



---

# Relatório de Finalização de Estágio

*Laboratório IoT: Internet das Coisas e Indústria 4.0*

---

Anderson Mendrot

David Vicente

Fabício Costa Souza

Lucas Bicalho Oliveira

Dezembro, 2019

# Sumário

<b>1</b>	<b>ATIVIDADES</b>	<b>3</b>
<b>1.1</b>	<b>Site do projeto</b>	<b>3</b>
<b>1.2</b>	<b>Aplicação Web de Monitoramento usando Shiny</b>	<b>5</b>
1.2.1	Logs da impressora 3D Objet30 Pro	5
1.2.2	<i>Datasets</i> da Nasa	7
1.2.3	Desenvolvimento de Banco de Dados	8
1.2.4	Implementação em R Shiny	9
<b>1.3</b>	<b>Análise de dados com Aprendizado de Máquina</b>	<b>12</b>
<b>1.4</b>	<b>Estudo sobre desenvolvimento de dashboards IoT</b>	<b>18</b>
<b>1.5</b>	<b>Funcionalidades para o SmartMonitor</b>	<b>18</b>
1.5.1	Banco de dados	19
1.5.2	APIs de acesso e <i>endpoints</i>	19
1.5.3	Buffer	20
1.5.4	API-Key	21
1.5.5	Testes	21
1.5.6	<i>Docker</i>	21
<b>1.6</b>	<b>Projeto de Sensoriamento Wi-Fi</b>	<b>22</b>
<b>1.7</b>	<b>Artigos e relatório técnico</b>	<b>23</b>
1.7.1	Artigo sobre Redes IoT	23
1.7.2	Artigo sobre Casos de uso	23
1.7.3	Revisão da Literatura e Artigo sobre Manutenção preditiva na indústria	23
<b>1.8</b>	<b>Projeto Executivo</b>	<b>24</b>
<b>1.9</b>	<b>Eventos e Visitas Técnicas</b>	<b>24</b>
1.9.1	Eventos	24
1.9.2	Visitas técnicas	25

# 1 Atividades

## 1.1 Site do projeto

Com a intenção de ter um ambiente de visualização de informações sobre o projeto, foram elaboradas algumas versões de websites. Inicialmente, foram feitas versões iniciais utilizando-se as ferramentas de criação de site *Wix* e *Google Sites*.

Posteriormente, como forma de padronizar o *layout* de acordo com o site oficial do Parque Tecnológico, as informações foram adaptadas dessa forma. Para isso, foram obtidos os códigos nas linguagens *HTML*, *CSS* e *JavaScript* relativos ao site principal e, a partir deles, inseriu-se os textos e figuras relativos ao projeto e criou-se algumas páginas com funcionalidades específicas.

A Figura 1 apresenta a página inicial do site, em que é descrita uma visão geral do projeto, assim como objetivos e atividades realizadas.



Figura 1 – Página inicial do site.

Também foram criadas outras cinco páginas, que podem ser acessadas por meio das

abas do menu:

- Sobre o projeto

Apresenta a missão geral do Parque Tecnológico e do projeto em si, descrevendo os principais objetivos e informações sobre a "Chamada pública do ministério da indústria, comércio exterior e serviços para a criação de plataformas de experimentação", em que o projeto do Parque Tecnológico de São José dos Campos foi aprovado em primeiro lugar.

- Artigos

Foram elaborados, pela equipe do projeto, três artigos relacionados ao tema de Internet das Coisas e Indústria 4.0, os quais são descritos na seção 1.7, mais adiante. A Figura 2 apresenta a página na qual há referência para os artigos elaborados.



Figura 2 – Página de artigos do site.

- Quem somos

Informa os nomes e páginas de perfil do *LinkedIn* dos colaboradores do projeto.

- Equipamentos

Apresenta informações sobre as máquinas sensoriadas no projeto.

- Empresas e Startups parceiras

Apresenta as empresas e startups parceiras e empresas patrocinadoras do projeto.

Todas as alterações nos códigos implementados foram primeiro testadas localmente e, então, substituídas no servidor do Parque Tecnológico.

## 1.2 Aplicação Web de Monitoramento usando Shiny

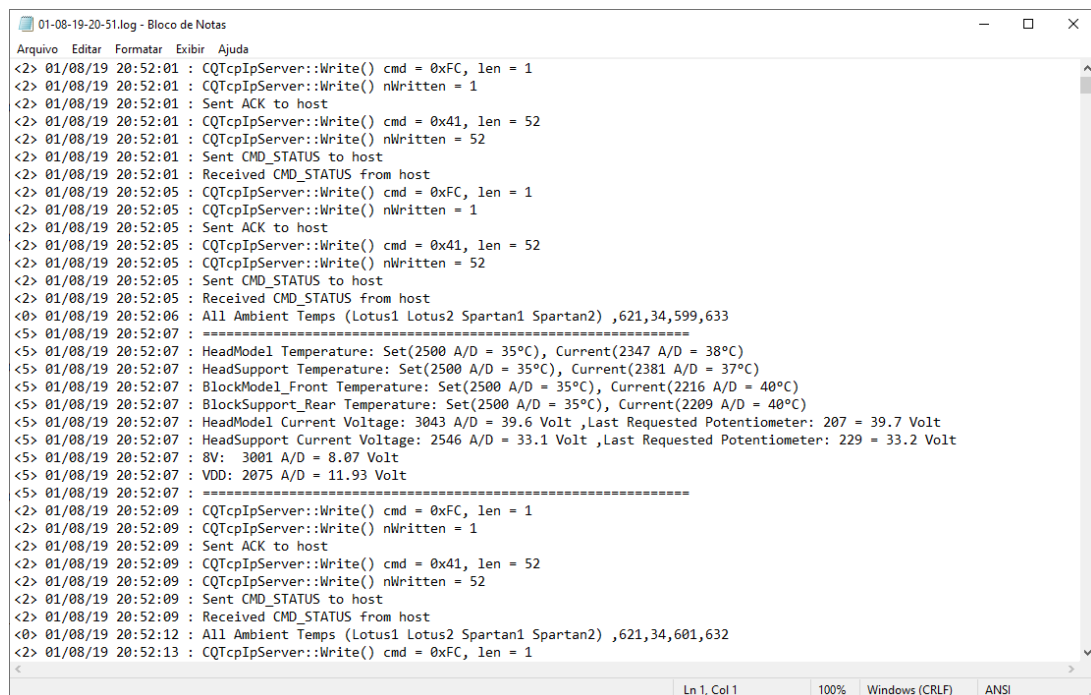
Esta seção explica como se deu a implementação da plataforma em Shiny, descrevendo a obtenção dos logs, *datasets* e a implementação utilizando Shiny.

### 1.2.1 Logs da impressora 3D Objet30 Pro

A análise dos dados das máquinas instrumentadas seria o passo seguinte após a finalização da implantação do *testbed*, no entanto, dado aos processos que ainda estavam ocorrendo de aquisição dos sensores, módulos e controladores, bem como a finalização de outros processos burocráticos, a ideia era fazer uma revisão da literatura sobre aprendizado de máquina e manutenção preditiva.

Os dados utilizados nos tutoriais, para ser de acordo com o que estávamos trabalhando, foram extraídos da impressora 3D Objet30 Pro do Laboratório de Manufatura Digital e Prototipagem Virtual do Parque Tecnológico e se refere à algumas impressões.

Como esses logs extraídos estavam no formato de texto (Figura 3), foi necessário implementar uma função, usando a linguagem de programação Python, que convertia o formato log para o formato em planilha CSV, contendo ao final 9 variáveis.



```
01-08-19-20-51.log - Bloco de Notas
Arquivo Editar Formatar Exibir Ajuda
<2> 01/08/19 20:52:01 : CQTcpIpServer::Write() cmd = 0xFC, len = 1
<2> 01/08/19 20:52:01 : CQTcpIpServer::Write() nWritten = 1
<2> 01/08/19 20:52:01 : Sent ACK to host
<2> 01/08/19 20:52:01 : CQTcpIpServer::Write() cmd = 0x41, len = 52
<2> 01/08/19 20:52:01 : CQTcpIpServer::Write() nWritten = 52
<2> 01/08/19 20:52:01 : Sent CMD_STATUS to host
<2> 01/08/19 20:52:01 : Received CMD_STATUS from host
<2> 01/08/19 20:52:05 : CQTcpIpServer::Write() cmd = 0xFC, len = 1
<2> 01/08/19 20:52:05 : CQTcpIpServer::Write() nWritten = 1
<2> 01/08/19 20:52:05 : Sent ACK to host
<2> 01/08/19 20:52:05 : CQTcpIpServer::Write() cmd = 0x41, len = 52
<2> 01/08/19 20:52:05 : CQTcpIpServer::Write() nWritten = 52
<2> 01/08/19 20:52:05 : Sent CMD_STATUS to host
<2> 01/08/19 20:52:05 : Received CMD_STATUS from host
<0> 01/08/19 20:52:06 : All Ambient Temps (Lotus1 Lotus2 Spartan1 Spartan2) ,621,34,599,633
<5> 01/08/19 20:52:07 : =====
<5> 01/08/19 20:52:07 : HeadModel Temperature: Set(2500 A/D = 35°C), Current(2347 A/D = 38°C)
<5> 01/08/19 20:52:07 : HeadSupport Temperature: Set(2500 A/D = 35°C), Current(2381 A/D = 37°C)
<5> 01/08/19 20:52:07 : BlockModel_Front Temperature: Set(2500 A/D = 35°C), Current(2216 A/D = 40°C)
<5> 01/08/19 20:52:07 : BlockSupport_Rear Temperature: Set(2500 A/D = 35°C), Current(2209 A/D = 40°C)
<5> 01/08/19 20:52:07 : HeadModel Current Voltage: 3043 A/D = 39.6 Volt ,Last Requested Potentiometer: 207 = 39.7 Volt
<5> 01/08/19 20:52:07 : HeadSupport Current Voltage: 2546 A/D = 33.1 Volt ,Last Requested Potentiometer: 229 = 33.2 Volt
<5> 01/08/19 20:52:07 : 8V: 3001 A/D = 8.07 Volt
<5> 01/08/19 20:52:07 : VDD: 2075 A/D = 11.93 Volt
<5> 01/08/19 20:52:07 : =====
<2> 01/08/19 20:52:09 : CQTcpIpServer::Write() cmd = 0xFC, len = 1
<2> 01/08/19 20:52:09 : CQTcpIpServer::Write() nWritten = 1
<2> 01/08/19 20:52:09 : Sent ACK to host
<2> 01/08/19 20:52:09 : CQTcpIpServer::Write() cmd = 0x41, len = 52
<2> 01/08/19 20:52:09 : CQTcpIpServer::Write() nWritten = 52
<2> 01/08/19 20:52:09 : Sent CMD_STATUS to host
<2> 01/08/19 20:52:09 : Received CMD_STATUS from host
<0> 01/08/19 20:52:12 : All Ambient Temps (Lotus1 Lotus2 Spartan1 Spartan2) ,621,34,601,632
<2> 01/08/19 20:52:13 : CQTcpIpServer::Write() cmd = 0xFC, len = 1
```

Figura 3 – Exemplo do formato dos logs da impressora 3D.

No geral, as variáveis extraídas desses logs eram referentes às temperaturas do cabeçote e suporte (configurada e a real).

Em especial, a variável Status, embora nem sempre era reportada no log, indicava condições de aquecimento da impressora, não sendo necessariamente referente ao mal funcionamento da mesma. A Figura 4 apresenta alguns boxplot's do Status em função das variáveis de temperatura real da máquina.

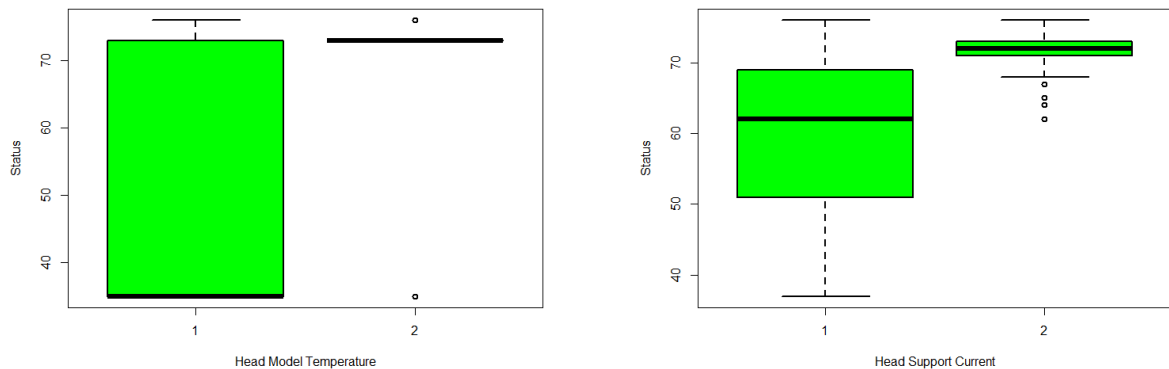


Figura 4 – Boxplot's do Status x Temperaturas

A Figura 5 indica a série temporal de todas as impressões nos logs obtidos.

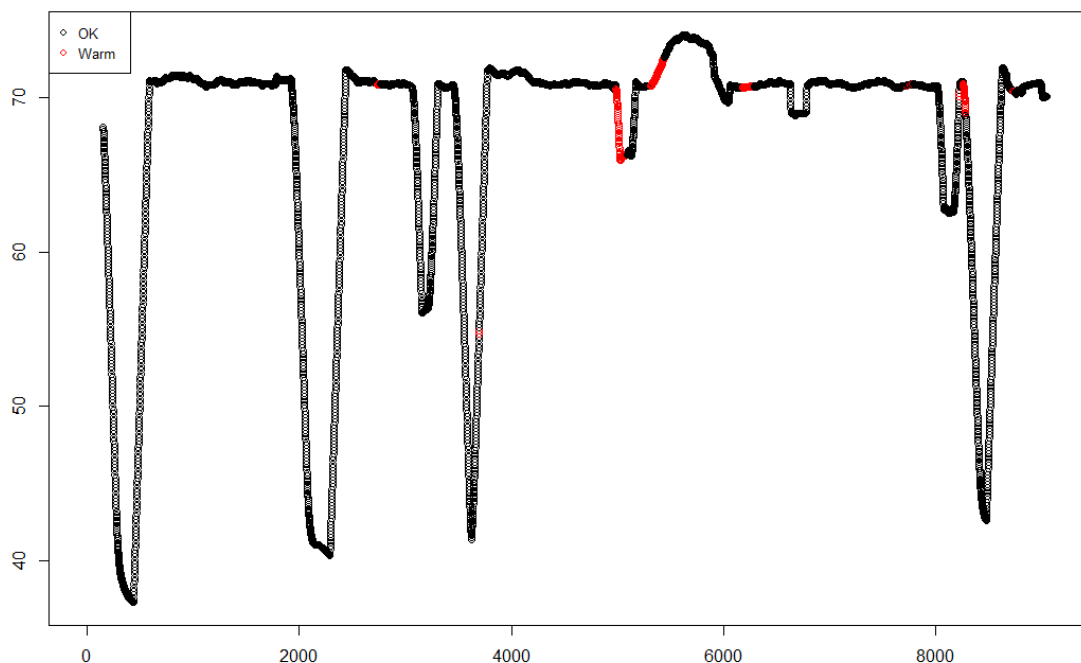


Figura 5 – Série temporal da temperatura do cabeçote da impressora 3D.

Com essa base de dados produzida, foi possível estudar alguns algoritmos de aprendizado de máquina, esses nos quais será falado mais adiante.

### 1.2.2 *Datasets da Nasa*

Para propósitos de testes com a plataforma e para uso em Aprendizado de Máquina, foram utilizados *datasets* pertencentes a NASA, mais especificamente, da seção *Prognostics Data Repository* (Repositório de dados prognósticos, em tradução livre). Tal seção contém *datasets* focados principalmente em séries temporais que vão desde um determinado estado até um estado de falha. O repositório está disponível em <https://ti.arc.nasa.gov/tech/dash/groups/pcoe/prognostic-data-repository>. A descrição e a lista de variáveis de cada *dataset* utilizada se encontra abaixo.

- *Small Satellite Power Simulation*

**Descrição:** Dados coletados a partir de experimentos simulados em pequenas baterias de satélite BP930 usando o sistema MACCOR.

**Variáveis utilizadas:** identificador (idsmallsatellite), tipo (type), tempo (time), tempo relativo (relativeTime), tensão (voltage), corrente (current), temperatura (temperature), data (date), capacidade (capacity), energia (energy), comentários (comment)

- *Randomized Battery Usage Data Set*

**Descrição:** Dados obtidos a partir de baterias que sofrem ciclos contínuos com perfis de corrente gerados aleatoriamente. Os ciclos de carga e descarga das baterias também são realizados após intervalos fixos que ocorrem em tempos aleatórios, a fim de fornecer parâmetros para o estado de funcionamento da bateria.

**Variáveis utilizadas:** identificador (idbattery), tipo (type), tempo (time), tempo relativo (relativeTime), tensão (voltage), corrente (current), temperatura (temperature), data (date), comentários (comment)

- *Battery Data Set*

**Descrição:** Dados obtidos após experiências com baterias de íon de lítio. A carga e descarga das baterias ocorrem em diferentes temperaturas.

**Variáveis utilizadas:** identificador (idbattery), tipo (type), tempo (time), tempo relativo (relativeTime), tensão (voltage), corrente (current), temperatura (temperature), data (date), comentários (comment)

- *HIRF Battery Data Set*

**Descrição:** Dados da bateria coletados das experiências nas aeronaves Edge 540 em uma câmara HIRF.

**Variáveis utilizadas:** time (t), revolutions per minute (RPM), forward motor controller sensor (FMC), after motor controller sensor (AMC), battery voltage of lower left front (LLF20V), battery voltage of upper left after (ULA20V)

### 1.2.3 Desenvolvimento de Banco de Dados

O sistema de gerenciamento de banco de dados utilizado para a implementação do banco de dados foi o MySQL. Para tanto, foi desenvolvido um diagrama que representa o banco utilizado pela plataforma em Shiny.

O banco contém tabelas para todos os quatro *datasets* utilizados, além de uma tabela para os logs da impressora 3D Objet30 Pro. Em relação aos *datasets*, eles foram processados de forma que ficassem em formato csv, o que permitiu que fossem importados para o MySQL. O mesmo se aplicou aos logs da impressora, os quais foram editados com o uso de um script em Python desenvolvido para formatação de dados brutos, em txt, para csv.

Todos os conjuntos de dados listados anteriormente se ligam a uma tabela *machine*, a qual representa de forma geral as máquinas utilizadas na plataforma. Além disso, foi criada uma tabela *usuario*, de forma a relacionar máquinas a usuários da plataforma. O diagrama para o bando descrito e encontra na Figura 6.

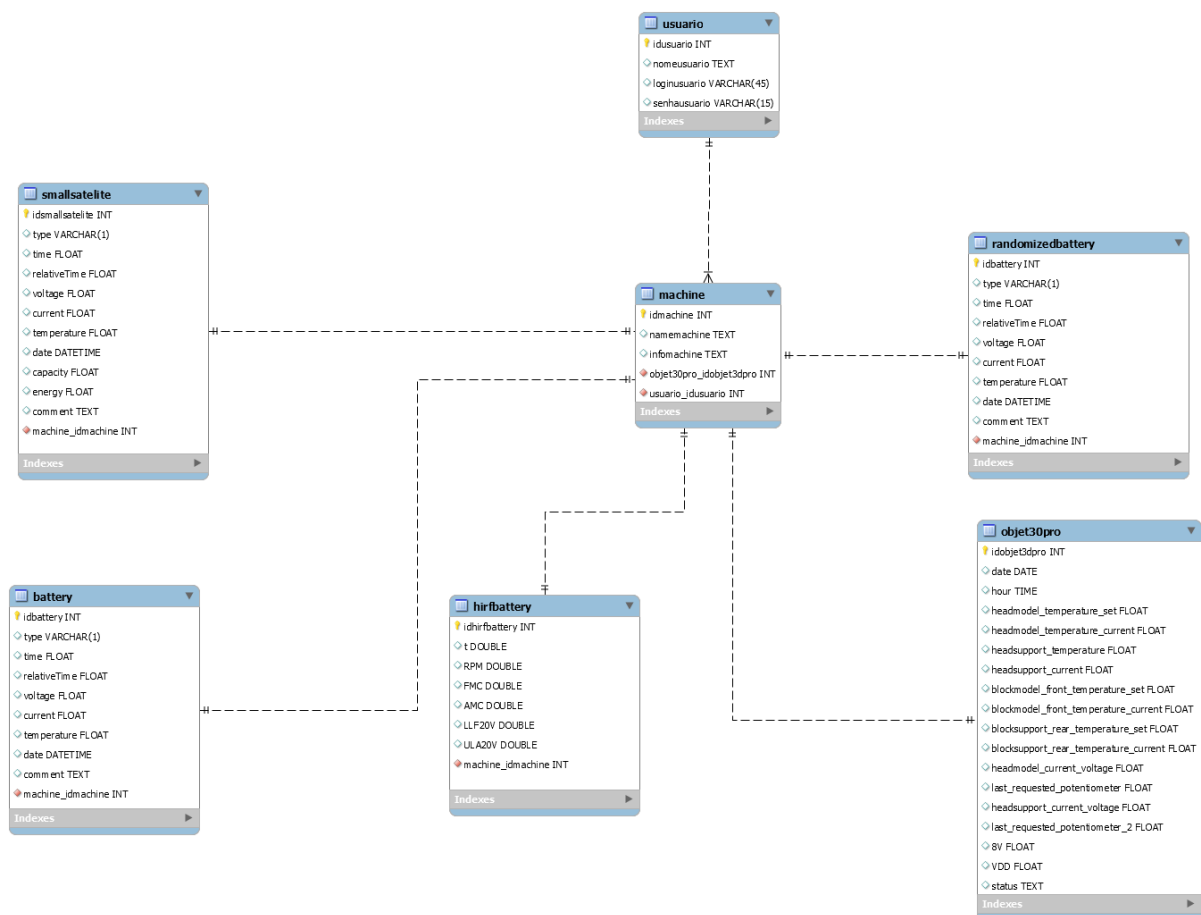


Figura 6 – Modelo relacional do banco de dados



## 1.2.4 Implementação em R Shiny

O Shiny é um pacote R para o desenvolvimento de aplicações interativas utilizando a linguagem R juntamente com CSS, HTML e JavaScript. Embora o conhecimento em R apenas seja suficiente para gerar uma aplicação funcional, o usuário tem a liberdade de modificar e/ou adicionar o *template* oferecido pelo pacote com base em seus conhecimentos em HTML e CSS, principalmente.

Usando esse pacote, foi possível desenvolver uma aplicação web conectada ao banco MySQL modelado com suas funcionalidades focadas na apresentação gráfica dos dados e inteligência artificial.

Além disso, a função que convertia o arquivo log, das impressoras 3D, para CSV foi implementada em R e integrada à esse sistema, dando a ela a possibilidade de analisar não apenas os dados dos sensores, como também das informações oferecidas pela máquina.

A Figura 7 apresenta o visual da página inicial. Vale ressaltar que não foi implementada essa funcionalidade de logar no sistema.

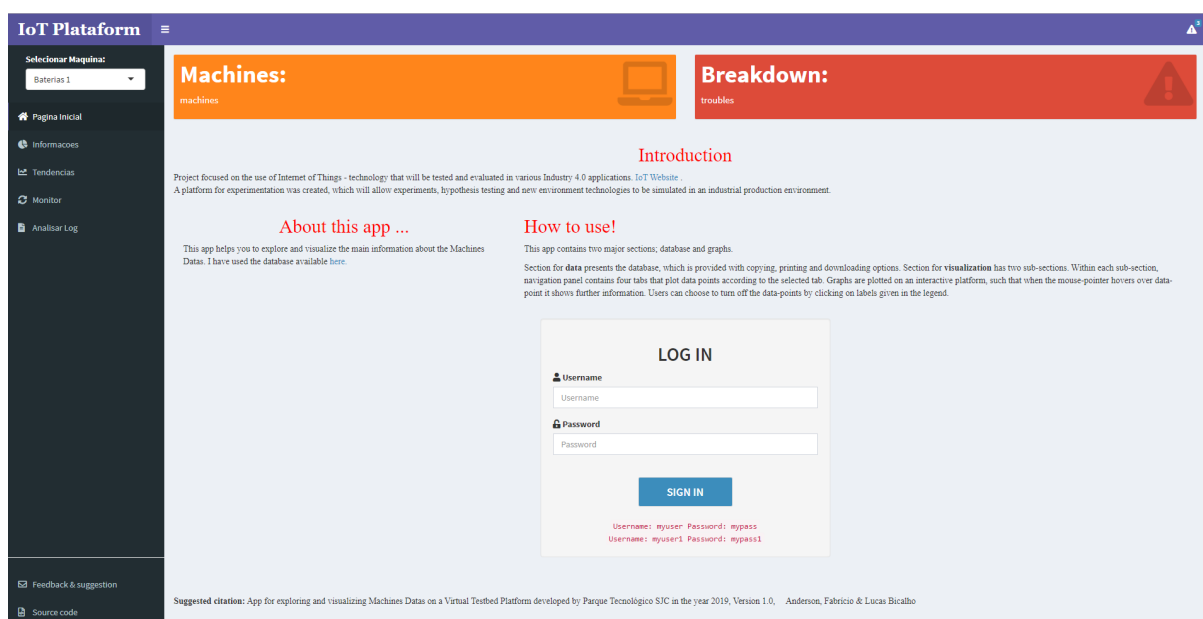


Figura 7 – Página Inicial da aplicação Shiny.

A Figura 8 apresenta a aba que diz respeito à visualização gráfica por um período fixo, a ideia era inserir um menu que o usuário indicasse o período da série temporal e assim apresentar diversos tipos de gráficos para um ou mais parâmetros, pois a finalidade dessa aba é dar diversos recursos para o próprio usuário analisar seus dados.

Apenas as funcionalidades referentes à conexão com o banco de dados e apresentação desses dados foi inserida no sistema.

A Figura 9 apresenta a aba referente à obtenção de dados brutos para *download* de dados. Nessa aba, foi implementada apenas a funcionalidade de fazer requisição no banco para os dados desejados e fazer o *download* do mesmo.

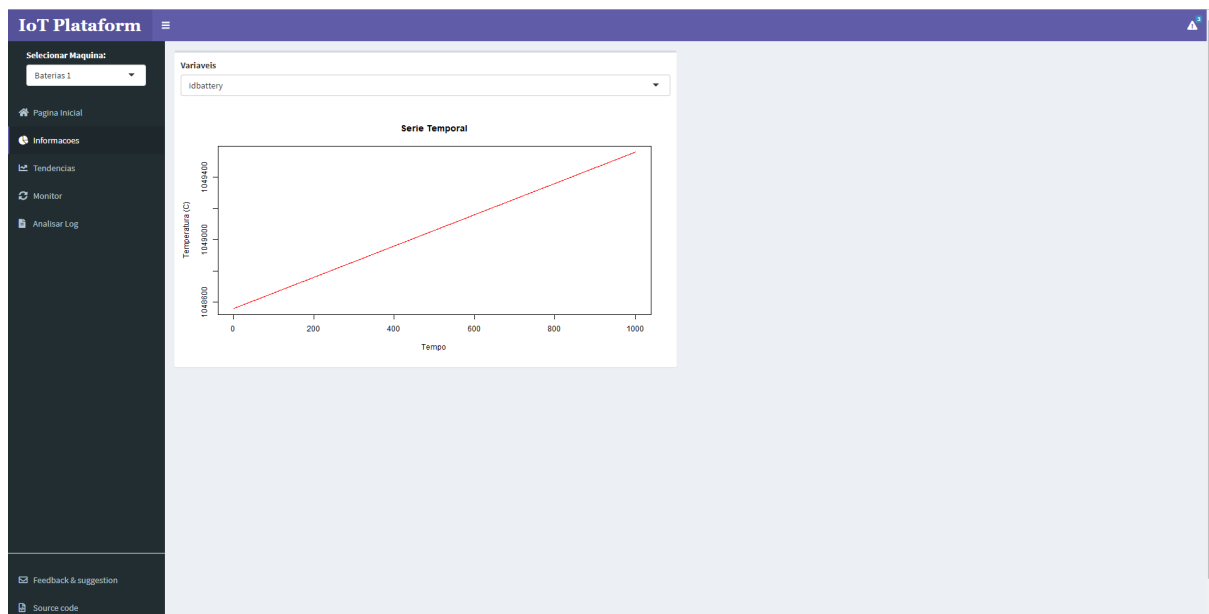


Figura 8 – Página Informações da aplicação Shiny.

A finalidade dessa aba é dar recursos para o usuário fazer o *download* apenas das informações que tem interesse, bem como o formato do banco desejado para suas análises.

	idbattery	type	time	relativeTime	voltage	current	temperature	date	comment	name_machine
1	1048561	C	5.04	0.04	3.838	-2.007	243.853	06/01/2014 13:36	reference charge	
2	1048562	C	15.04	10.04	3.865	-2	243.853	06/01/2014 13:36	reference charge	
3	1048563	C	25.04	20.04	3.878	-2	242.916	06/01/2014 13:36	reference charge	
4	1048564	C	35.04	30.04	3.888	-2	242.448	06/01/2014 13:36	reference charge	
5	1048565	C	45.04	40.04	3.895	-2	242.604	06/01/2014 13:36	reference charge	
6	1048566	C	55.04	50.04	3.9	-2	242.292	06/01/2014 13:36	reference charge	
7	1048567	C	65.04	60.04	3.905	-2	24.198	06/01/2014 13:36	reference charge	
8	1048568	C	75.04	70.04	3.908	-2	241.667	06/01/2014 13:36	reference charge	
9	1048569	C	85.04	80.04	3.911	-2	241.199	06/01/2014 13:36	reference charge	
10	1048570	C	95.04	90.04	3.914	-2	241.667	06/01/2014 13:36	reference charge	
11	1048571	C	105.04	100.04	3.916	-2	242.136	06/01/2014 13:36	reference charge	
12	1048572	C	115.04	110.04	3.918	-2	242.136	06/01/2014 13:36	reference charge	
13	1048573	C	125.04	120.04	3.92	-2	242.136	06/01/2014 13:36	reference charge	
14	1048574	C	135.04	130.04	3.922	-2.001	242.292	06/01/2014 13:36	reference charge	
15	1048575	C	145.04	140.04	3.925	-2	242.448	06/01/2014 13:36	reference charge	
16	1048576	C	155.04	150.04	3.927	-2.001	242.604	06/01/2014 13:36	reference charge	
17	1048577	C	165.04	160.04	3.929	-2	242.292	06/01/2014 13:36	reference charge	
18	1048578	C	175.04	170.04	3.931	-2	24.276	06/01/2014 13:36	reference charge	
19	1048579	C	185.04	180.04	3.933	-2	242.916	06/01/2014 13:36	reference charge	
20	1048580	C	195.04	190.04	3.936	-2	242.916	06/01/2014 13:36	reference charge	

Figura 9 – Página Tendências da aplicação Shiny.

A Figura 10 apresenta a aba Monitor, na qual se refere à informações referentes ao estado corrente da máquina, em tempo real, e informações da Inteligência Artificial como chances de erro e vida útil, por exemplo. Nessa aba, a ideia é oferecer ao usuário os recursos do comportamento corrente da máquina.

Por fim, a Figura 11 apresenta a aba de análise de logs, na qual foi inserida a função implementada para impressora Objet30 Pro.

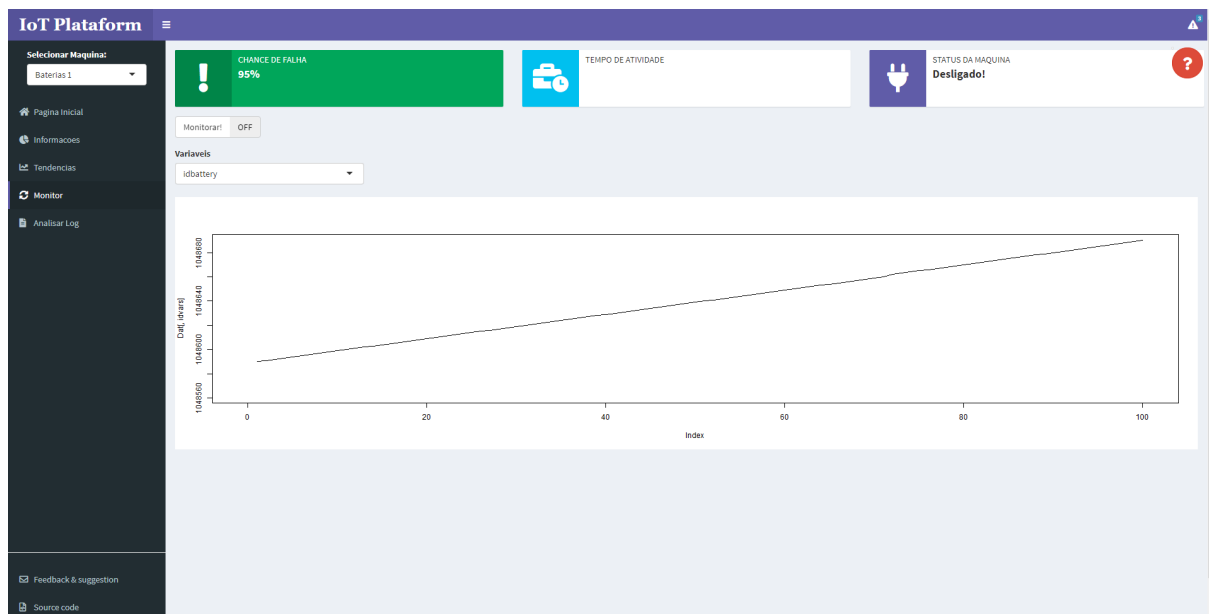


Figura 10 – Página Monitor da aplicação Shiny.

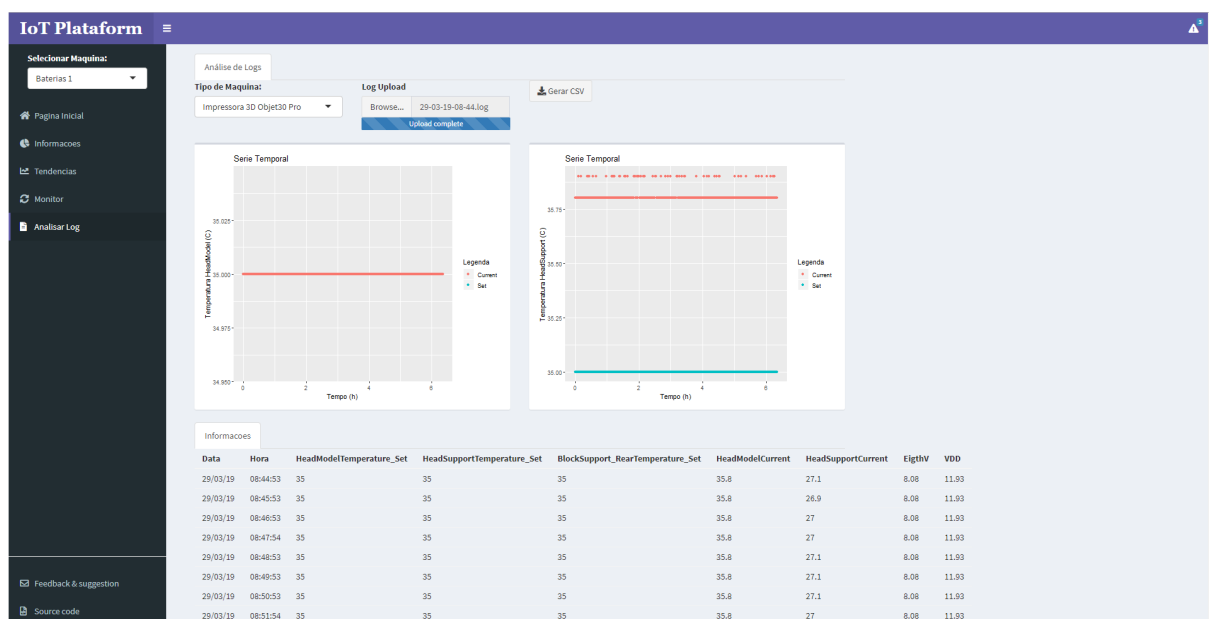


Figura 11 – Página Analisar Log's da aplicação Shiny.

Esses foram os requisitos principais pensados para o funcionamento da aplicação web. Cabe ressaltar que nem todas foram integradas/implementadas no sistema.

O projeto contém, basicamente, três pastas (Pages, Providers e www) e três códigos em R, sendo eles app, globalVars e ui. A pasta page contém os arquivos para as respectivas abas da plataforma, nelas é possível encontrar suas respectivas funções e estruturas web que são utilizadas no Shiny, já em Providers tem-se funções para manipulação no banco de dados, gráficos e filtros (Figura 12). Em app está a função *server* que trata de requisições reativas e de *input/output*.

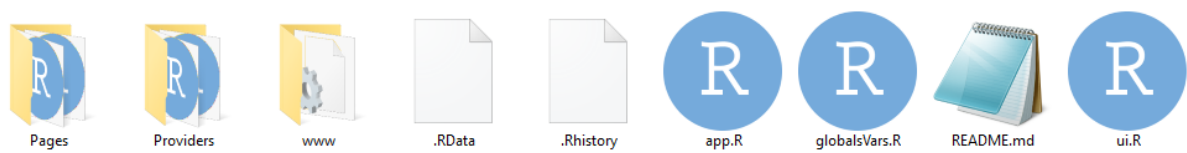


Figura 12 – Arquivos da aplicação web em Shiny

Pode-se dizer que após os conhecimentos adquiridos no desenvolvimento da API, essa plataforma poderia ser refeita com diversas melhorias, inclusive de programação, como por exemplo eliminando o trabalho do Shiny de se comunicar diretamente com o banco, deixando isso para a API em Django, e se preocupar mais com a representação gráfica e estruturas interativas para a visualização e exportação de dados.

### 1.3 Análise de dados com Aprendizado de Máquina

Com base nos algoritmos vistos em artigos de manutenção preditiva e aqueles principais em Aprendizado de Máquina, foram selecionados os seguintes algoritmos supervisionados:

- *Support Vector Machine* (SVM);
- Regressão Logística;
- Árvore de Decisão;
- k-Nearest Neighbors (kNN)
- Discriminante de Fisher
- Gradient Boosting
- Redes Neurais MLP
- Random Forest

Já para classificação não supervisionada foi selecionado os seguintes algoritmos:

- *Support Vector Machine* (SVM);
- DBSCAN;
- Árvore de Decisão;
- Expectation-Maximization;
- Hierarchical Clust;
- KMeans;
- Fuzzy C-Means;
- Gaussian Mixture Models;

Esses algoritmos foram aplicados na base de dados das informações da impressora, no entanto, os resultados não foram satisfatórios, indicando que as informações contidas nele (histórico de dados adquirido) ainda não refletiam o bom ou mal funcionamento da impressora. Cabe ressaltar que todos esses algoritmos foram utilizados, mesmo com os resultados ruins, apenas para fins de aprendizado.

Primeiramente, foi aplicado o PCA (*Principal Component Analysis*) para agrupar variáveis correlacionadas, a Figura 13 apresenta o resultado obtido.

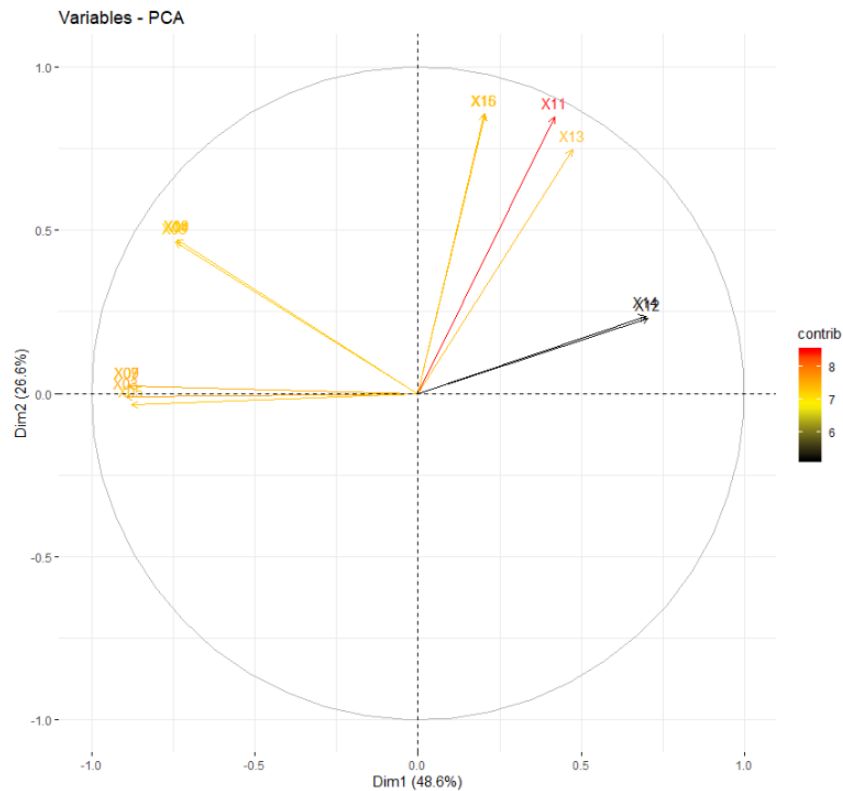


Figura 13 – Gráfico de correlação bidimensional (biplot) do PCA

O objetivo do PCA é resumir um conjunto de variáveis em um menor através da relação entre elas, em outras palavras, tenta-se eliminar redundância no conjunto de variáveis explorando a correlação entre elas. O gráfico *biplot* é uma representação visual das relações entre as variáveis que pretendemos trabalhar, sendo cada variável representada por um vetor e a proximidade (ou sobreposição) desses vetores indicam forte correlação.

Outro gráfico que auxilia na interpretação do PCA é o apresentado na Figura 14, esse no qual reflete o que o PCA de fato faz e a interpretação é semelhante ao gráfico anterior, os pontos próximos indicam variação semelhante.

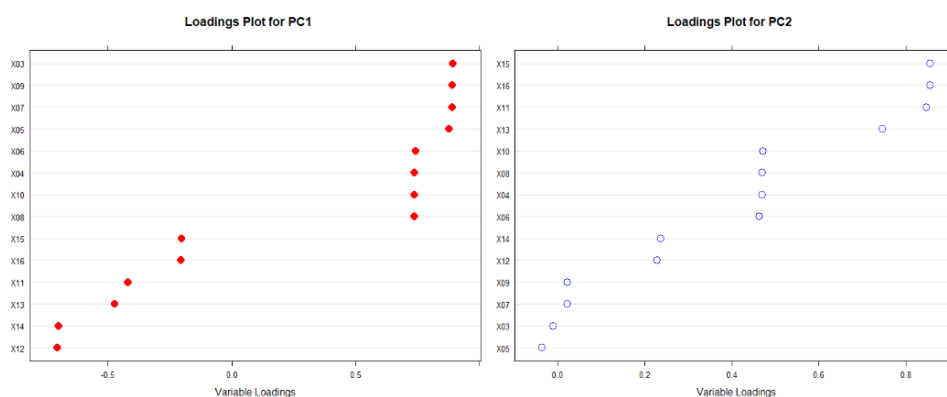


Figura 14 – Gráfico de dispersão do PCA

Da Figura 13, é possível definir quatro grupos de variáveis, sendo eles (X3, X9, X7, X5), (X6, X4, X10, X8), (X15, X16, X11, X13) e (X14, X12). De cada um desses grupos, tem-se um componente principal, aquele que explica melhor a variação das demais, sendo eles X3 (Head Model Temperature), X6 (Head Support Current), X15 (8V) e X14 (Head Support Last Requested Potentiometer) respectivamente.

Trabalhando com o componente principal de cada grupo, podemos não só resumir a análise, como também simplificar os modelos matemáticos que irão trabalhar com essas informações. Cabe ressaltar que, a presença de correlação em modelos insere um problema de dependência entre as variáveis, o que enviesa ou reduz a credibilidade do modelo.

Com o conjunto à ser analisando definido, primeiro foi realizada uma classificação não supervisionada usando os algoritmos apresentados anteriormente. Na maioria dos algoritmos de classificação não supervisionada, tem-se como entrada a base de dados à ser agrupada e o número de grupos esperados, no qual podemos encontrar usando uma busca exaustiva por exemplo, e a saída são os *clusters*, o agrupamento dos dados (não das variáveis como no PCA).

Da análise realizada, todos os algoritmos concordaram com dois agrupamentos (*clusters*), mas discordaram quando a quantidade era três ou mais. Esse fato se dá à características do algoritmo, como por exemplo, o DBSCAN (*Density-based spatial clustering of applications with noise*) que se baseia em densidade para encontrar os *clusters* (e possui ótimos resultados para problemas não lineares) ou o KMeans (que usa distância euclidiana).

A Figura 15 apresenta o resultado do DBSCAN obtido com dois *clusters*. Sendo o ponto preto um *outlier*.

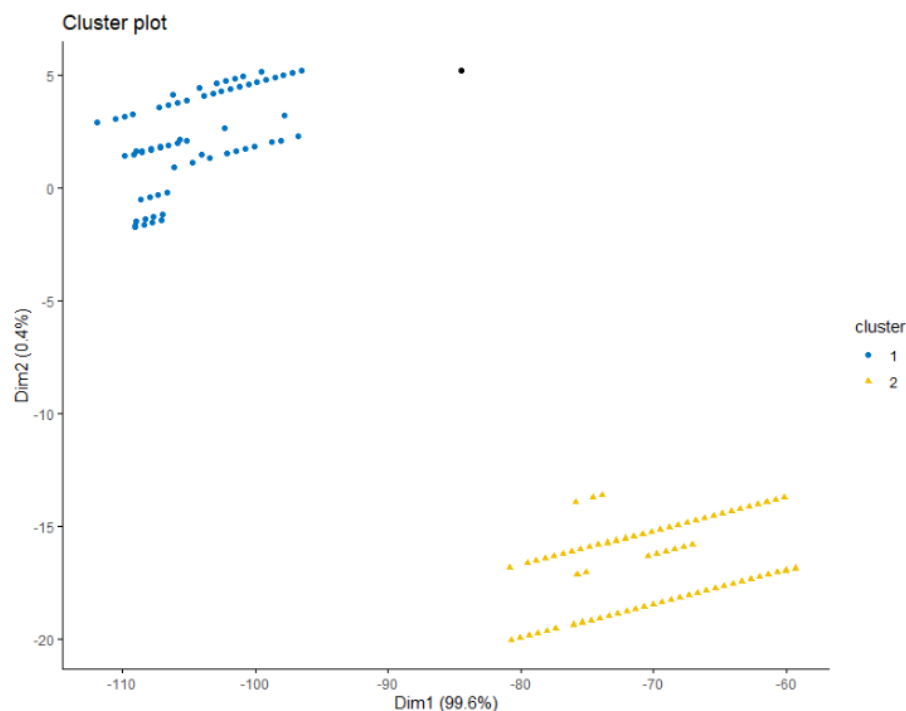


Figura 15 – Gráfico de agrupamento do DBSCAN

A pretensão nessa análise era investigar as características que foram agrupadas pelo algoritmo. Primeiro, fizemos uma comparação entre a variável Status e o que foi agrupado, obtendo o seguinte resultado (Tabela 1).

Clusters	Status	
	OK	Warm
<i>Outlier</i>	3	0
Cluster 1	7135	396
Cluster 2	1259	6

Tabela 1 – Tabela de classificação Clusters x Status

Nota-se que o Cluster 1 capturou 99% do Status *warm*, indicando que uma das características presentes nesse *cluster* é de temperatura elevada, no entanto, ainda não está claro quais características foram agrupadas. Para auxiliar na interpretação, foi analisado a distribuição dos dados para cada grupo (Figura 16).

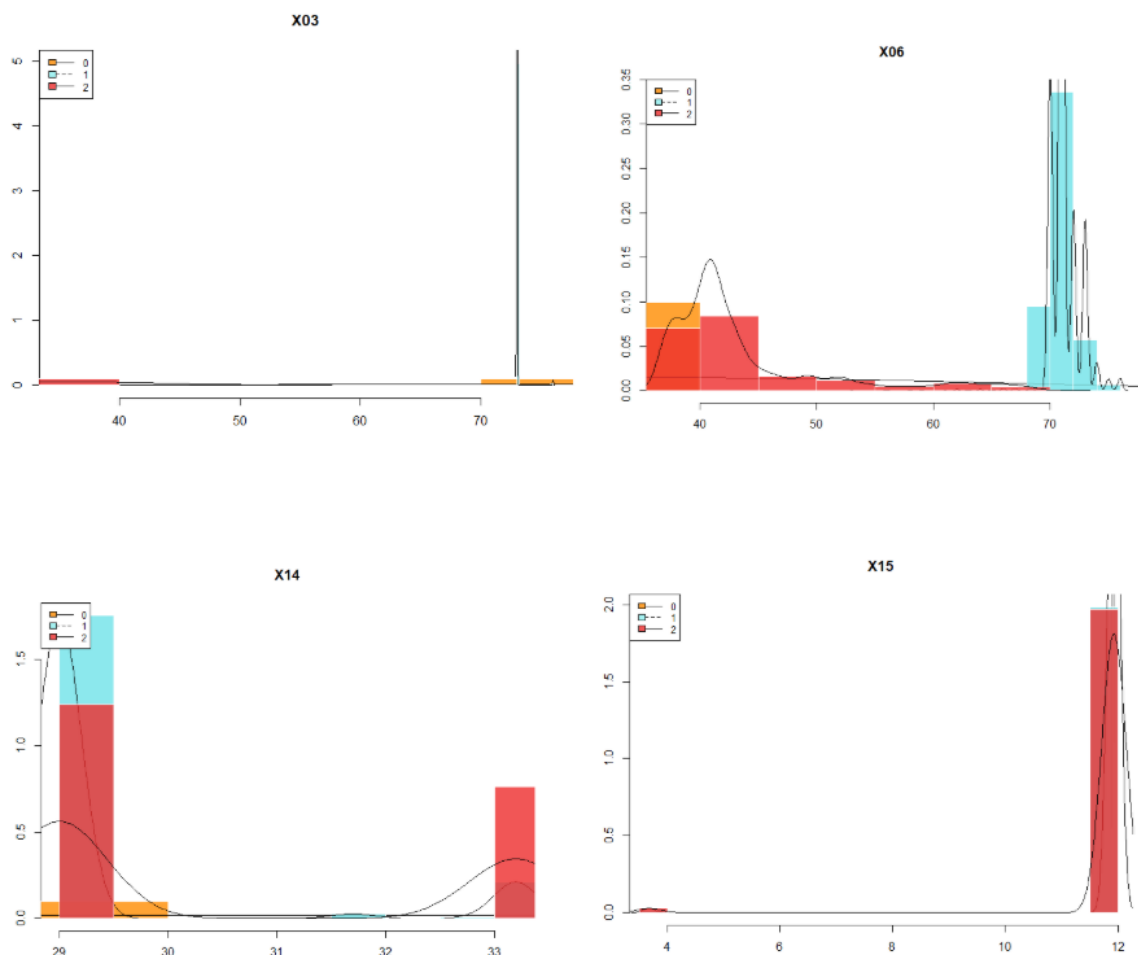


Figura 16 – Histogramas para cada variável com seus respectivos *clusters*

Percebe-se que para variável X03 e X06 os grupos foram separados entre temperatura próxima de 40 e próxima de 70, no entanto, as variáveis X14 e X15 ficaram mistas, mostrando que a influência do modelo foi devido à essas duas primeiras.

Desse ponto e com base na série temporal dos dados (Figura 5), pode-se perceber que o histórico de dados obtido não possui nenhuma característica relacionada às condições de falha da máquina ou não é possível realizar uma separação útil apenas com as informações extraídas pois a impressão que dá é que a separação foi baseada em um ponto de inflexão da série temporal quanto ao período de aquecimento/resfriamento do bico.

Mesmo com essas conclusões, resolvemos aplicar os algoritmos de classificação supervisionada com a finalidade de deixar códigos prontos para quando começarmos a analisar os dados do *testbed* já teríamos algo formatado para trabalharmos com os dados.

Para tais algoritmos, resolvemos usar como variável resposta o Status e as explicativas a X03, X06, X14 e X15. Primeiro, separamos a base de dados em duas, uma para ajustar o modelo, também chamado de treino, e a segunda para avaliar o desempenho do modelo, denominado teste.

A Figura 17 apresenta a curva ROC para o modelo ajustando usando regressão logística.

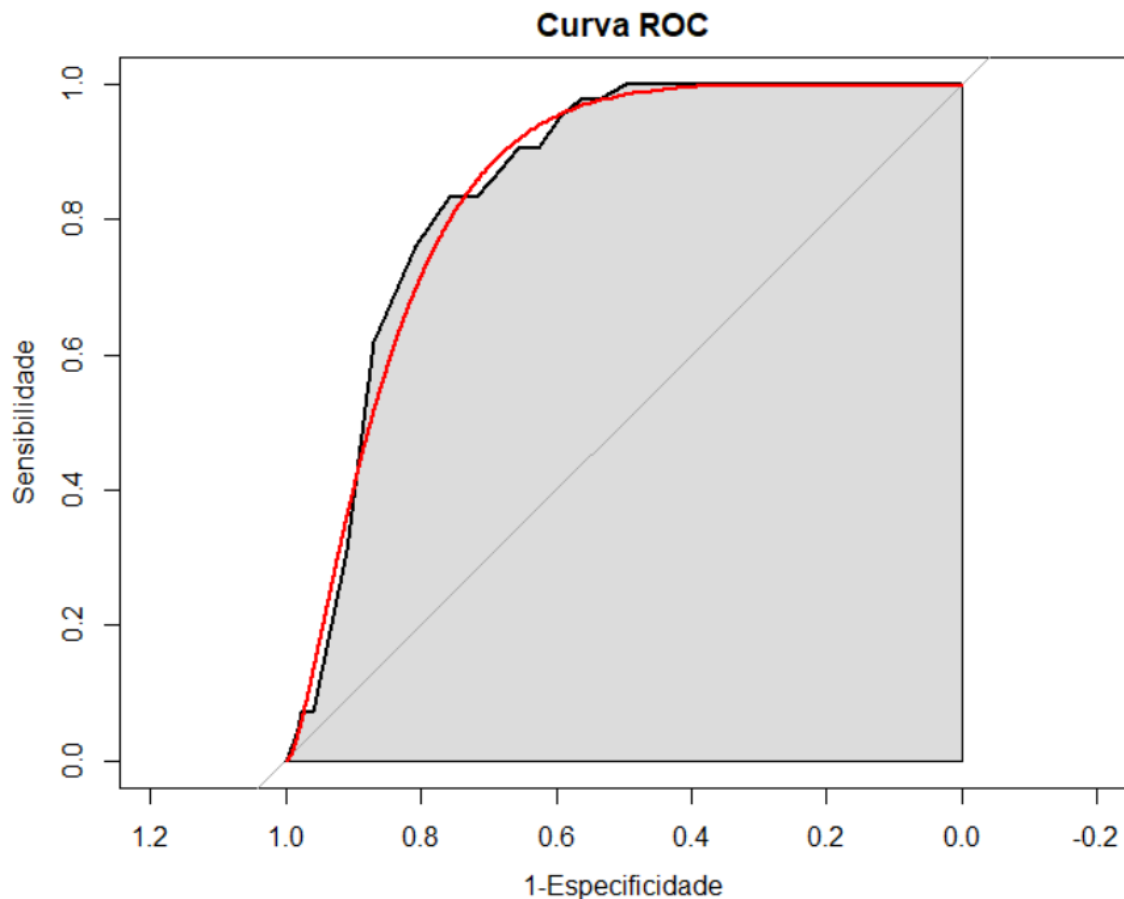


Figura 17 – Curva ROC para o modelo ajustado usando Regressão Logística.

A curva ROC auxilia em duas questões, a primeira é definir um limiar que maximize a sensibilidade e especificidade, ou seja, o ponto em que devemos considerar se algo pertence ou não à uma classe e a segunda indica o poder preditivo do classificador. Basicamente, a área sob a curva varia de 0 à 1 e quanto mais próximo de 1, maior será o poder preditivo.



O modelo ajustado apresentou uma área sob a curva de 0.85 e uma acurácia de 77% e a matriz de classificação é apresentada na tabela 3.

Tabela 2 – Matriz de classificação do modelo ajustado.

Status		OK	Warm	% correta
Observado	OK	134	43	76
	Warm	7	34	83
Total		141	77	77

Aplicando esse modelo na base de teste obtivemos uma acurácia de 68%.

Tabela 3 – Matriz de classificação aplicando o modelo na base de teste.

Status		OK	Warm	% correta
Observado	OK	109	86	56
	Warm	33	144	81
Total		142	230	68

Esses resultados indicam que as variáveis explicativas não são suficientes ou não conseguem explicar a variação da variável Status, em outras palavras, a relação que desejamos encontrar entre esse conjunto de variáveis e a resposta Status é fraca. Podemos mensurar esse característica fraca através do pseudo- $R^2$  (coeficiente de determinação), que é uma métrica que varia de 0 à 1 e nos diz o quão bem está ajustado o nosso modelo e consegue justificar os valores observados.

O pseudo- $R^2$  de Cox-Snell e Nagelkerke, as métricas mais utilizadas para essa análise, foram respectivamente 0.2 e 0.4, indicando que a separação indica pelo modelo não é tão boa assim, o que reflete na predição da base teste, no qual houve um decaimento de 77% para 68% e espera-se um desempenho entorno disso caso optássemos por utilizar esse modelo, o que claramente não é recomendado.

Devido a qualidade dos dados, não achamos necessário ou útil explorar mais as informações do banco tal como estatística descritiva, seleção de variáveis, implementação de uma coletânea (conjunto de modelos de aprendizado de máquina para definir um resultado), análise de série temporal, tratamento de *outliers*, dentre outras questões intrínsecas na análise de dados.

Embora também tenhamos aplicado essa base de dados da impressora 3D em outros algoritmos, não achamos necessário descrever os resultados obtidos, pois foram semelhantes. Enquanto a extração de logs vem de um *software* no sistema operacional que fica na impressora, as informações referentes a utilização de material, interrupção na impressão, dentre outras, é feita em um software externo à impressora (e não podem ser exportados), essa questão impossibilitou o casamento entre tais informações que poderiam auxiliar na análise dos dados.

## 1.4 Estudo sobre desenvolvimento de dashboards IoT

Durante o primeiro semestre do estágio, antes do advento das necessidades de desenvolvimento web para plataformas desenvolvidas no segundo semestre do ano, foram realizados estudos sobre as melhores linguagens/frameworks front-end e back-end e o melhor sistema de gerenciamento de banco de dados (SGBD) que deveriam ser utilizados para a implementação de uma plataforma web que representasse um dashboard IoT.

Para tanto, levou-se em consideração diversos aspectos, como a quantidade de tempo necessária para o aprendizado da linguagem/framework (ou seja, a curva de aprendizado), o quão relevante é a utilização da linguagem/framework na área de desenvolvimento de dashboards IoT, o quão rápida é a linguagem/framework no processamento de uma grande quantidade de dados, quais são os frameworks e SGBD que melhores se integram para desenvolvimento, o quão estabelecida a linguagem/framework está na área de desenvolvimento web, dentre outros.

Após um profundo estudo, verificou-se que, por exemplo, a linguagem *JavaScript* é de extrema relevância, com os frameworks front-end *Angular 2*, *React.js* e *Vue.js* sendo bem utilizados. Além disso, o framework back-end *Node.js* apareceu como sendo bem utilizado. Já em relação a linguagem *Python*, verificou-se que dois frameworks relevantes são *Django* e *Flask*, os quais permitem tanto implementação em back-end e front-end. Por fim, no caso dos SGBDs, as possíveis escolhas foram estabelecidas entre o *PostgreSQL* e o *MySQL*.

Além disso, como o *Vue.js* se mostrou um *framework* de curva de aprendizado mais rápida que os outros dois desenvolvidos em JavaScript, logo foi considerada uma opção válida de uso, até por conta de sua integração com bibliotecas de plotagem de gráficos.

Já no caso do back-end, o *Django* foi fortemente comparado com o *Node.js* como opção de escolha; como alguns pontos positivos do *Django*, estava o fato de que o desenvolvimento ocorre em uma linguagem com a qual já tínhamos mais familiaridade (*Python*), ao invés do *Node.js*, o qual se mostrou mais complexo de ser aprendido.

Por fim, como não foram vistas diferenças que impossibilitariam o uso de um ou outro SGBD, na época o *MySQL* foi preferido em detrimento do *PostgreSQL*, muito por conta da familiaridade que temos com a ferramenta.

Assim, o *stack* escolhido na ocasião foi o uso do *Vue.js* como *framework* de front-end, *Django* para a parte do back-end e o sistema de gerenciamento de banco de dados *MySQL*.

## 1.5 Funcionalidades para o SmartMonitor

Foram implementadas diversas funcionalidades para a plataforma Smart Monitor, pertencente a empresa executora do projeto IoT. Para tanto, foi utilizado o framework *Django*, o qual é um *framework* para desenvolvimento rápido para web, escrito em Python, que utiliza o padrão MVT (Model-View-Template).

Outra ferramenta utilizada foi o Django Rest Framework, o qual foi criado para facilitar o desenvolvimento da arquitetura REST em sistemas que utilizam Django. Além disso, foi feito uso do framework de testes do Django para testar a implementação realizada. Por fim, utilizou-se o sistema de gerenciamento de banco de dados PostgreSQL. Uma visão geral das funcionalidades desenvolvidas se encontra abaixo.

### 1.5.1 Banco de dados

Primeiramente, foi desenvolvido o banco de dados a ser migrado para o PostgreSQL a partir de um modelo relacional já fornecido. Tal implementação foi realizada no arquivo `models.py` presente no projeto Django.

O próximo passo foi a migração dos modelos para o PostgreSQL. Além disso, houve necessidade de geração de dados de testes, pois ao início da implementação não havia dados reais. Portanto, foram criados arquivos no formato *json* para importação no banco.

### 1.5.2 APIs de acesso e *endpoints*

Implementou-se métodos HTTP relacionadas ao REST (*Representational State Transfer*), o qual é um padrão de arquitetura para desenvolvimento de serviços web. Dessa forma, as APIs desenvolvidas tiveram o objetivo de permitir a comunicação entre um cliente web e um servidor utilizando REST.

O primeiro método, denominado GET, visou a obtenção de dados do banco (os quais representavam, por exemplo, os valores monitorados com o uso de sensores instalados em máquinas) pelo cliente web, de forma que este pudesse utilizar os dados para qualquer futuro processamento, como a exibição dos dados obtidos em gráficos de séries temporais.

O modelo do projeto Django (MVT) tornou a implementação dividida entre os arquivos `urls.py` e `views.py`. O primeiro deles representa o endereço pelo qual os dados são obtidos, e o segundo, a implementação dos métodos em si.

A Figura 18 exibe um exemplo de *endpoint* pelo qual os dados são visualizados utilizando a interface gráfica do Django Rest Framework, exibindo informações como identificador do modelo de equipamento utilizado, identificador e nome da empresa à qual o equipamento pertence, identificador do equipamento, dentre outros.

O próximo passo foi a implementação do método POST.

A partir do POST, é possível que dados de sensores conectados às máquinas sejam armazenados no banco de dados presente no servidor, de forma que possam ser consultados pelo cliente web posteriormente.

A Figura 19 apresenta uma interface gráfica para o método POST. A inserção de dados está representada em formato *json*, sendo que os dados incluem tanto identificadores de máquinas e equipamentos quanto as variáveis *timestamp* e *value*, que representam o tempo em que uma medição foi obtida por determinado sensor e o valor medido.

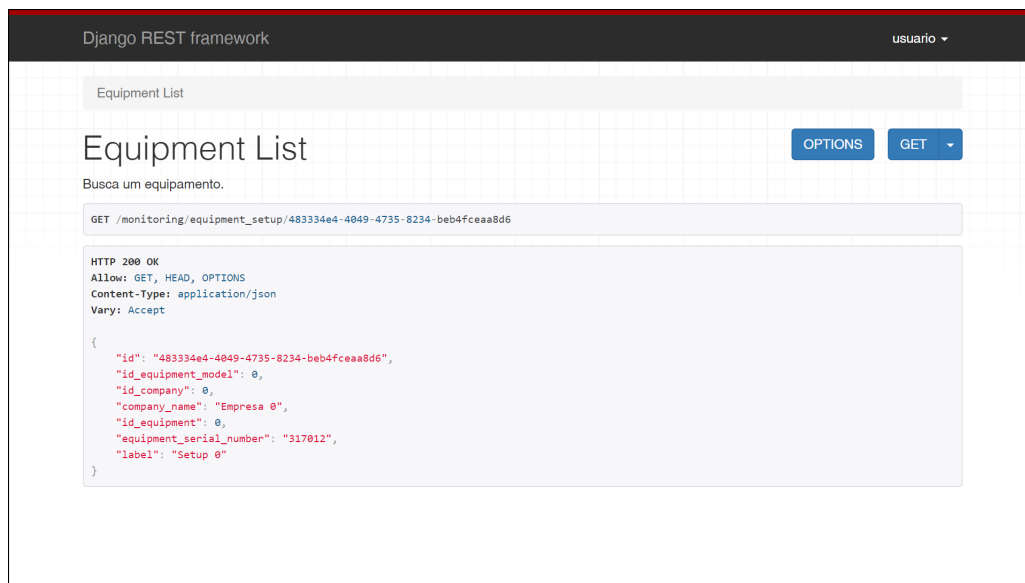


Figura 18 – Exemplo de API GET no navegador

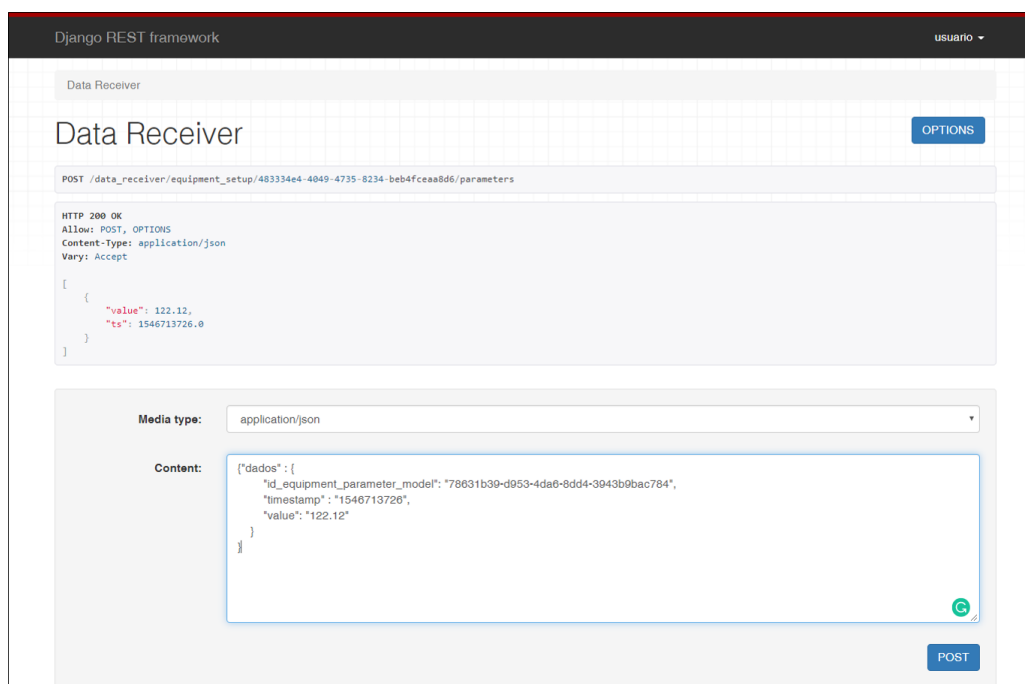


Figura 19 – Exemplo de API POST no navegador

### 1.5.3 Buffer

Foi necessário o desenvolvimento de um *buffer* para a plataforma, o qual teve o papel de permitir o armazenamento rápido de dados em uma cache. Dessa forma, por exemplo, os dados dos últimos 30s podem ficar disponíveis a um cliente web em um espaço de memória alocado para isso no servidor, de forma a diminuir a quantidade de acessos no banco. Assim, há duas funções presentes no código do Buffer. A função Set realiza a inserção/modificação dos dados no Buffer, já a função Get permite a obtenção dos dados.

### 1.5.4 API-Key

Foram desenvolvidas também API Keys, as quais são credenciais que são fornecidas para acesso a funcionalidades de uma API. Essa medida garante que um usuário não credenciado ou que não possui autorização para realizar determinadas ações ou capturar informações no banco de dados tenha seu acesso negado.

No caso, utilizou-se do próprio suporte do *framework Django* para esta implementação, de forma que são utilizadas duas chaves, descritas a seguir:

- Chave pública: é fornecida pelo back-end e enviada para o cliente web. A posse de tal chave por terceiros não possibilita o acesso a API, pois é necessário o uso de uma chave privada para tal;
- Chave privada: está de posse do cliente web, o qual deve utilizá-la para acesso externo às funcionalidades da API.

As API Keys possuem um período a partir do qual as chaves expiram, de forma que deve-se haver requisições para a geração de novas chaves. Tal fator aumenta a segurança sobre o acesso à API.

### 1.5.5 Testes

Por fim, foram realizados testes para verificar o funcionamento da interação entre o método POST e o Buffer. Assim, como a ferramenta de testes do *Django* gera um novo banco de dados específico para a realização dos testes, criou-se um código para geração de dados aleatórios a serem utilizados para isso.

O teste implementado teve o objetivo de comparar os dados presentes no banco e no *Buffer* após um determinado número de inserções realizadas. Assim, o fluxo normal de execução seria como segue: um usuário criado para testes realiza a chamada do método POST; em seguida, o POST insere os dados no banco, instancia um *Buffer* e realiza a chamada da função de SET do *Buffer*. Por fim, o *Buffer* insere os valores em sua fila.

Definiu-se que seria testado o valor de 15 inserções, ou seja, se ao final de todas as inserções houver 15 valores tanto na tabela na qual os dados foram inseridos quanto no *Buffer*, então o teste retorna um status de sucesso.

### 1.5.6 Docker

Após a conclusão das implementações, foi realizado um estudo sobre o conceito e funcionamento do *Docker*, que é um software contêiner que fornece uma camada de abstração e automação para virtualização de sistema operacional no Windows e no Linux.

## 1.6 Projeto de Sensoriamento Wi-Fi

A Figura 20 apresenta um protótipo inicial pensado para o desenvolvimento desse projeto. A ideia é fazer o NodeMCU atuar como um servidor que recebe requisições das aplicações e envia os dados para os sensores via TCP/IP.

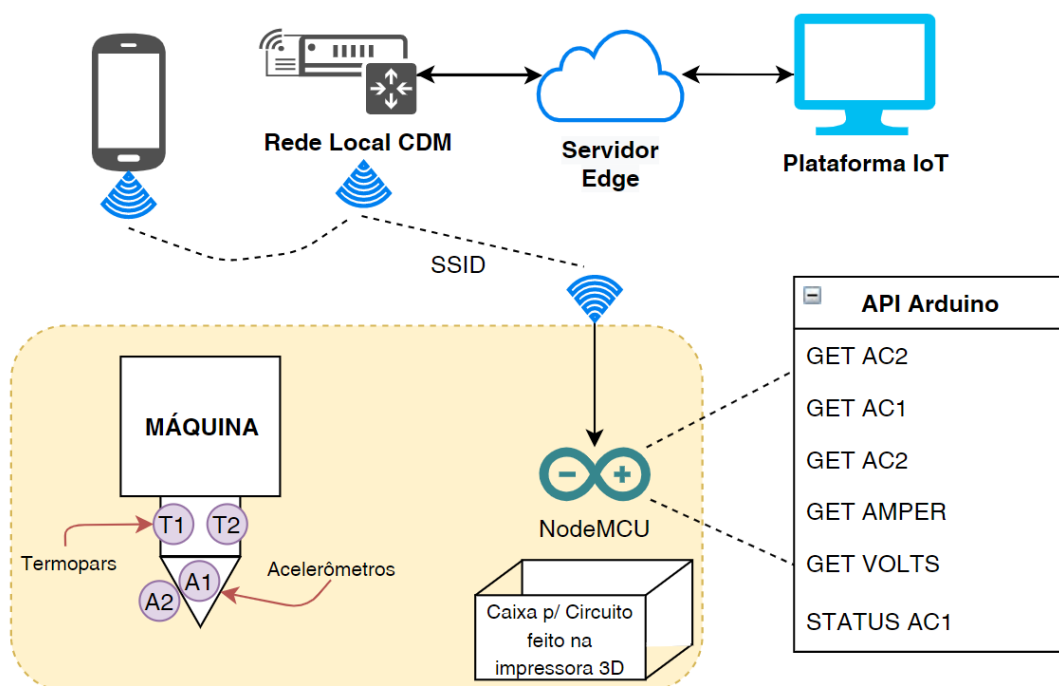


Figura 20 – Protótipo da estrutura para o sensoriamento wi-fi

O orçamento previsto para foi de aproximadamente de R\$ 350,00 (Figura 21).

#	Produto	Preço	Qtde	Total
1	Sensor De Temperatura DS18B20	R\$ 16,90	1	R\$ 16,90
2	Acelerômetro E Giroscópio 3 Eixos 6 Dof MPU-6050	R\$ 19,50	2	R\$ 39,00
5	Protoboard 830 Pontos	R\$ 13,50	1	R\$ 13,50
6	NodeMCU ESP32	R\$ 66,40	1	R\$ 66,40
7	Jumpers - Femea/femea - 20 Unidades De 20 Cm	R\$ 6,00	1	R\$ 6,00
8	Jumpers - Macho/macho - 20 Unidades De 20 Cm	R\$ 6,00	1	R\$ 6,00
9	Jumpers - Macho/femea - 20 Unidades De 20 Cm	R\$ 6,00	1	R\$ 6,00
10	Módulo Relé 5v 2 Canais	R\$ 12,80	1	R\$ 12,80
11	Componentes básicos (resistores, capacitores, MUX, transistor)	R\$ 30,00	1	R\$ 30,00
12	Chave Gangorra Mini Liga / Desliga 6a 125v Ac   3a 250v Ac	R\$ 2,00	1	R\$ 2,00
13	Fonte NodeMCU 9v	R\$ 15,00	1	R\$ 15,00
14	Cabo USB NodeMCU	R\$ 21,89	1	R\$ 21,89
16	Caixa Circuito	R\$ 28,98	1	R\$ 28,98
17	Roteador TP-Link Simples	R\$ 60,00	1	R\$ 60,00
18	Módulo Ethernet ENC28J60 Shield	R\$ 24,90	1	R\$ 24,90

<b>Total</b>
R\$ 349,37

Figura 21 – Orçamento previsto para o projeto de sensoriamento.

## 1.7 Artigos e relatório técnico

### 1.7.1 Artigo sobre Redes IoT

O primeiro artigo elaborado discutiu conceitos de Redes de Internet das Coisas. Inicialmente, é dada uma visão geral sobre IoT e sistemas relacionados, assim como as três camadas em que podem ser divididos: hardware, infraestrutura e aplicação.

Em relação às Redes IoT, discutiu-se sobre exemplos das três categorias em que se pode dividi-las: as de curta distância (WiFi, Bluetooth, RFID, ZigBee, 3G, etc.); de média distância (LPWAN: LoRa e SIGFOX); e de longa distância (redes celulares e redes LPWAN celulares, como NB-IoT). Para cada tipo, foram apresentadas as principais características e comparativos com vantagens e desvantagens, além de exemplos com aplicações reais.

### 1.7.2 Artigo sobre Casos de uso

O segundo artigo teve o objetivo de apresentar alguns dos casos de uso de Internet das Coisas em regiões urbanas, rurais e na indústria. Em relação a regiões urbanas, foi dado um enfoque maior em redes IoT como LoRa e Sigfox, além de projetos de sensoriamento e automatização para cidades inteligentes. Já no caso de zonas rurais, o foco foi em soluções de IoT na agricultura e na pecuária, ou seja, na agricultura inteligente.

Por fim, o estudo sobre uso em indústrias envolveu IIoT (Internet das Coisas Industrial), apresentando soluções para coleta de dados de ativos industriais. A partir dos dados coletados, pode-se realizar estudos destes, possivelmente em tempo real, de forma a realizar ações que tragam otimização e maior eficiência à produção.

### 1.7.3 Revisão da Literatura e Artigo sobre Manutenção preditiva na indústria

Foi realizada uma revisão da literatura sobre os trabalhos que vem sendo desenvolvidos e patentes registradas sobre o tema de manutenção preditiva na indústria. A finalidade era não só apresentar os conceitos e vantagens, como também a metodologia (algoritmos, estruturação, dificuldades alvo, etc.) e considerações avaliadas em projetos da área.

Os projetos e patentes encontrados se mostraram diversos. No geral, os sistemas desenvolvidos envolvia infraestrutura, estruturação e/ou ciência de dados (seja de forma qualitativa ou quantitativa), o que vai contra a visão inicial de que essa temática é apenas Inteligência Artificial, ou seja, estávamos enxergando apenas uma parte do sistema.

O artigo escrito visou passar uma visão objetiva e compreensiva apresentando, principalmente, a importância de se adotar ou integrar sistemas de manutenção preditiva na empresa, dando como exemplo uma aplicação na manufatura subtrativa, essa na qual está em nosso contexto.

## 1.8 Projeto Executivo

Acompanhamos e ajudamos na revisão do Projeto Executivo elaborado pela Tytans Systems para o projeto. Este documento foi essencial para compreender tanto a dimensão quanto a organização do projeto e todos seus requisitos, considerações necessárias e exigências burocráticas importantes para sua execução e desenvolvimento.

Com isso, ficou demonstrada a importância da parte administrativa para que projetos desse porte sejam concretizados, por não só envolverem conhecimentos técnicos, como também de gerência e colaboração de uma equipe como um todo.

## 1.9 Eventos e Visitas Técnicas

### 1.9.1 Eventos

- Nexus Summit.

Evento realizado no Parque Tecnológico de São José dos Campos, foi criado para reunir o ecossistema de *startups* e empreendedores do Brasil. A partir da realização do evento, pudemos entender melhor sobre os desafios enfrentados por startups nos dias atuais.

- ProMarking: Workshop Indústria 4.0

Este workshop nos trouxe novos conhecimentos sobre Indústria 4.0, mais especificamente dentro dos seguintes tópicos que foram abordados durante a realização dele: Os 10 pilares da Indústria 4.0; Rastreabilidade Industrial. O futuro do seu negócio; Como resolver os principais problemas de gravação e rastreabilidade na indústria.

- Balcão 360 – Gestão e Tecnologia

O evento é um serviço exclusivo do Escritório de Negócios do Parque Tecnológico e visa aproximar as empresas dos pesquisadores de diversas áreas, de forma a auxiliar empresas e empresários a aumentar a competitividade e melhorar os resultados com base nas pesquisas e técnicas desenvolvidas dentro da universidade.

Durante a participação nos eventos, pudemos interagir com diferentes pesquisadores que nos trouxeram conhecimentos úteis sobre a questão de Internet das Coisas e Probabilidade e Estatística. As conversas realizadas foram importantes dentro dos projetos desenvolvidos no período do estágio.

- Projeto Colmeia

Criado para integrar alunos de graduação e pós-graduação das diversas universidades do Vale do Paraíba, professores e profissionais das empresas da região. No 1º Colmeia, houve separação dos integrantes entre oito grupos de interesse, de forma a desenvolver



pesquisas e projetos relacionados às atuais demandas de mercado. Os tópicos foram internet das coisas, desenvolvimento para celulares, desenvolvimento web, seleção e desenvolvimento de materiais, blockchain, machine learning, data science e sistemas embarcados.

Já no 3º Colmeia, houve um direcionamento maior no projeto PIPE - FAPESP, com a abertura do evento tendo sido realizada por um representante da FAPESP. Em seguida, houve a abertura de mesas para prospecção e alinhamento de projetos nos temas Data Science, Internet das Coisas, Sistemas Embarcados, Machine Learning, Seleção e Desenvolvimento de Materiais, Desenvolvimento Web, Blockchain e Segurança Cibernética.

- RM Vale TI

Feira de Tecnologia e Inovação da Região Metropolitana do Vale do Paraíba, criada pelo APL TIC Vale, ocorre desde 2014 no Parque Tecnológico. O evento reúne exposição de produtos e serviços de TI, congresso, rodadas de negócios e contatos entre startups e investidores. Dentro deste escopo, tivemos a oportunidade de participar principalmente da exposição, e dos painéis realizados dentro do congresso.

### 1.9.2 Visitas técnicas

- Thyssenkrupp

Pudemos ter uma visão aprofundada do trabalho dos operadores de máquinas da indústria, além de entender os desafios enfrentados por indústrias do setor.

- Conexão Local

Esta visita nos permitiu visualizar o funcionamento de uma startup que trabalha no conceito de "fábrica de software", além de entender as linguagens e os programas utilizados pela Conexão Local, o que foi de auxílio no desenvolvimento das plataformas durante o período de estágio.

- 3DTecnologia

Fomos apresentados às impressoras 3D que são desenvolvidas por ela. Um ponto relevante na visita foi a questão do monitoramento por meio de sensores que a startup realiza nas impressoras, adquirindo parâmetros como temperatura com o uso de microcontroladores.

- Promarking

A Promarking é uma empresa que realiza gravação a laser e rastreabilidade digital. Com a visita, pudemos verificar o funcionamento das máquinas utilizadas para a fabricação dos produtos e os desafios enfrentados pela empresa.