitiva para a configuração e exibição de visualizações a partir dos dados coletados dos dispositivos

1.1 Objetivo

Projeto e desenvolvimento de uma ferramenta Web simples e aberta para facilitar e agilizar a construção de técnicas de visualização de informações em aplicações IoT, potencializando o desenvolvimento de novas propostas por acadêmicos, entusiastas e comunidade *maker* em geral.

Os objetivos específicos do trabalho são listados abaixo.

- Desenvolver um sistema capaz de obter dados de monitoramento e controle a partir de dispositivos IoT, por meio de protocolos padronizados.
- Utilizar bibliotecas para visualizar os dados no ambiente web.
- Criar um sistema em que as visualizações possam ser compartilhadas entre os usuários.
- Permitir que usuários não experts criem e configurem visualizações de informações de forma simples.
- Facilitar os testes e validações de protótipos de dispositivos IoT.

2. TRABALHOS RELACIONADOS

Inicialmente, e apresentada na Seção 2.1 a metodologia empregada para análise de trabalhos e ferramenta relacionadas. Especificamente, são apresentadas as fontes das pesquisas, quais foram as palavras-chaves e em qual período as pesquisas foram publicadas. Também explica brevemente quais foram os requisitos para selecionar as ferramentas que foram analisadas na Seção 2.3.

Na Seção 2.2 estão descritos os principais trabalhos relacionados com o tema, focando principalmente nos aspectos relacionados com visualização de informações. Finalmente, a Seção II-C aprofunda a análise das ferramentas, descrevendo quais pontos foram destacados para que os estudos fossem realizados com os *softwares* que mais se aproximam da solução proposta.

2.1 Metodologia

Em um primeiro momento foram realizadas pesquisas dentro da literatura acadêmico-científica para constatar o estado da arte no tema. A busca foi feita utilizando duas bases de pesquisa *online*: *Google scholar*⁴ e *IEEEExplore*⁵. Essa etapa do trabalho teve como objetivo entender como estão sendo conduzidas as pesquisas envolvendo o emprego de técnicas de visualização de informações no contexto da IoT.

Para executar a busca por obras relacionadas, foram utilizadas as seguintes palavras-chaves: *Internet of Things*, *Internet das Coisas*, *Visualização de Dados*, *Data Visualization*, *Dashboard*, *MQTT*. Nesse contexto, foram priorizados os artigos dos últimos cinco anos, compreendendo o período de 2015 até 2019. De forma complementar, foram analisadas duas obras consideradas referências na área [1] e [2] por conta do volume de citações e pela relevância dos veículos nas quais foram publicadas.

Após a pesquisa acadêmica, foi realizada uma segunda varredura em busca de ferramentas *open source* disponíveis no mercado para entender como funcionam, quais seus pontos fortes e fracos, e também o preço no caso das ferramentas pagas. A opção por ferramentas *open source* se dá

⁴ <https://scholar.google.com.br/>

⁵ <https://ieeexplore.ieee.org/Xplore/home.jsp>

exatamente pelo foco do presente trabalho. E de conhecimento que existem proprietárias (*e.g.*, Tableau⁶) que atendem os requisitos de visualização e já possuem *plugins* focados em IoT. No entanto, entende-se que tais ferramentas não atendem o público-alvo do presente trabalho, ou seja, iniciantes, entusiastas e movimento *maker*.

2.2 Análise da literatura acadêmica

Os trabalhos analisados na presente seção têm como intuito descrever as iniciativas na área de visualização de dados no contexto da IoT e fazer paralelização com a solução proposta.

O trabalho apresentado por Galletta *et al.* [5] propõem um novo método para visualização do status da saúde de pacientes a partir de medições advindas de diferentes fontes de dados, tais como sensores vestíveis, dispositivos IoT para análise sanguínea e equipamentos médicos. Colocando a proposta de Galletta *et al.* [5] em perspectiva com o trabalho proposto, eles convergem no entendimento sobre as potencialidades que as técnicas de visualização podem trazer para a análise e raciocínio dos dados fornecidos por dispositivos IoT.

O artigo apresentado por Nuamah e Seong [6] coloca o humano como figura central nos sistemas IoT e a interação humano-máquina como necessidade essencial e complementar, pois, segundo os autores, o computador tem a capacidade de processar grandes quantidades de dados e o ser humano habilidades mais aguçadas para detectar padrões. Com isso vem a necessidade de aplicações IoT apresente informações intuitivas e fáceis de entender.

Na pesquisa apresentada por Khalid *et al.* [7] e discutida a viabilidade de se criar um *framework* geral para visualização de informações, já que esta é uma lacuna a ser completada no âmbito de IoT. O artigo destaca que por mais que tenham pesquisas para solucionar e muito difícil cobrir toda a análise visual dos dados gerados por dispositivos IoT, por conta da diversidade de domínios que precisam de um foco específico.

Em suma, a partir das pesquisas realizadas, pode-se constatar que o mote apresentado na presente proposta ainda não foi plenamente explorado na comunidade acadêmico-científica. Pelo contrário, estudos recentes [6] [7] mostram que ainda existem muitas lacunas e oportunidades relacionando com IoT e visualização de informações.

2.3 Análise das ferramentas

As ferramentas analisadas foram ThingsBoard⁷, a ferramenta mais focada em IoT, Grafana⁸, e Kibana⁹, ambos sistemas apenas para visualização de informações. O software Kibana não foi possível ser instalado devido a ao erro "Kibana server is not ready yet", foram feitas pesquisas em fóruns, como o Elastic¹⁰, empresa responsável pelo *software*, e *stackoverflow*¹¹ para tentar corrigir o erro, porém não foi possível testar o sistema porque o erro não foi corrigido. Então os outros dois *softwares* restantes foram instalados via terminal, ambas ferramentas tem uma seção para *download* em seus

⁶ <https://ieeexplore.ieee.org/Xplore/home.jsp>

⁷ <https://thingsboard.io/>

⁸ <https://grafana.com/>

⁹ <https://www.elastic.co/pt/products/kibana>

¹⁰ <https://discuss.elastic.co/>

¹¹ <https://stackoverflow.com/>

sites, em uma máquina virtual com o sistema operacional Ubuntu 18.04 com 3Gb de memória RAM. E os dados foram enviados por um *script* que publicava dados a cada 5 minuto em um *broker* MQTT.

Devido a cada uma das ferramentas possuir particularidades tanto em questões funcionais como não-funcionais, tornou-se necessário definir um conjunto macro de itens prioritários a serem analisados em cada uma delas e posicionados em perspectiva com a solução que está sendo proposta. Foram analisados diferentes aspectos para distinguir e verificar suas vantagens e desvantagens, são eles:

- **Instalação e configuração:** Nesse ponto, pretende-se verificar se há necessidade de instalação do sistema em questão e, quando houver, quais as dificuldades para que o software esteja em funcionamento. Isso tornou-se importante para que usuários iniciantes tenham mais facilidade em ter sua utilização o mais rápido possível e se preocupar apenas com o funcionamento de seus dispositivos.
- **Protocolos da IoT:** Com o crescimento das tecnologias na área de IoT foram criados protocolos de rede específicos para serem utilizados nesse contexto e com isso as novas aplicações são geralmente arquitetadas com isso em vista. Portanto, a ferramenta ter os recursos necessários para que os dispositivos se conectem sem modificações no modo de conexão nativo dos aparelhos, torna-se um item importante a ser avaliado.
- **Armazenamento de Dados:** O objetivo desse ponto de análise é verificar se a ferramenta que está sendo avaliada permite que os dados fornecidos pelos dispositivos IoT sejam armazenados por períodos de tempo mais longo prazo. Esse aspecto se torna relevante pois dados gerados no contexto da IoT podem ser passíveis de análise no tempo, bem como dados históricos podem ser úteis tanto para caracterizar comportamentos passados como para aplicação de técnicas de predição.
- **Técnicas de visualização:** Como cada dispositivo tem uma aplicação e seus dados sem ânticas particulares. Nesse sentido, o usuário ter disponível um leque maior de opções para escolher qual forma de visualização tais dados, se adaptando com sua aplicação, e um requisito fundamental para as técnicas empregadas consiga atender cada usuário da melhor forma possível.
- **Facilidade de uso:** Dentro do contexto de IoT vem crescendo uma comunidade denominada "*Maker*", que são pessoas entusiastas na criação de dispositivos IoT, mas não são profissionais. Por isso, ter uma interface amigável e intuitiva se torna mandatório para atender esse público.

O quadro comparativo das ferramentas analisadas, em relação aos aspectos acima listados, está apresentado na Figura 1.

3. SOLUÇÃO PROPOSTA

3.1 Arquitetura da Ferramenta

O sistema será dividido em 3 partes principais, visando uma melhor organização do projeto. O *back-end*, responsável pela interação com os bancos de dados, comunicação com as outras duas partes. Os *plugins* de comunicação com os dispositivos, onde cada subsistema ficará responsável por receber os dados oriundos dos aparelhos IoT e enviá-los para o *back-end* para serem salvos, serão distribuídos em três *plugins* que farão a comunicação com um dos protocolos de rede disponíveis na ferramenta (MQTT, HTTP e CoAP). E o *front-end*, parte de interação com o usuário, encarregado de gerar as

técnicas de visualização de informações e requisitar ao *back-end* os novos dados para que os gráficos sejam atualizados com frequência sem a necessidade de a página ser recarregada. Diagrama de Componentes na Figura 2.

3.2 Plugins de comunicação

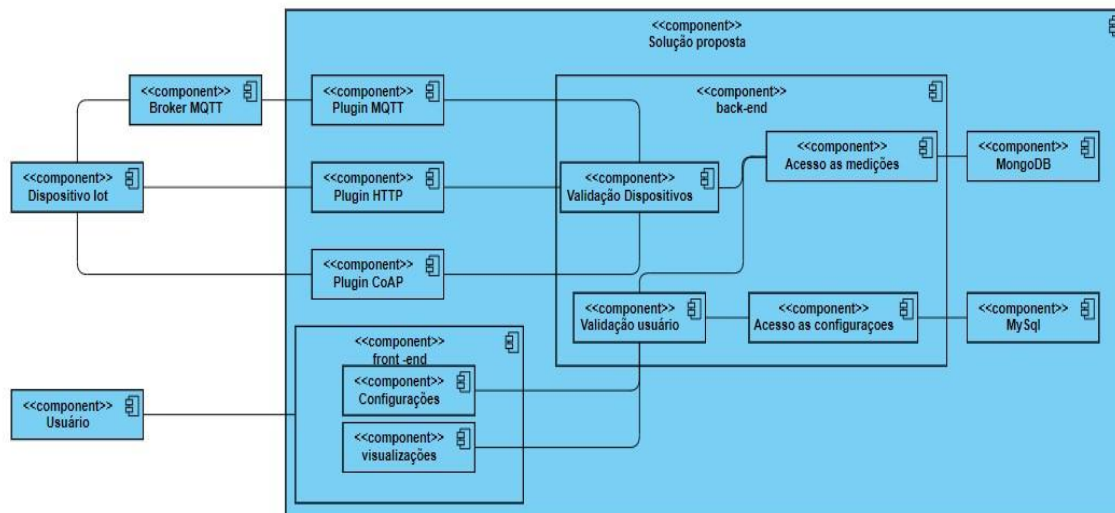
Os *plugins* de comunicação e uma parte chave da ferramenta, ela e o que diferencia de sistemas de visualização de dados, colocando a solução proposta no âmbito de IoT. Para isso esses subsistemas ficam encarregados de estabelecer a comunicação com os dispositivos através dos protocolos IoT mais utilizados e difundidos nesse meio (MQTT, HTTP e CoAP). Então para cada protocolo de rede e necessário uma maneira diferente de conexão, por esse motivo e necessário os três *plugins*.

Figura 1. Tabela comparativa.

| Pontos Analisados | ThingsBoard | Grafana | Proposta |
|---------------------------|--|--|--|
| Instalação e configuração | Processo de instalação e configuração do sistema é simples e documentado no site. Baseia-se no uso de contêiner, sendo necessário executar apenas duas instruções por linha de comando para deixar a ferramenta operacional. | Instalação fácil porque possui pacotes de instalação para as principais distribuições Linux e já está pronto para ser utilizado. | Não será necessário instalação porque o serviço será disponibilizado on-line. |
| Protocolos de rede | Tem compatibilidade com os principais protocolos utilizados no contexto de IoT (MQTT, HTTP, CoAP). | É necessário criar uma ferramenta para conectar com o protocolo desejado, pois o Grafana só se conecta ao banco de dados. | Terá compatibilidade com os principais protocolos utilizados no contexto de IoT (MQTT, HTTP, CoAP). |
| Armazenamento de dados | Utiliza-se de um banco de dados Cassandra para salvar os dados gerados por dispositivos e tem a capacidade de guardar os dados para que se tenha dados históricos dos aparelhos. | É necessário um banco de dados fora do sistema para que os dados sejam salvos, pois o software não se tem um sistema para salvar dados. | Utilizará um banco de dados <i>MongoDB</i> para coletar os dados vindos dos dispositivos e ter todo o histórico das medições feitas pelo aparelho desde do momento em que ele for cadastrado no sistema. |
| Técnicas de visualização | Não possui uma grande variedade de visualizações, mas tem as mais básicas, como gráficos em linhas, gráficos em barra, indicadores e interações com mapas. Mas tem é possível importar novas visualizações no formato JSON. | Também não possui uma grande variedade de visualizações, mas tem um espaço reservados para <i>plugins</i> que adicionam mais gráficos e <i>dashboards</i> completas. | Possuirá um grande número de visualizações de dados que será apresentado conforme os dados que serão informados pelo usuário. |
| Facilidade de uso | É um programa focado um público mais avançado e não é muito amigável para iniciantes e não tem nada no software que indique o caminho para conectar um dispositivo ou implementar um gráfico. | Tem uma interface amigável e intuitiva, o sistema já se inicia com um tutorial indicando como deve ser conectado um bando de dados e como gerar uma visualização de dados. | Será um sistema web intuitivo e amigável principalmente para iniciantes, com um tutorial quando uma conta se conectar a primeira vez para que seja algo que todos possam utilizar. |

1. **MQTT:** MQTT e um protocolo de rede que utiliza o modelo *Publisher-Subscribe*, onde os sistemas que irão comunicar-se precisam conectar-se a um *broker* onde quando houver uma publicação quem estiver inscrito recebera a mensagem. Sabendo disso, a linguagem escolhida foi para realizar essa tarefa foi o JavaScript (JS), mais especificamente NodeJS para ser utilizado no lado do servidor, pois seu funcionamento assíncrono e orientado a eventos e uma forma muito eficaz para inscrever em vários *brokers* e com um evento escutando quando algum dado for publicado.

Figura 2. Diagrama de componentes.



Com o dado recebido, o *plugin* fara uso de um pacote para JS chamado Axios, que se comporta como um cliente HTTP, consumindo recursos de um servidor, que então faz uma requisição para o servidor de *back-end* enviando os dados como parâmetro, onde ser ao salvos no MongoDB para serem utilizados posteriormente.

2. **HTTP:** O protocolo HTTP utiliza-se da estrutura cliente servidor, onde o cliente faz requisições ao servidor, enviando os dados como parâmetro e recebendo uma resposta assim que o servidor processar a requisição. Para fazer esse *plugin* será utilizado o modelo Representational State Transfer (REST), utilizando o *framework Express* para NodeJS, utilizado para tratar rotas. Após o dado enviado ele também ser a enviado ao *back-end* a partir do pacote Axios.
3. **CoAP:** O protocolo CoAP se assemelha muito ao HTTP, porque também utiliza o modelo cliente-servidor e a arquitetura REST, portanto será necessário apenas a adaptação para o pacote coap ao invés do Express. Quando o servidor receber dados, fara o mesmo procedimento dos outros de enviar ao *back-end* para ser validado e salvo.

3.3 Back-end

O *back-end* será feito com base no modelo REST, pois esse modelo assume a responsabilidade de todas as interações com os bancos de dados, isso facilitara todas as validações quando houver requisições, com isso tanto os acessos oriundos dos dispositivos quanto os vindo do usuário ser ao tradados no mesmo lugar, evitando, assim, a reescrita de código e o sobrecarregamento do sistema. A linguagem escolhida para isso foi JavaScript mais especificamente NodeJS, porque essa linguagem, além de ser utilizada em outras partes do projeto deixando-o mais homogêneo, quando utilizada com o *framework Express* se torna muito boa para o tratamento de rotas e também tem ótimos pacotes para realizar as validações necessárias para o controle de acesso aos bancos de dados.

1. **Bancos de dados:** O banco de dados implementado no sistema para as configurações de usuário ser a MySQL, pois como os dados que ele salvara ser aos padrões. A utilização do MySQL dará um controle maior sobre a manipulação dos dados e a capacidade de busca em várias tabelas ao mesmo tempo reduz o número de requisições que precisar ao ser feitas ao

servidor durante o acesso dos usuários. Então a maioria das partes do sistema será salva em um banco MySQL, ele guardará todos os dados de usuários, dispositivos e configurações do sistema.

O segundo banco de dados utilizado no sistema será o MongoDB, um banco NoSQL que possui uma boa escalabilidade e não necessita de um modelo relacional prévio para funcionar, além de ter uma ótima integração com NodeJS através do pacote Mongoose. Por esses motivos ele foi escolhido para salvar os dados oriundos das medições feitas pelos dispositivos IoT, porque esses dados terão um volume muito maior do que qualquer outra parte do sistema, e como cada dispositivo terá uma entrada de dados diferentes a possibilidade de não haver um modelo relacional deixa aberto para quais forem os dados enviado a certeza de que serão salvos sem chance de dar erro no momento em que forem inseridos no banco de dados.

3.4 Front-end

Para o *front-end*, será utilizado *React* uma biblioteca para *JavaScript* que auxilia a criação de interfaces interativas para web, utiliza a mecânica de componentes, que consiste a em encapsular códigos de *html* e *JavaScript* onde podem ser requisitados e combinados de qualquer forma, isso dá a possibilidade de reutilizar uma grande quantidade de código. Esta biblioteca também tem a vantagem de ter uma fácil atualização de uma forma fácil e apenas nos pontos onde são necessários, isso faz com que os dados sejam atualizados nos sem a necessidade de recarregar a página. Combinando com *React Router* ainda temos uma navegação pelas páginas de forma fluida, também sem a necessidade de recarregamento da página no navegador, dando uma aparência de aplicativo e não um site comum, mas ainda com modificação da URL.

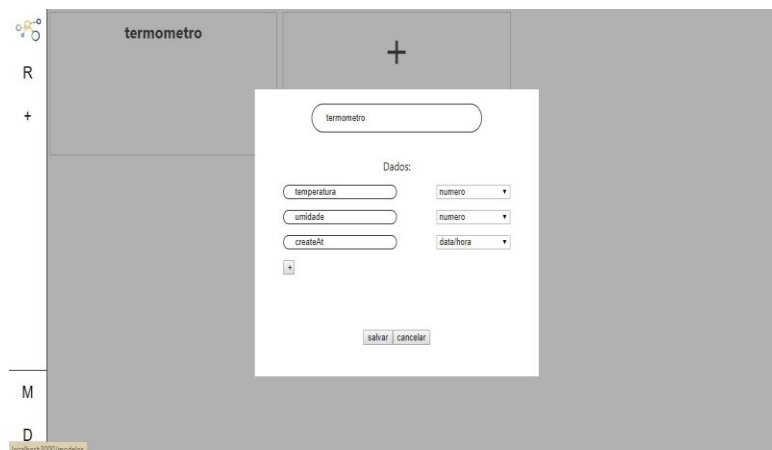
4. RESULTADOS E DISCUSSOES

A ferramenta se utiliza de três elementos para funcionar: modelo, dispositivo e *dashboard*. É necessário ter as três partes do sistema para ser possível visualizar dados com ele, também é necessário serem adicionadas pelo usuário em uma ordem específica. Primeiramente é adicionado o modelo, que descreve como será a tipagem dos dados de um dispositivo. Logo após o dispositivo onde será indicado em qual o protocolo de rede utilizado e seus parâmetros, assim como qual o modelo que o dispositivo irá seguir. Por último é configurado o *dashboard* onde são adicionados as visualizações, informando o dispositivo e seu dados que comporão a visualização em questão.

4.1 Modelo

O modelo tem a intenção de reaproveitar configuração, fazendo com que o usuário possa utilizar o mesmo modelo para vários dispositivos. Também tem a função de validar os itens oriundos dos dispositivos, evitando, assim, a entrada de dados que não foram indicados no modelos, pois os mesmos não podem constituir as visualizações. Um exemplo do formulário de modelos na Figura 3.

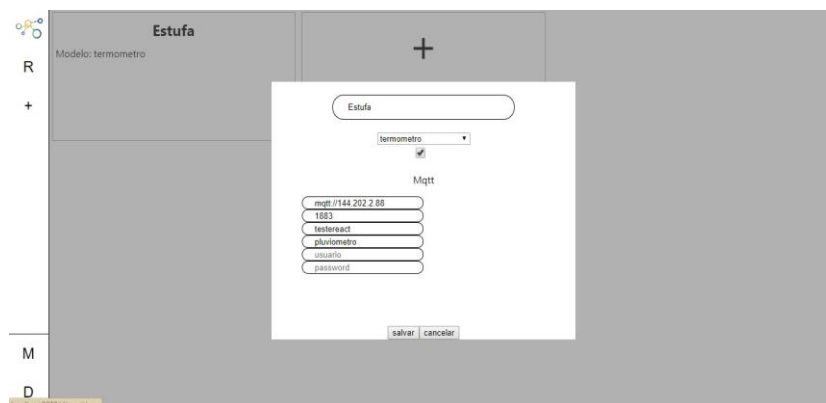
Figura 3. Tela de gerenciamento de modelos.



4.2 Dispositivo

Os dispositivos ficam com a responsabilidade de configurar como será feita a entrada de dados, no momento estão disponíveis os protocolos MQTT e HTTP, e indicando seu *token* de acesso do dispositivo, para que ele seja validado quando for enviado e o sistema aceite a entrada de seus dados. Um exemplo do formulário de dispositivos na Figura 4.

Figura 4. Tela de gerenciamento de dispositivos.



4.3 Dashboard

Item principal da ferramenta, onde as visualizações são gerenciadas e é possível ver os dados gerados. Inicialmente dando a opção de 4 visualizações: *gauge*, gráfico de linha, gráfico de barras e gráfico radar. Deixando o usuário escolher qual lhe agrada mais e se encaixa melhor com o formato do dado utilizado. Um exemplo da página de Dashboard na Figura 5.

Figura 5. Tela de gerenciamento de dashboards.



5. CONCLUSÕES

O desenvolvimento chegou a seu objetivo inicial de facilitar a visualização de dados no contexto de IoT, conseguindo funcionar com dois dos principais protocolos de rede que são utilizados na área.

Os passos seguintes serão aumentar a gama de visualizações disponíveis na ferramenta, proporcionando mais opções ao usuário. Também ser a melhorado os envio dos dados para as visualizações, que no momento e feito através de requisições HTTP com intervalos de 1 segundo, passando a utilizar *webSockets* que evitara uma grande quantidade de processamento de requisições ao *back-end*.

6. REFERÊNCIAS

- [1] L. Atzori, A. Iera, and G. Morabito, "**The internet of things: A survey**," 2010, vol. 54, no. 15, pp. 2787 – 2805. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1389128610001568>
- [2] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "**Internet of things (iot): A vision, architectural elements, and future directions**," 2013, vol. 29, no. 7, pp. 1645 – 1660, including Special sections: Cyber-enabled Distributed Computing for Ubiquitous Cloud Network Services Cloud Computing and Scientific Applications — Big Data, Scalable Analytics, and Beyond. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S0167739X13000241>
- [3] P. Ray, "**A survey on internet of things architectures**," 2018, vol. 30, no. 3, pp. 291 – 319. [Online]. Available: <http://www.sciencedirect.com/science/article/pii/S1319157816300799>
- [4] S. K. Card, J. Mackinlay, and B. Shneiderman, Readings in Information Visualization: Using Vision to Think, 1st ed. San Diego, CA, USA: Academic Press, 1999.
- [5] A. Galletta, L. Carnevale, A. Bramanti, and M. Fazio, "**An innovative methodology for big data visualization for telemedicine**," Jan 2019, vol. 15, no. 1, pp. 490–497.
- [6] J. Nuamah and Y. Seong, "Human machine interface in the internet of things (iot)," 06 2017, pp. 1–6.
- [7] M. Y. Khalid, P. H. H. Then, and V. Raman, "**Exploratory study for data visualization in internet of things**," in 2018 IEEE 42nd Annual Computer Software and Applications Conference (COMPSAC), July 2018, vol. 02, pp. 517–521.