



UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia

Departamento Engenharia Elétrica

Débora Ramos Romagnolo

Irrigador de Plantas Residencial: Uma solução com IoT

2019

São Carlos – SP

Débora Ramos Romagnolo

Irrigador de Plantas Residencial: Uma solução com IoT

Trabalho de Conclusão de Curso apresentado à Banca examinadora do curso de Engenharia Elétrica da Universidade Federal de São Carlos como requisito para obtenção do título de Engenheira Eletricista.

Orientador: Prof. Dr. Robson Barcellos

2019
São Carlos – SP

Débora Ramos Romagnolo

Irrigador de Plantas Residencial: Uma solução com IoT

Trabalho de Conclusão de Curso apresentado à Banca examinadora do curso de Engenharia Elétrica da Universidade Federal de São Carlos como requisito para obtenção do título de Engenheira Eletricista.

Aprovada em 11 de Julho de 2019

BANCA EXAMINADORA

Prof. Dr. Robson Barcellos – Orientador
Universidade Federal de São Carlos

Prof. Dr. Samuel Lourenço Nogueira
Universidade Federal de São Carlos

Prof. Dr. Helder Vinícius Avanço Galeti
Universidade Federal de São Carlos

*À minha família,
pelo amor incondicional.
Aos meus amigos,
pela cumplicidade.*

AGRADECIMENTOS

Agradeço aos meus pais, pelo incentivo e apoio nos estudos, fundamentais para que eu decidisse seguir por este caminho e aos meus irmãos, que são os melhores parceiros.

Aos amigos que estão sempre ao meu lado, Lucas Basso, Lucas Bueno, Beatriz Capelli, Camila Yumi, Bruno Giangiulio e Amanda Bento. Com vocês sou muito mais forte.

À Prof. Ana Luiza Perdigão pelo carinho, pelas broncas de mãe e por sempre me fazer acreditar no meu potencial.

Ao Prof. Helder Galeti pela paciência e todo apoio intensivo durante os anos de graduação.

Ao Prof. Robson Barcellos por me acolher num momento conturbado de transições e incertezas. Nossos encontros geraram conversas que serão inesquecíveis.

Aos meus amigos e guerreiros da elétrica Julia Paschoal, Raphael Zanandrea, Alini Alvarenga, Leonardo Conceição e Gabriela Yoshida. A nossa união foi, sem dúvida, o que garantiu a nossa conquista nesta universidade.

Ao Cursinho Pré-Vestibular pela oportunidade de me tornar mais humana, por todo o crescimento profissional e por me fazer enxergar que sou uma pessoa única;

À UFSCar, pelo acesso ao conhecimento e pela experiência transformadora. Por ser o alicerce para todo meu desenvolvimento pessoal e profissional. Depois de 5 anos convivendo este organismo pulsante que ela é, tenho a sólida consciência de que o conhecimento e a educação têm o poder de mudar realidades como a minha.

Ao meu melhor amigo e companheiro Heitor Depieri, que esteve comigo todos os dias nos últimos 7 anos. Não cabe nestas páginas o tanto que sou grata por ter você ao meu lado.

Por último, agradeço a todas as mulheres cientistas, operárias, artistas e bruxas que vieram antes de mim e que me permitiram estar aqui hoje.

*“Como mulher eu não possuo país.
Como mulher, meu país é o mundo todo.”*

- Virginia Woolf

RESUMO

O presente trabalho se constitui no desenvolvimento e construção de um protótipo de irrigador automático de plantas para espaços domésticos, remotamente controlado e monitorado pela internet, utilizando o microcontrolador *Arduino*. O sistema conta com sensores para medições de temperatura do ambiente, umidade do solo, umidade relativa do ar e luminosidade. O protótipo possui uma bomba de irrigação que é acionada de acordo com os dados fornecidos pelos sensores ou por um temporizador configurado pelo usuário. Todos os dados obtidos pelos sensores são enviados para a plataforma *ThingSpeak*, o que permite seu armazenamento em nuvem.

Palavras-chave: Irrigador Automático, *Arduino*, Internet das Coisas, Automação via *web*, *ThingSpeak*.

ABSTRACT

The present work consists in development and construction of an automatic prototype plant irrigation system for domestic spaces, the device is remotely controlled and monitored through the Internet using the *Arduino* microcontroller. The system has sensors for measuring the room temperature, soil moisture, relative humidity and luminosity. The prototype has an irrigation pump that is driven according to the data provided by the sensors or a timer configured by the user. All the data obtained by the sensors are sent to the *ThingSpeak* platform, which allows them to be stored in the cloud.

Key-words: Automatic Irrigator, *Arduino*, Internet of Things, Web Automation, *ThingSpeak*.

Sumário

1. Introdução.....	18
1.1. Objetivos.....	19
1.2. Estrutura do Trabalho	20
2. Internet das Coisas.....	21
2.1. Dispositivos Inteligentes	22
3. Fatores de Cultivo.....	24
3.1. Umidade do Solo e Irrigação	24
3.2. Luminosidade	24
3.3. Temperatura e Umidade do Ar	25
4. Trabalhos Relacionados	26
5. Materiais e Métodos	27
5.1. Etapas do Projeto	27
5.2. Requisitos do Sistema	27
5.3. Materiais Utilizados.....	28
5.3.1. Arduino.....	28
5.3.2. Medição de Umidade do Solo	30
5.3.3. Medição de Luminosidade	31
5.3.4. Medição de Temperatura e Umidade Relativa do Ar	32
5.4. Estação local	33
5.4.1. Calibração dos sensores.....	33
5.4.2. Evolução do protótipo	34
5.5. ThingSpeak	35
5.6. Página Web.....	40
6. Resultados e Discussão	42
7. Trabalhos Futuros.....	48
8. Conclusão.....	49
9. Referências	50
ANEXO A.....	54
ANEXO B.....	63
ANEXO C.....	72

1. Introdução

Com a constante evolução da tecnologia da informação, a sociedade tem passado por diversas transformações sociais que impactam no seu estilo de vida, na maneira de pensar e nos meios de comunicação. Os dispositivos móveis e a internet estão influenciando diversos aspectos da vida em sociedade, como práticas sociais de diferentes naturezas (VILAÇA e ARAUJO, 2016).

A crescente popularização da tecnologia móvel está presente em toda a sociedade, uma vez que os dispositivos móveis atendem à demanda dos usuários de estarem o tempo todo conectados na internet (GEWEHR, 2016). Em 2014 foram comercializados mais de 1,2 bilhões de dispositivos móveis no mundo, representando um aumento de 23% em relação ao ano anterior. Apenas na América Latina houve o aumento de 59% (POVO, 2015).

A expansão de tecnologias móveis e facilidade de comunicação promovem a conexão de objetos de uso cotidiano à internet, como aparelhos domésticos e dispositivos eletrônicos em geral. Esta prática, também em constante crescimento, é conhecida como Internet das Coisas ou *Internet of Things (IoT)*. A proposta das soluções em *IoT* é possibilitar a conexão dos equipamentos com a rede e facilitar a troca de informações, o monitoramento em tempo real e o controle de dispositivos de forma automática ou programada.

O recente estudo *The Internet of Things: Today and Tomorrow* publicado pela *Aruba*, uma companhia mantida pela *Hewlett-Packard*, prevê que a disseminação das tecnologias de *IoT* ocorrerá a partir de 2019, uma vez que 85% das empresas planejam implantar *IoT* até 2019, impulsionadas pela necessidade de inovação e eficiência comercial (ARUBA, 2019). Estima-se que até 2020 a *IoT* possuirá o mercado de 1,7 trilhão de dólares (IDC, 2019).

A escolha do presente trabalho foi motivada pela ascensão do uso da tecnologia para automatizar uma tarefa. O cultivo de plantas em ambientes urbanos ganha cada vez mais espaço no cotidiano das pessoas. Simples cuidados, como regar plantas e observar a quantidade de horas que ficam expostas ao sol tornam-se tarefas difíceis (PETRONILHO, 2015).

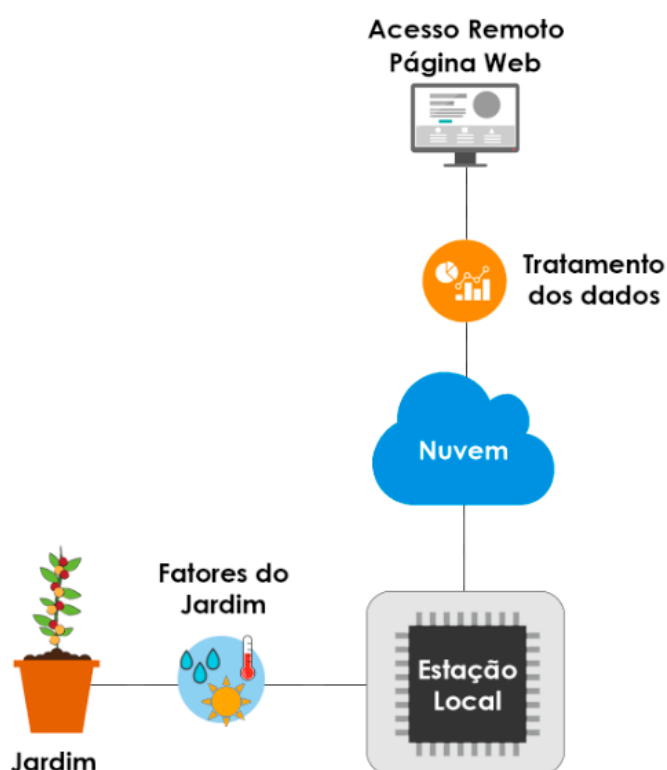
1.1. Objetivos

Este trabalho tem como objetivo desenvolver uma solução em *IoT* para fazer a irrigação automática de um jardim (ou vasos de plantas) e o seu controle e monitoramento remoto. Para atingir este objetivo, os seguintes passos foram realizados:

- Desenvolvimento de uma estação local de *hardware* com um microcontrolador *Arduino* e sensores responsáveis pela coleta de medidas de temperatura, umidade relativa do ar, luminosidade e umidade do solo. A estação também conta com uma bomba de irrigação que será acionada por um relé;
- *Upload* dos dados medidos pelos sensores em nuvem, para promover o acesso remoto;
- Desenvolvimento de uma página web para controle da estação local, através de nuvem.

A figura 1 apresenta um diagrama esquemático da solução proposta.

Figura 1 – Diagrama esquemático da solução proposta.



Fonte: Autoria Própria

1.2. Estrutura do Trabalho

Este trabalho está organizado da seguinte forma: o Capítulo 2 fornece um panorama sobre Internet das Coisas, sua definição e aplicações, bem como as tecnologias de hardware e comunicação que permitem seu desenvolvimento. O Capítulo 3 apresenta os fatores básicos necessários para o cultivo de plantas. O Capítulo 4 apresenta um breve resumo dos principais trabalhos utilizados como referência. No Capítulo 5 são apresentadas as etapas do projeto, os requisitos do sistema, os materiais utilizados e os procedimentos utilizados. No Capítulo 6 estão presentes os resultados e discussão. O Capítulo 7 apresenta as perspectivas de melhoria para trabalhos futuros. No Capítulo 8 está a conclusão do trabalho e no Capítulo 9 as referências bibliográficas consultadas.

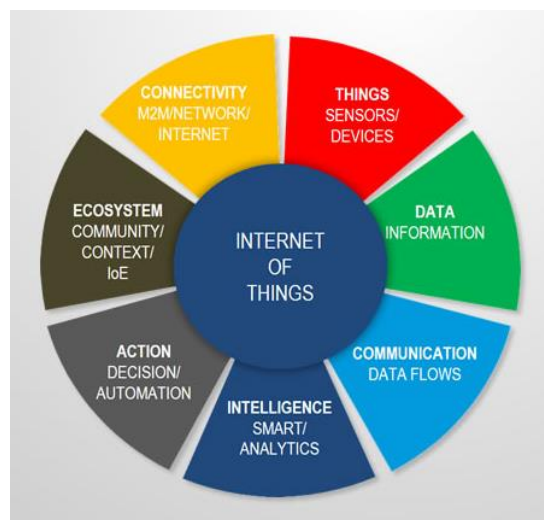
2. Internet das Coisas

Segundo Kevin Ashton (MARINO, VASCONCELOS e MORAES, 2017):

Internet das Coisas significa sensores conectados na internet e se comportando de maneira semelhante à internet, fazendo conexões e *ad hoc*, compartilhando dados livremente e permitindo aplicações inesperadas, para que os computadores possam entender o mundo à sua volta e se tornar o sistema nervoso da humanidade. (ARUBA, 2019)

Pode-se compreender a *IoT* como a junção de vários tipos de tecnologia de forma complementar, que fazem a conexão do mundo virtual com o mundo físico. A figura 2 apresenta algumas características encontradas em um sistema de *IoT*.

Figura 2 – Características comuns em um sistema de *IoT*.



Fonte: *IoT Heading for Mass Adoption by 2019 Driven by Better-Than-Expected Business Results* (I-SCOOP, 2019).

“Coisas”: São objetos inteligentes, que são responsáveis pela transformação das ações ou estímulos do ambiente em dados.

Comunicação: Diz respeito às técnicas para conectar os objetos desejados, como o *wi-fi*, *bluetooth*, *RFID* entre outros. Trata-se de um fator crítico, pois implica no consumo de energia e é o bloco responsável pela transmissão dos dados, para que estes sejam tratados em seguida.

Dados: São resultado da interação dos sensores com o ambiente. Os dados precisam de tratamento para que sejam transformados em informação.

Ação: Os sistemas de *IoT* são essencialmente voltados para a automação, seja numa aplicação industrial, de negócios ou residencial. A automação está ligada com o objetivo do projeto e depende dos sensores dos objetos e consequentemente dos dados obtidos por eles.

Conectividade: Todas as definições de *IoT* incluem o elemento de conectividade com uma rede de coisas, sensores, dispositivos, dados. Existem vários protocolos e padrões de conectividade para redes de *IoT*, seja este sistema com fio ou sem fio.

Inteligência: Consiste na capacidade do objeto de tomar decisões sem a interferência humana.

Ecossistema: O ambiente digital composto pela interação de diversos objetos, ferramentas digitais, indivíduos especializados e outros elementos de integração. Por meio da troca de informações, a *IoT* tem por objetivo criar uma rede inteligente superior aos elementos individuais (PEREIRA, 2013).

Empresas têm investido em plataformas que possam aproximar a *IoT* do desenvolvimento de novas soluções, como é o caso do Arduino, que recentemente lançou a plataforma *Arduino Yunn*, que possui um módulo *wi-fi* integrado. Além do *Arduino*, pode-se citar o *NodeMCU*, que consiste em um sistema de código aberto, concorrendo no mercado com o *Arduino Yunn* (MARINO, VASCONCELOS e MORAES, 2017).

2.1. Dispositivos Inteligentes

Um dispositivo inteligente deve ser capaz de interagir com o ambiente e por isso a definição do hardware é um elemento crítico para uma aplicação *IoT*. Os principais elementos de hardware de um dispositivo inteligente são:

- **Processamento/memória:** Unidade composta por um microcontrolador, um conversor analógico-digital para receber sinais dos sensores e uma memória interna para armazenamento de programas e dados. É desejável que esta unidade seja a menor possível e econômica em termos de energia.
- **Comunicação:** É a unidade que permite a conexão de um objeto com a rede ou com outros objetos. A comunicação pode ser com fio ou sem fio.
- **Energia:** É a unidade que fornece energia para o sistema/objeto inteligente. Podem ser utilizadas baterias, energia solar, energia elétrica, entre outras.

- **Sensores/atuadores:** Unidade responsável pelo monitoramento do ambiente em que o objeto se encontra. Os sensores são responsáveis pela captura de grandezas físicas como temperatura, pressão, umidade, entre outros capazes de gerar resposta aos estímulos do ambiente. Os atuadores são responsáveis por produzir ações mediante um comando, seja esta ação motivada pela leitura das variáveis do ambiente ou um comando externo.

Além dessa estrutura básica, os objetos inteligentes possuem um endereço de *internet protocol (IP)*, o que permite que eles possam se comunicar com outros objetos inteligentes ou com a rede (SANTOS, SILVA, *et al.*, 2017). É interessante também o desenvolvimento de objetos inteligentes capazes de tomar decisões de acordo com o contexto, com base no histórico de situações conhecidas anteriormente, sem que haja a interação humana direta.

3. Fatores de Cultivo

3.1. Umidade do Solo e Irrigação

A água é um fator essencial para a sobrevivência, desenvolvimento e reprodução dos seres vivos conhecidos. As principais funções da água nas plantas estão relacionadas à estrutura, ao transporte de nutrientes, ao metabolismo e ao crescimento (EMBRAPA, 2012).

O processo de irrigação do solo consiste na aplicação de água em quantidades adequadas, com o objetivo de proporcionar a umidade necessária para o desenvolvimento das plantas nele cultivadas (CARVALHO, 2016).

A determinação da umidade do solo em uma amostra de terra é importante para a automação do processo de irrigação pois este parâmetro pode ser utilizado para decidir quando existe necessidade de a planta ser irrigada (CORRÊA e GOMES, 2016).

Pode-se obter a medida de umidade do solo a partir de propriedades físicas, tais como capacitância e resistência elétrica de uma sonda colocada no solo, as quais variam de acordo com a quantidade de água no solo (DINIZ, 2017).

3.2. Luminosidade

A absorção da luz pela planta é um dos fatores que torna possível a obtenção de energia (glicose) necessária para seu desenvolvimento. Este processo é conhecido por fotossíntese e por este motivo, a luz é o fator do ambiente mais importante para o crescimento das plantas, pois sem energia, nenhum outro processo é executado pela planta (EMBRAPA, 2012).

A quantidade de luz necessária para o desenvolvimento e crescimento de uma planta varia conforme a espécie. Assim, para garantir o desenvolvimento adequado de uma planta é necessário conhecer a sua espécie e o intervalo de tempo de exposição a luz ideal.

As plantas são categorizadas em três grandes grupos: plantas de dia longo, plantas de dia curto e as neutras (EMBRAPA, 2012). Na tabela 1 é apresentado o intervalo de tempo ideal de exposição à luz por categoria.

Tabela 1 – Tempo ideal de exposição a luz necessária por grupo de plantas.

Grupo	Tempo de Exposição a luz
Plantas de dia longo	10 a 12h
Plantas de dia curto	5 a 6h
Neutras	Indiferente

Fonte: Horas em pequenos Espaços – EMBRAPA (EMBRAPA, 2012).

3.3. Temperatura e Umidade do Ar

A temperatura da planta é muito próxima da temperatura do ambiente em que ela se encontra. Nos momentos em que a temperatura é mais elevada, o processo metabólico é mais acelerado e, nos períodos mais frios, o metabolismo tende a diminuir.

Além disso a temperatura tem um importante papel na realização da fotossíntese, uma vez que as reações bioquímicas envolvidas neste processo são dependentes da temperatura. Toda espécie de planta possui um limite máximo, ótimo e mínimo de temperatura para cada fase do seu crescimento. Como exemplo, na tabela 2 são apresentadas as temperaturas ideais para as fases de desenvolvimento da melancia. A maioria das plantas não resistem às temperaturas menores que 6°C (FERREIRA, SOUZA, *et al.*, 2014).

Tabela 2 – Faixas de temperatura do ambiente para a melancia em distintas fases de desenvolvimento.

Estágio do desenvolvimento	Temperatura(°C)
Germinação mínima	16
Floração ótima	20-21
Desenvolvimento ótimo	23-28
Maturação do fruto	23-28

Fonte: EMBRAPA (EMBRAPA, 2010).

Elevadas temperaturas associadas a elevados níveis de umidade do ar favorecem a incidência de doenças fúngicas nas plantas (BASTOS, PACHECO e FRAZÃO, 2002). A umidade relativa abaixo de 60% pode ser prejudicial por aumentar a taxa de transpiração das plantas e acima de 90% ocorre a redução de absorção dos nutrientes devido à redução da transpiração das plantas (BASTOS, PACHECO e FRAZÃO, 2002).

Por estes motivos, para a automação de um jardim, é importante monitorar a temperatura e a umidade relativa do ar, para melhor adequar os cuidados às necessidades da espécie cultivada.

4. Trabalhos Relacionados

O conceito de um irrigador automático é bastante popular no que se refere à automação residencial e no campo da agricultura de precisão, visto que o controle e a automação da irrigação é fundamental para a otimização dos recursos (FERNANDES, PREUSS e SILVA, 2017).

(DINIZ, 2017) propôs um sistema com software e hardware livres aplicado na medição de grandezas de umidade e temperatura do solo durante o ciclo das plantas, para permitir a otimização do uso da água no campo. Neste trabalho os sensores de umidade foram utilizados para obter a curva característica do solo, que pode ser usada para generalizar o comportamento do solo para diversos pontos de umidade e temperatura.

(KIM, LEE, *et al.*, 2015) utiliza sensores de umidade do solo, temperatura e pressão do solo para monitorar as operações de irrigação.

(BENNIS, FOUCHAL, *et al.*, 2015) desenvolveu um sistema que contém sensores de umidade e temperatura do solo conectados à uma rede sem fio. As informações obtidas a partir dos sensores acionam os atuadores. O sistema transmite os dados para um aplicativo da web.

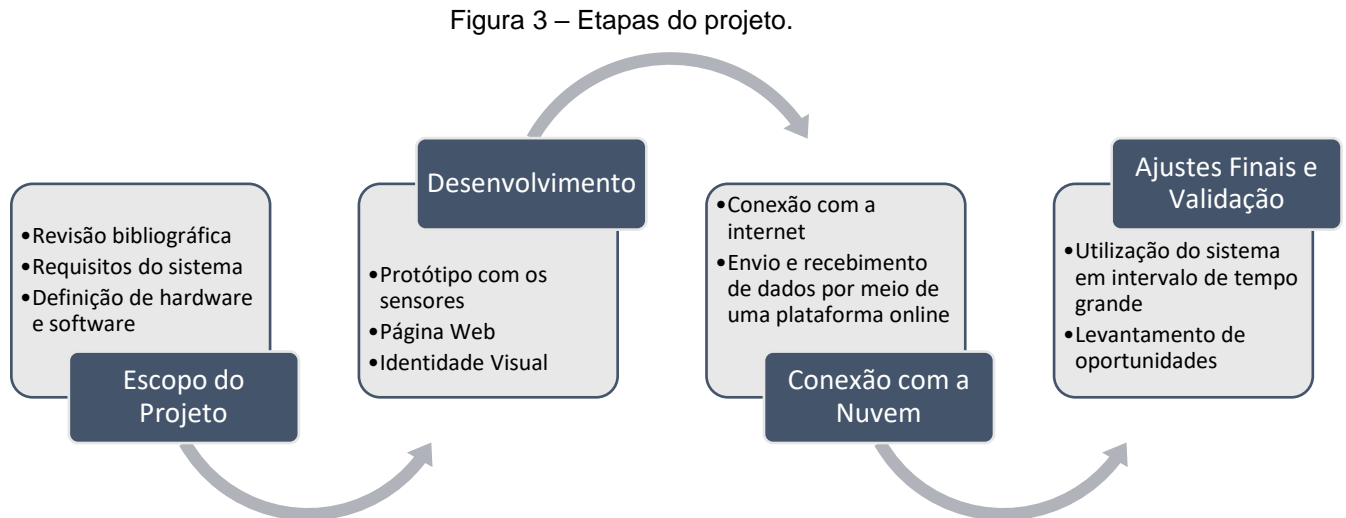
(CARVALHO, 2016) propõe um dispositivo de baixo custo para monitoramento da umidade do solo utilizando energia solar como fonte de energia. Para estabelecer a conexão com a internet é utilizado o módulo *wi-fi NRF24L01* conectado a um *Arduino*.

(GIRARDI, 2018) desenvolveu um sistema de monitoramento de umidade e temperatura por meio de sensores e um *Arduino*. O sistema foi desenvolvido para melhorar as práticas de cultivo de cogumelos *A. blazei*, que são fungos que requerem o monitoramento e controle constante dos fatores de cultivo para obtenção de sucesso na safra.

5. Materiais e Métodos

5.1. Etapas do Projeto

No diagrama da figura 3 são apresentadas as etapas de desenvolvimento do sistema proposto.



Fonte: Elaborado pela Autora.

5.2. Requisitos do Sistema

Para o desenvolvimento de um software ou um sistema, é necessário definir os requisitos, que são as descrições dos serviços propostos pelo sistema e suas restrições operacionais (SOMMERVILLE, 2011). Os requisitos definem como o sistema deve se comportar, o que deve fazer, e quais os recursos providos.

Neste trabalho, temos os seguintes requisitos:

- O sistema deve permitir a leitura da umidade do solo, temperatura, umidade relativa do ar e luminosidade do local a ser monitorado;
- Todas as medições obtidas devem ser informadas para o usuário de forma visual;
- Os controles do sistema devem ser feitos via página *web*;
- Os dados deverão ser armazenados em nuvem;
- O sistema deve ser capaz de, sem a intervenção direta do usuário, acionar a irrigação no momento em que a umidade do solo atingir o nível definido como “solo seco” ou após um determinado tempo desde a última irrigação (*timer*).
- O sistema deve ser capaz de, sem a intervenção direta do usuário, interromper a irrigação no momento em que a umidade do solo atingir o nível de umidade do solo definido como “solo úmido” (modo umidade) ou após um determinado tempo desde o

início do acionamento da irrigação (modo *timer*), isto é, no modo *timer*, o sistema conta o tempo que a bomba fica ligada.

- O sistema deve permitir o acompanhamento dos momentos em que o acionamento da água foi feito.

5.3. Materiais Utilizados

Para desenvolvimento do protótipo foram utilizados os materiais presentes na tabela

3.

Tabela 3 – Lista de materiais utilizados no projeto.

Item
Arduino MEGA 2560
Arduino UNO
Arduino Ethernet Shield W5100
Higrômetro FC-28
LDR 5 mm
Sensor de temperatura e umidade DHT22
Protoboard e fios para conexões
Bomba d'água JT100 5V
Mangueira de plástico de 1 cm
Resistores e leds variados

Fonte: Elaborada pela Autora.

5.3.1. Arduino

A facilidade de conectar diferentes tipos de sensores periféricos nos microcontroladores, contribui para a utilização destes dispositivos eletrônicos no desenvolvimento de aplicações *IoT* (SANTOS, 2014). Além disso, também são dispositivos que podem ser conectados facilmente na internet. Alguns microcontroladores, como *NodeMCU* já possuem um módulo *wi-fi* integrado. Neste projeto, optou-se por utilizar o microcontrolador Arduino, que também é bastante popular em aplicações de *IoT*.

O *Arduino* é uma plataforma *open-source* para protótipos eletrônicos, que une *hardware* e *software*. A comunicação entre o *Arduino* e o computador ocorre via porta serial, através de um cabo *USB*. Com o *Arduino* é possível conectar facilmente vários tipos de sensores e colocar os dados lidos destes sensores em suas saídas. Para o

desenvolvimento de uma aplicação, utiliza-se a linguagem de programação *Arduino*, cuja interface de desenvolvimento *Arduino IDE* é escrita em linguagem *Java*. O *Arduino IDE* possui uma janela chamada *Monitor Serial* que auxilia no recebimento e envio de dados sem a necessidade de recorrer a uma ferramenta externa.

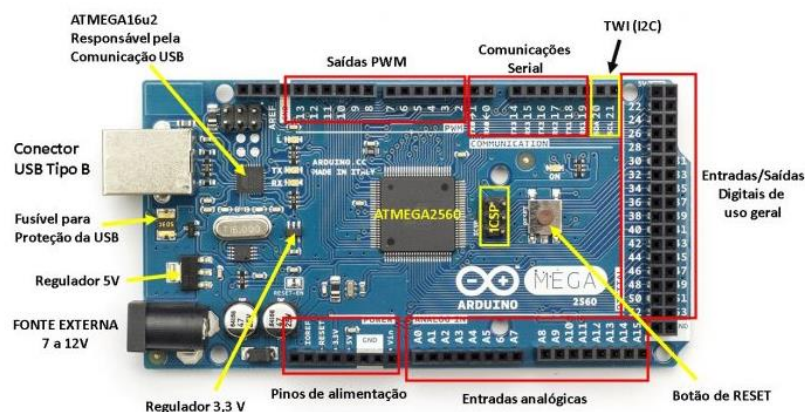
O *Arduino* tem sido utilizado em vários projetos devido à facilidade de adaptá-lo, tanto para projetos simples como para aplicações avançadas. Pode-se conectar o *Arduino* a vários tipos de periféricos como: displays, botões, sensores, módulos *Ethernet* ou *Wi-Fi*, utilizando bibliotecas específicas para cada tipo de dispositivo periférico.

Inicialmente foi utilizado no projeto o *Arduino UNO*, que possui 32 kB de memória flash, 2 kB de *RAM* e 1kB de *EEPROM*. Este modelo possui 14 pinos digitais que operam em 5V e oferecem até 40 mA de corrente que podem ser configurados como entradas ou saídas. O *Arduino UNO* também possui 6 portas analógicas que fazem a conversão do sinal analógico para o digital com uma resolução de até 10 bits (0-1023).

Durante o desenvolvimento detectou-se que o *Arduino UNO* não atendeu os requisitos de memória necessários para o envio de dados para a plataforma *ThingSpeak*. Desta forma, foi necessária a troca do *Arduino UNO* por um *Arduino MEGA 2560*, que atendeu aos requisitos de memória de escrita e administração de todos os periféricos presentes.

O *Arduino MEGA 2560*, que possui 256 kB de memória *flash*, 8 kB de *RAM* e 4 kB de *EEPROM* (ARDUINO, 2019). Além disso, ele possui 54 pinos digitais que operam em 5V e oferecem até 40 mA de corrente que podem ser configurados como entradas ou saídas. O *Arduino Mega 2560* também possui 16 entradas analógicas que fazem a conversão A/D. Na figura 4 é exibido um *Arduino MEGA 2560* e a pinagem de seus conectores.

Figura 4 – *Arduino MEGA 2560*.

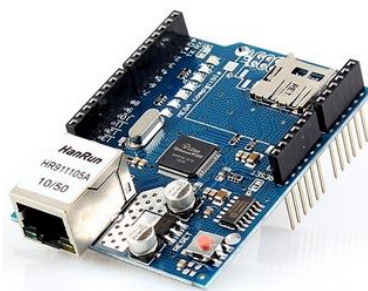


Fonte: Embarcados (EMBARCADOS, 2019)

O *Arduino Mega 2560* pode ser conectado a um computador e à Internet para buscar e enviar dados e atuar sobre eles. Por exemplo, ele pode mandar um lote de dados obtidos de sensores para um site, dados estes que podem ser exibidos em forma de gráfico (FERNANDES, PREUSS e SILVA, 2017), como implementado neste projeto.

A conexão com a internet foi feita utilizando um *Ethernet Shield W5100* (Figura 5) conectado ao *Arduino*, que fornece acesso à rede utilizando a biblioteca *Ethernet Library*.

Figura 5 – *Ethernet Shield W5100*

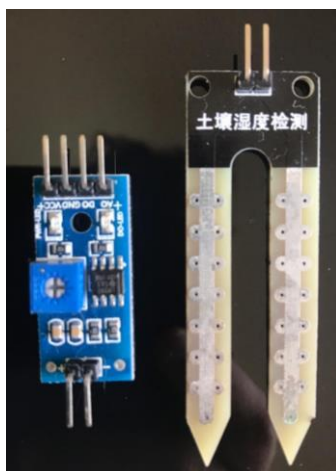


Fonte: FilipeFlop (FILIPFLOP)

5.3.2. Medição de Umidade do Solo

Para obter a medição da umidade do solo, foi utilizado o método da resistência elétrica, que é feito por meio da aplicação de uma sonda (higrômetro) com dois eletrodos que são enterrados no solo. O método consiste em monitorar a variação da tensão entre os dois eletrodos enterrados, uma vez que a tensão é inversamente proporcional à umidade do solo (GIRARDI, 2018). À direita, na figura 6, é exibida a sonda e à esquerda, o módulo eletrônico utilizado para transmitir para o *Arduino* a medida analógica de umidade feita pela sonda.

Figura 6 – Higrômetro F-28 e módulo de leitura.

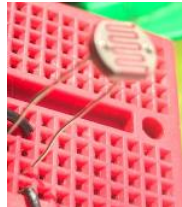


Fonte: Elaborado pela Autora.

5.3.3. Medição de Luminosidade

A medida de luminosidade pode ser obtida utilizando-se um circuito divisor de tensão com um fotoresistor (*LDR – Light Dependent Resistor*), exibido na figura 7, cuja resistência varia com a intensidade de luz recebida.

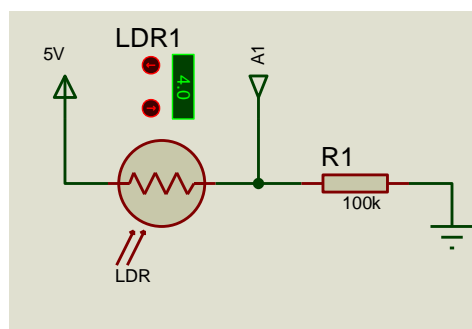
Figura 7 – Light Dependent Resistor



Fonte: Elaborado pela Autora.

Foi montado o circuito divisor de tensão exibido no esquema da figura 8. Para obter a medição da luminosidade, conectou-se a porta analógica do pino A1 do Arduino.

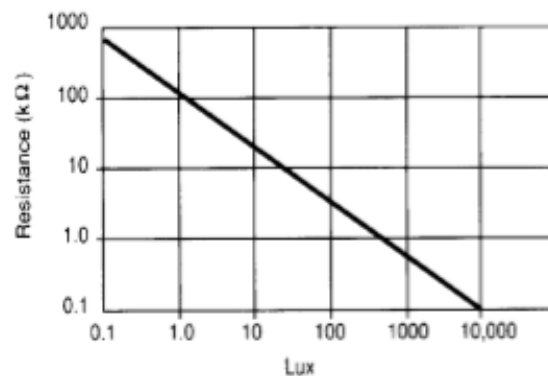
Figura 8 – Esquema do Circuito Responsável pela obtenção da luminosidade.



Fonte: Elaborado pela Autora.

A variação da resistência em função da luminosidade que incide sobre a superfície do LDR é exibida na figura 9.

Figura 9 – Curva de resistência do LDR em função da luminosidade



Fonte: Datasheet do LDR (SUNROM, 2019).

5.3.4. Medição de Temperatura e Umidade Relativa do Ar

Para obtenção das medidas de temperatura e umidade relativa do ar utilizou-se o sensor digital *DHT22*, cujas características elétricas e de precisão são exibidas na tabela 4.

Tabela 4 – Características do DHT22 retiradas do datasheet.

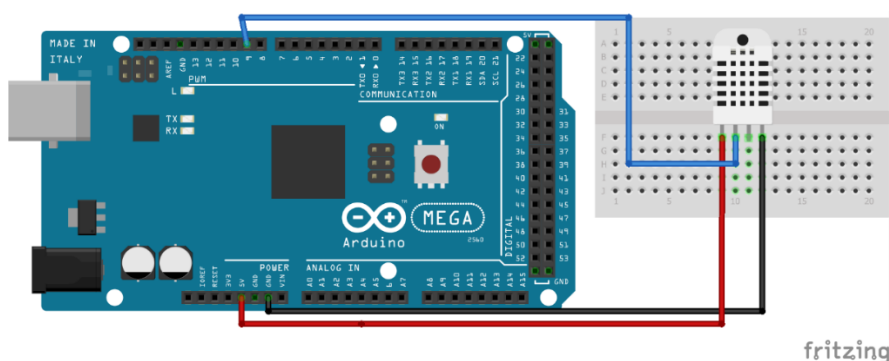
Alimentação	3,3V – 6V
Faixa de Leitura – Umidade	0 – 100%
Precisão – Umidade	5%
Faixa de Leitura – Temperatura	-40 – 125 °C
Precisão – Temperatura	+/- 0.5 °C
Intervalo entre medições	2 s

Fonte: Datasheet do fabricante (DHT 22 Datasheet).

O *DHT22* fornece a medida de temperatura em graus Celsius (°C) e a medida de umidade relativa do ar em porcentagem de umidade na atmosfera (%), não sendo necessária a calibração do sensor.

O sensor *DHT22* possui 4 pinos, sendo um pino de alimentação de 3.3-6V, um pino para a transferência dos dados, um pino para conexão com o terra (0V) e um pino sem função. Para utilizar o *DHT22* é preciso incluir a biblioteca *DHT.h* na programação do *Arduino* e a ligação deve ser feita conforme o esquema da Figura 10.

Figura 10 – Esquema de ligação do DHT22 com o Arduino.



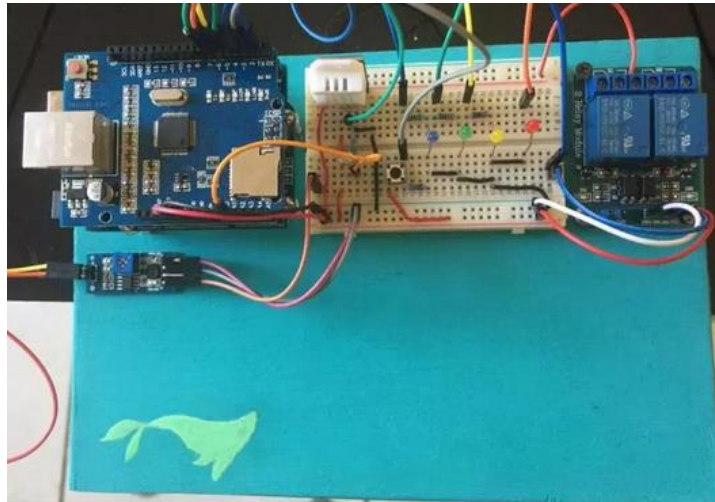
Fonte: Elaborado pela Autora no software *Fritzing*.

indicar a presença ou não de luz de forma visual ao usuário.

5.4.2. Evolução do protótipo

Na figura 12 é exibido a primeira versão do protótipo da estação local desenvolvido com o Arduino UNO. O *led* azul foi adicionado para indicar quando a bomba é acionada.

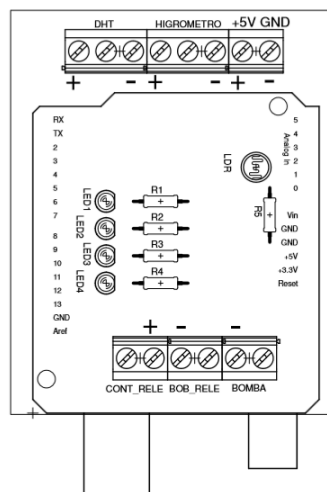
Figura 12 – Primeira versão do protótipo da estação local utilizando Arduino UNO.



Fonte: Autoria Própria

Para aumentar a confiabilidade do protótipo, foi confeccionada uma placa de circuito impresso. Na figura 13 é apresentado o desenho do posicionamento dos componentes da placa gerado com o software *Eagle*.

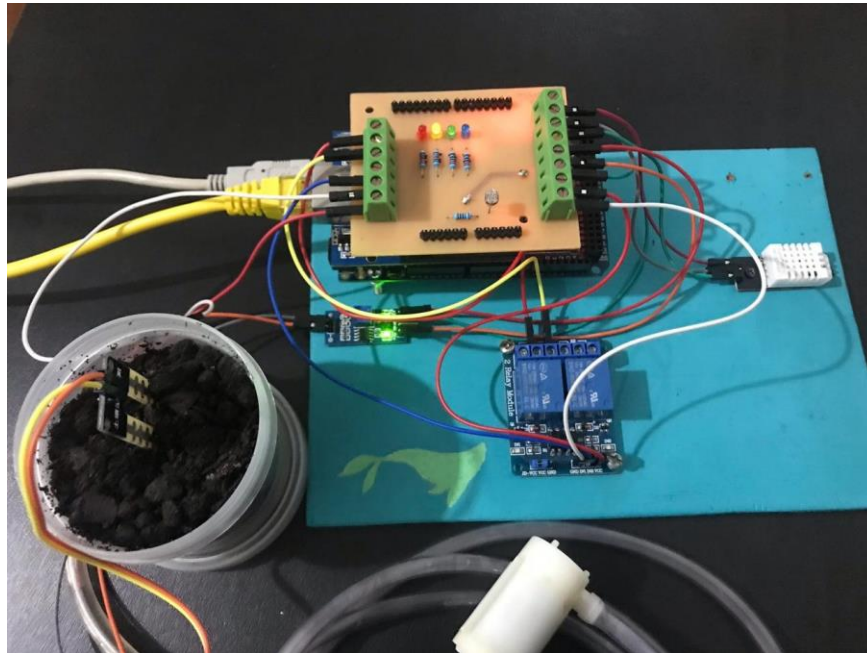
Figura 13 – Desenho do posicionamento dos componentes da placa.



Fonte: Autoria Própria

O protótipo final desenvolvido é exibido na Figura 14.

Figura 14 - Protótipo final da estação local.



Fonte: Autoria Própria.

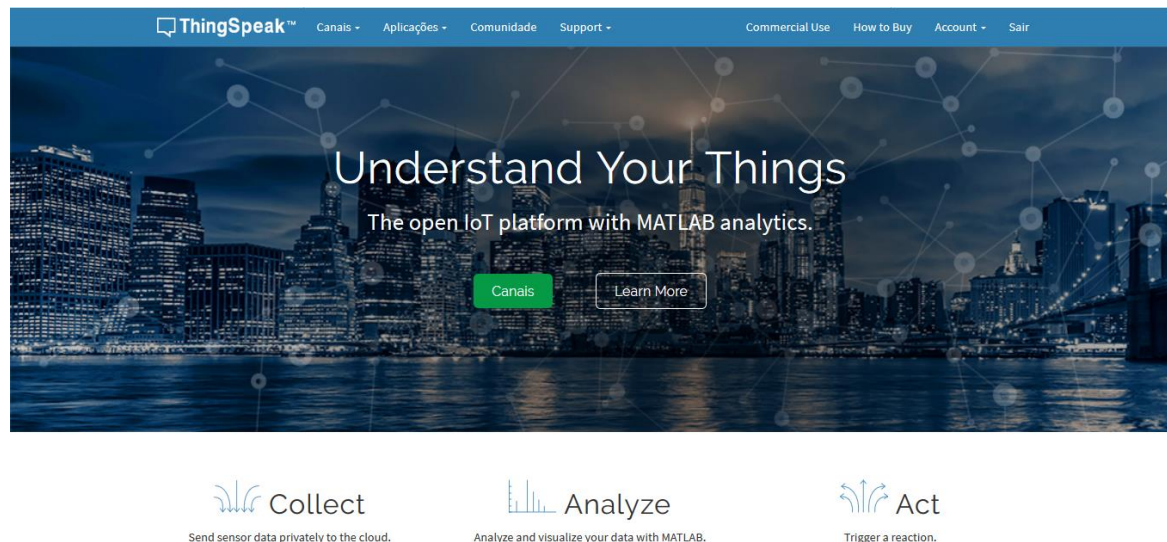
O código fonte da estação local no Anexo A.

5.5. ThingSpeak

Para armazenar em nuvem os dados obtidos pelos sensores, neste trabalho, utilizou-se a plataforma *ThingSpeak*. O *ThingSpeak* é um aplicativo e API (*Application Programming Interface*) *opensource* gratuito de *Internet of Things (IoT)* que utiliza o protocolo *HTTP*. Uma API é um conjunto de instruções de programação para acesso a um software (MAUREIRA, OLDENHOF e TEERNSTRA, 2014).

O *ThingSpeak* permite registrar dados obtidos por sensores por meio de uma interface amigável que, além do armazenamento, permite a visualização dos dados por meio de *widgets*. *Widgets* são elementos gráficos de interação com o usuário tais como janelas, botões, barras de rolagem, menus, entre outros. A figura 15 apresenta a página inicial do *ThingSpeak*.

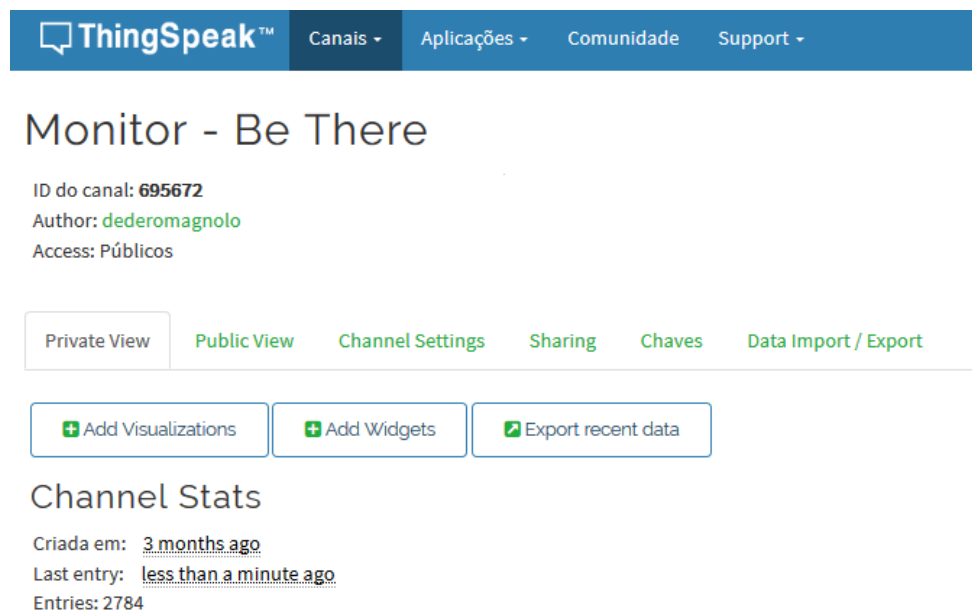
Figura 15 – Página Inicial do *ThingSpeak*.



Fonte: (THINGSPEAK, 2019)

Para utilizar as funcionalidades do *ThingSpeak* no *Arduino* é necessário declarar a biblioteca *ThingSpeak.h* na programação do *Arduino*. Todo canal possui um número de identificação (ID) que é utilizado para estabelecer comunicação, como exibido na figura 16.

Figura 16 – Painel de controle do canal elaborado pelo sistema.

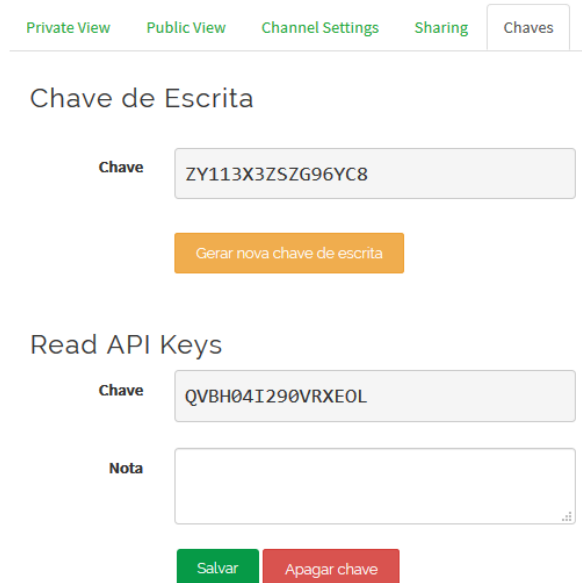


Fonte: Elaborado pela Autora.

Além disso, o canal possui uma chave de escrita e uma chave de leitura únicas, que são usadas na programação do *Arduino* e da página *web* para executar a ação de leitura ou escrita de dados durante uma conexão com o canal. As chaves do canal criado para a

utilização do sistema deste trabalho são exibidas na Figura 17.

Figura 17 – Chaves de escrita e leitura do canal.



Private View Public View Channel Settings Sharing Chaves

Chave de Escrita

Chave ZY113X3ZSZG96YC8

Gerar nova chave de escrita

Read API Keys

Chave QVBH04I290VRXEOL

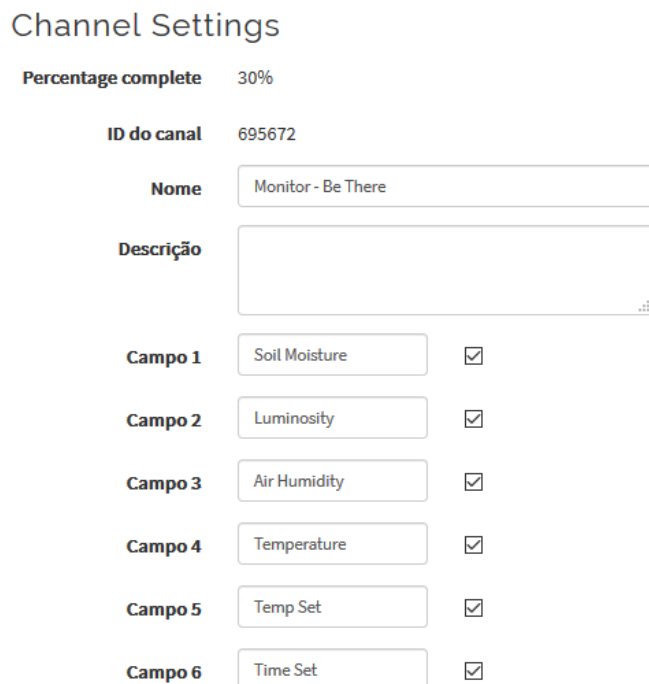
Nota

Salvar Apagar chave

Fonte: Elaborado pela Autora.

No menu “*Channel Settings*” do painel de controle do canal do *ThingSpeak*, foram configurados os campos para cada medição enviada pelo *Arduino* (Figura 18).

Figura 18 – Configurações de campos do canal Be There.



Channel Settings

Percentage complete 30%

ID do canal 695672

Nome Monitor - Be There

Descrição

Campo 1 Soil Moisture ☒

Campo 2 Luminosity ☒

Campo 3 Air Humidity ☒

Campo 4 Temperature ☒

Campo 5 Temp Set ☒

Campo 6 Time Set ☒

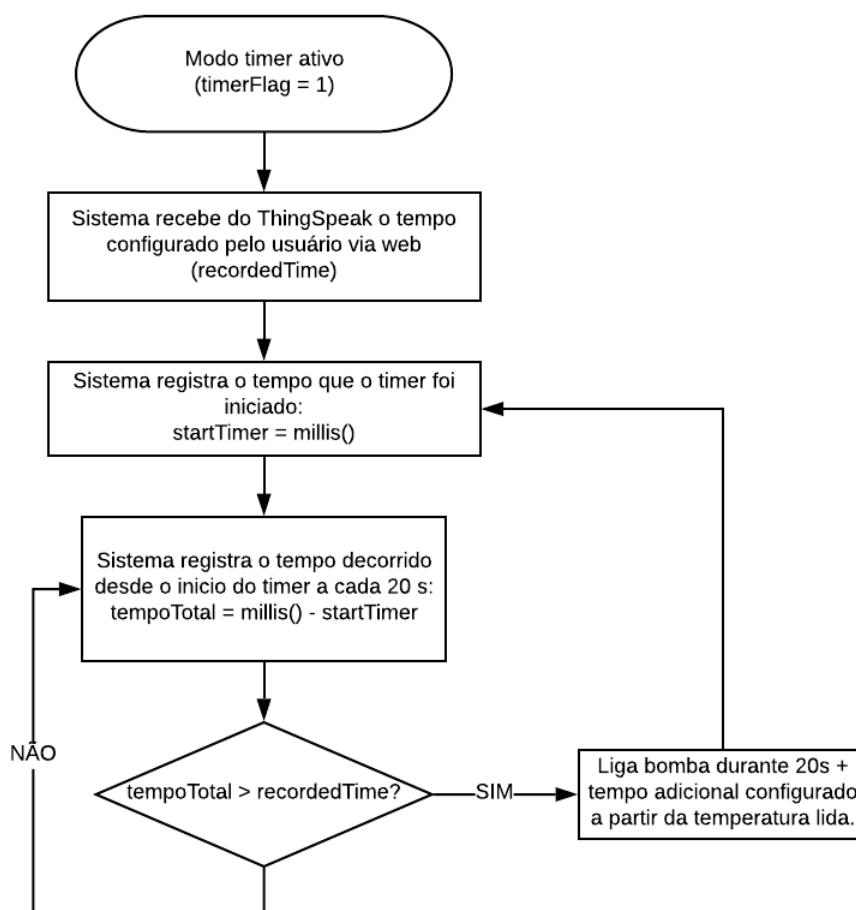
Fonte: Elaborado pela Autora.

Os campos 1,2,3 e 4 foram configurados para armazenar as medições de umidade do solo, luminosidade, umidade relativa do ar e temperatura, respectivamente.

O campo *Temp Set* foi configurado para receber um valor de temperatura fornecido pelo usuário por meio da página web do sistema, como mostrado na Figura 1. Este valor é utilizado no modo *Timer* para configurar a partir de qual temperatura o usuário deseja que a bomba permaneça ligada por tempo adicional.

O campo *Time Set* recebe o valor “1” (ligado) ou “0” desligado por meio do botão *Timer* presente no menu “*Settings*” da página web. Na figura 19 é exibido um fluxograma para ilustrar o funcionamento do “*Modo Timer*”, utilizando a função *millis()* do *Arduino*.

Figura 19 – Fluxograma – Modo Timer



Fonte: Elaborada pela Autora.

Para envio de múltiplas informações do *Arduino* para o *ThingSpeak*, é preciso atribuir cada medição coletada pelos sensores aos respectivos campos, por meio da função *setField()*, presente na biblioteca do *ThingSpeak* para *Arduino*. O envio dos campos

configurados com o *setField()* é feito utilizando o comando *writeFields()*, conforme exibido no código da figura 20.

Figura 20 – Trecho do código com as funções *setField()* e *writeFields()* utilizadas para enviar as medições e status da bomba para o ThingSpeak.

```
//set fields
ThingSpeak.setField(1, umidade);
ThingSpeak.setField(2, luminosidade);
ThingSpeak.setField(3, umidadeAr);
ThingSpeak.setField(4, temperatura);

//write fields
int x = ThingSpeak.writeFields(mainChannelNumber,myWriteAPIKey);
```

Fonte: Elaborado pela Autora.

Para detectar uma possível falha de conexão com o canal, a função *writeFields()* retorna um código de erro. Foi implementada uma rotina para informar o usuário caso seja detectada alguma falha na conexão ou durante a tentativa de escrita no canal. Na figura 21 são exibidos os possíveis erros do *ThingSpeak* durante a execução de uma função.

Figura 21 – Códigos de erro e o significado retornados pelas funções do ThingSpeak

Return Codes

Value	Meaning
200	OK / Success
404	Incorrect API key (or invalid ThingSpeak server address)
-101	Value is out of range or string is too long (> 255 characters)
-201	Invalid field number specified
-210	setField() was not called before writeFields()
-301	Failed to connect to ThingSpeak
-302	Unexpected failure during write to ThingSpeak
-303	Unable to parse response
-304	Timeout waiting for server to respond
-401	Point was not inserted (most probable cause is the rate limit of once every 15 seconds)
0	Other error

Fonte: (THINGSPEAK, 2019)

5.6. Página Web

Para controlar e monitorar a estação local pela nuvem foi desenvolvida uma página *web*. Esta página se conecta à plataforma *ThingSpeak* para possibilitar o envio de comandos à estação local e para realizar a leitura dos dados que estejam em nuvem.

A página *web* foi desenvolvida com o editor gratuito e *open-source* *Brackets* com a linguagem de programação *html/css*. As funções da página foram desenvolvidas utilizando scripts na linguagem *Javascript* e o *framework JQuery*. Para possibilitar o acesso à página desenvolvida utilizando internet foi utilizado o servidor gratuito *InfinityFree*.

Também foi desenvolvida uma identidade visual para o sistema proposto utilizando-se o software de edição de imagem *Adobe Photoshop*, exibida na figura 22.

Figura 22 – Identidade visual do sistema.



Fonte: Autoria própria

Foi desenvolvido o menu da página com as seguintes opções de seleção:

- ***Dashboard:*** Exibe ao usuário o status resumido do solo. Também contará com um *widget* de clima que informará a temperatura e o tempo (chuvoso, nublado ou com sol) do dia atual e da semana.
- ***Charts:*** Neste menu estão dispostos os gráficos das medições de temperatura, umidade relativa do ar, umidade do solo e luminosidade presentes no *ThingSpeak*. Os gráficos oferecem a informação em tempo real, isto é, conforme a estação local envia os dados ao *ThingSpeak*, na página os dados são exibidos no mesmo momento.
- ***Library:*** O usuário pode criar perfis para suas plantas e armazenar informações sobre as mesmas. Este menu não foi implementado neste trabalho devido à

complexidade da programação e por não influenciar no funcionamento do sistema de irrigação automática proposto neste trabalho.

- **Settings:** Neste menu são disponibilizados botões para configurar o modo de operação da estação local que são:

Modo *timer*: A irrigação é feita em intervalos de tempo definidos, independentemente da umidade do solo. A bomba permanece ligada por 20 segundos a cada loop do timer (tempo padrão). O tempo em que a bomba fica ligada recebe um adicional de tempo se a temperatura lida pela estação local for maior que temperatura máxima configurada pelo usuário. Quanto maior a diferença ($\Delta Temp$) entre a temperatura configurada e a temperatura lida, maior o tempo que a bomba permanece ligada.

Modo umidade do solo: A irrigação é feita quando o solo estiver seco, de acordo com a calibração da seção 5.4.1 e somente quando houver luz no ambiente.

O código *html* e o código *css* da página desenvolvida estão nos anexos B e C, respectivamente.

6. Resultados e Discussão

A tabela 5 exibe os valores digitalizados das tensões enviadas pelo módulo eletrônico do higrômetro para cada status do solo. O status do solo foi modificado pela adição de água, conforme descrito na seção 5.4.1.

Tabela 5 – Valores digitalizados de tensão enviadas pelo módulo eletrônico do higrômetro em função da quantidade de água adicionada na amostra de solo

Medição do Higrômetro (0-1023)	Status da amostra do solo
43	Encharcado
171	Úmido
278	Úmido
286	Úmido
303	Parcialmente úmido
354	Parcialmente úmido
385	Parcialmente úmido
426	Parcialmente úmido
626	Seco
752	Seco
940	Seco
1019	Seco

Fonte: Elaborado pela Autora.

A tabela 6 exibe os intervalos correspondentes a cada um dos status do solo. Para indicação visual, foram usados *leds* das cores verde e vermelho.

Tabela 6 – Status do solo em função da medição do higrômetro.

Status do Solo	Limites	Cor do Led
Úmido	$626 \geq \text{Umidade}$	Verde
Seco	$\text{Umidade} \geq 626$	Vermelho

Fonte: Elaborado pela Autora.

Para detectar a presença de luz no ambiente, variou-se a luz incidente no *LDR* para registrar os valores digitalizados de tensão obtidos pelo sensor LDR para diferentes níveis de luminosidade (Tabela 7).

Tabela 7 – Valores digitalizados de tensão do sensor LDR para diferentes níveis de luminosidade.

Medição do LDR (0-1023)	Status do ambiente
110	Muita luz
250	Luz incidente
450	Luz moderada
612	Pouca luz
722	Pouca luz
891	Sem luz
1012	Sem luz

Na tabela 8 foram estabelecidos os limites para indicar a presença de luz utilizando os dados da tabela 7.

Tabela 8 – Limites para indicar a presença de luz.

Status do ambiente	Limites
Com luz	<i>LDR</i> < 891
Sem luz	<i>LDR</i> ≥ 891

A figura 23 exibe a captura da tela do *Monitor Serial* do *Arduino IDE* quando o *loop* de medição é executado, que acontece a cada 1 minuto.

Figura 23 – Medições exibidas no Monitor Serial do Arduino IDE.

```

----- BE THERE - REPORT -----
----- Soil Status -----
Soil Moisture: 69%
Luminosity: 4%
Your soil is moist and your garden has no light

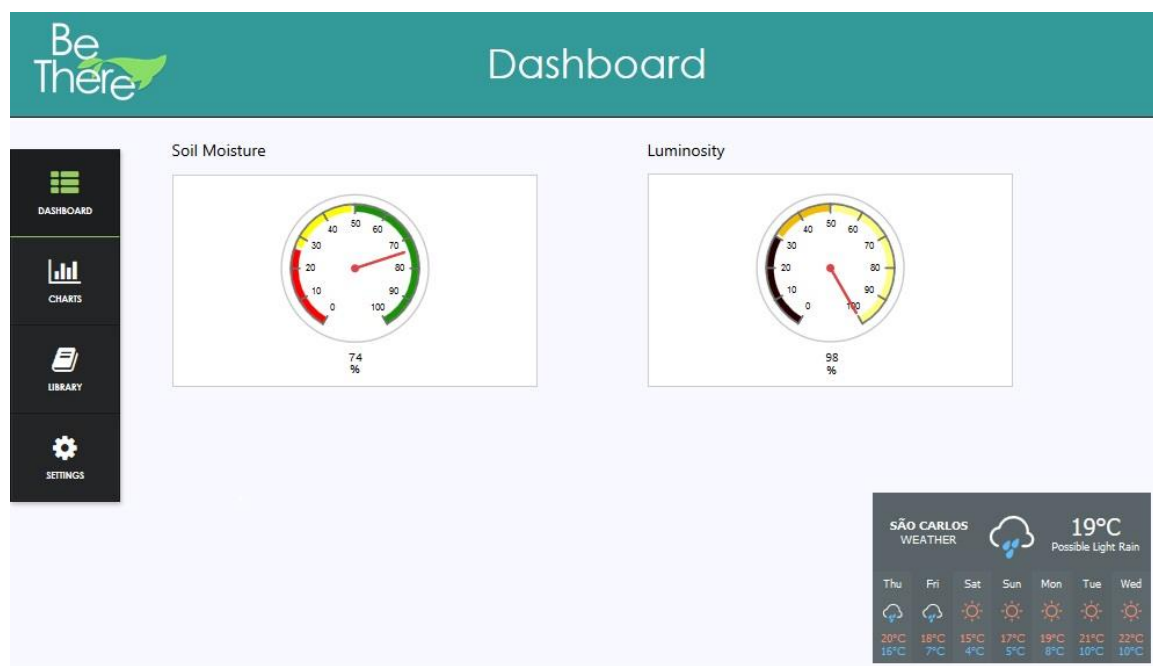
----- Weather Status -----
Humidity: 75.60%
Temperature: 22.90°C

```

Fonte: Autoria Própria.

A tela inicial “*Dashboard*” da página *web* desenvolvida é exibida na figura 24. Esta página possibilita o controle e monitoramento da estação local através da nuvem.

Figura 24 – Tela inicial Dashboard

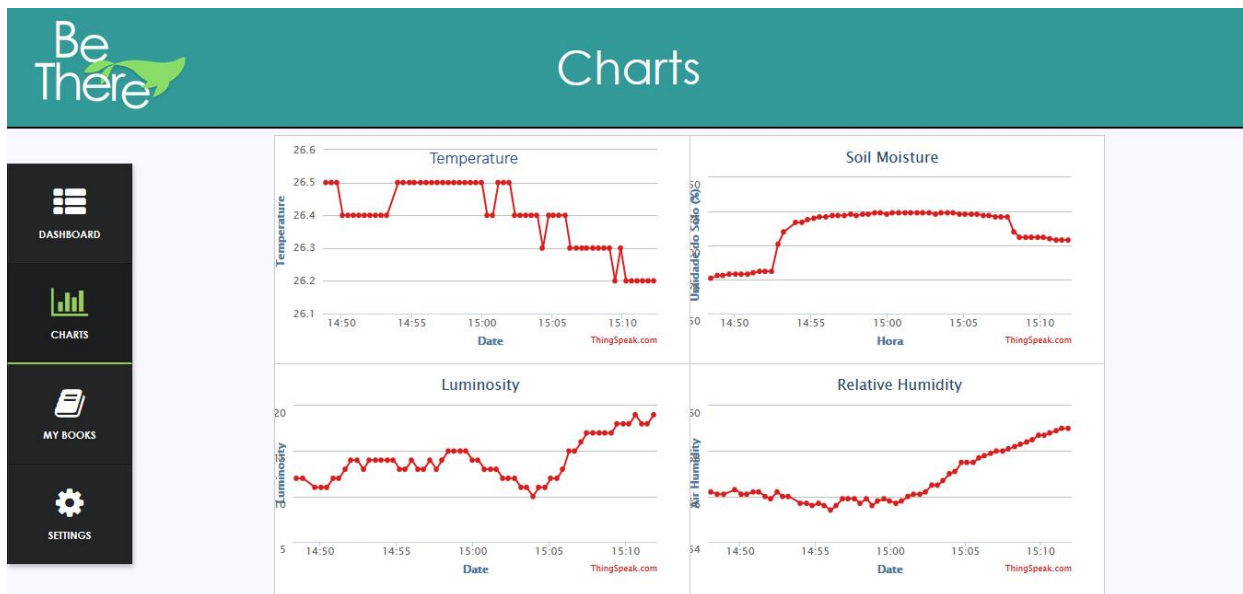


Fonte: Autoria Própria.

A tela “*Dashboard*” possui um conceito semelhante à tela “*Library*”, que diz respeito à experiência do usuário e sua interação com o objeto inteligente. Como o foco deste trabalho é o desenvolvimento da estação local e seu controle remoto utilizando o *ThingSpeak*, o desenvolvimento elaborado destas telas foi deixado para trabalhos futuros.

A tela “*Charts*” é exibida na figura 25. Os gráficos desta tela são *widgets* do *ThingSpeak*.

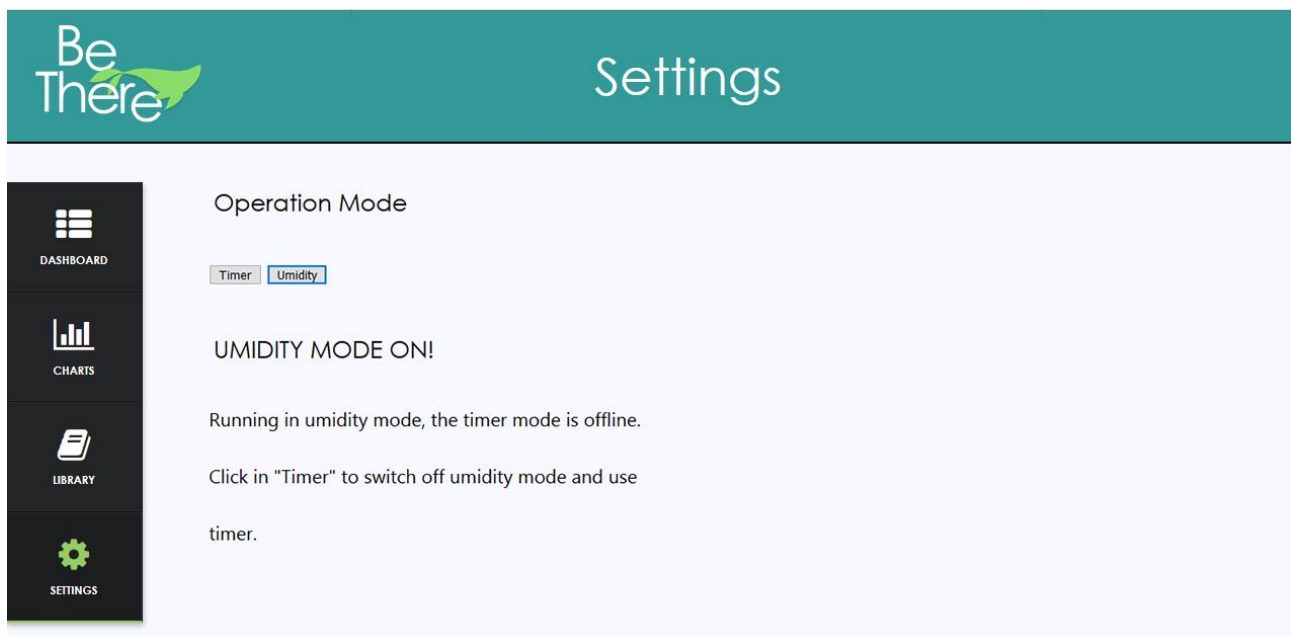
Figura 25 – Tela “Charts”



Fonte: Autoria Própria

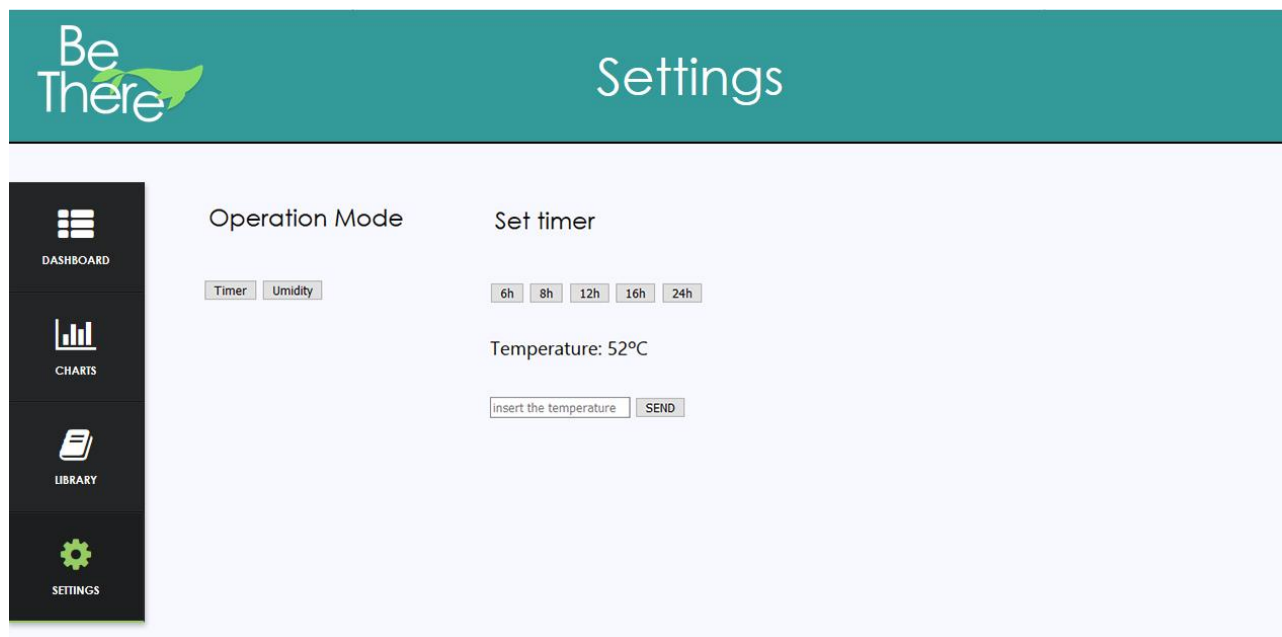
A tela “Settings” contém dois botões de seleção do modo de operação da estação local. A figura 26 exibe a tela “Settings” para o Modo Umidade e a figura 27 exibem a tela “Settings” para o Modo Timer.

Figura 26 - Tela “Settings” exibida no Modo Umidade.



Fonte: Autoria Própria.

Figura 27 – Tela “Settings” exibida no Modo *Timer*.



Fonte: Autoria Própria.

A figura 28 mostra a captura de tela do *Monitor Serial* quando o modo *timer* é ativado.

Figura 28 – Captura de Tela do *Monitor Serial* quando o modo *timer* está ligado

```
00:09:21.714 -> ----- BE THERE - REPORT -----  
00:09:21.794 -> Timer Mode On!  
00:09:21.794 -> Umidity Mode Off!  
00:09:21.794 -> ----- Soil Status -----  
00:09:21.834 -> Soil Moisture: 83%  
00:09:21.874 -> Luminosity: 84%  
00:09:21.874 -> Your soil is moist and your garden has incident light  
00:09:21.914 ->  
00:09:21.914 -> ----- Weather Status -----  
00:09:21.954 -> Humidity: 71.30%  
00:09:21.994 -> Temperature: 23.70°C  
00:09:21.994 ->  
00:09:23.514 -> Data sent with success!
```

Fonte: Autoria Própria.

A figura 29 mostra a captura de tela do *Monitor Serial* quando o Modo Umidade está ativo.

Figura 29 – Captura de Tela do *Monitor Serial* quando o modo Umidade está ativo.

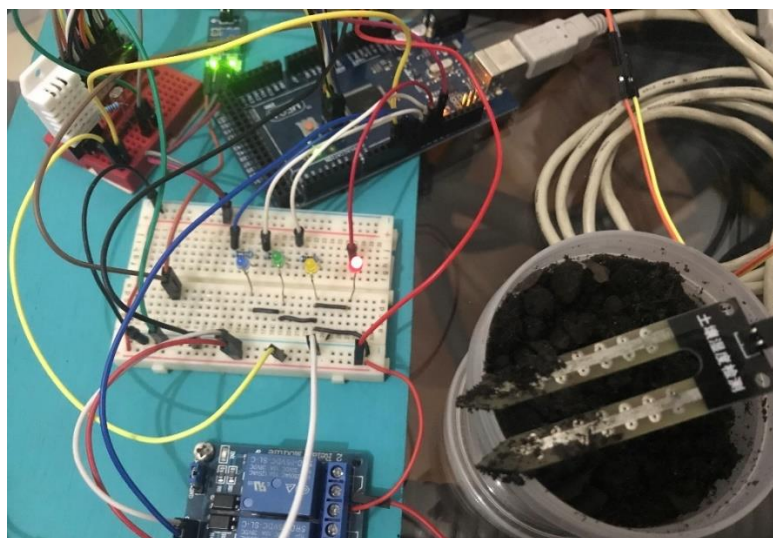
```
00:52:44.947 -> ----- BE THERE - REPORT -----  
00:52:44.986 -> Timer Mode Off!  
00:52:44.986 -> Umidity Mode On!  
00:52:45.026 -> ----- Soil Status -----  
00:52:45.026 -> Soil Moisture: 1%  
00:52:45.066 -> Luminosity: 9%  
00:52:45.066 -> Your soil is dry and your garden has no light  
00:52:45.106 ->  
00:52:45.106 -> ----- Weather Status -----  
00:52:45.146 -> Humidity: 53.40%  
00:52:45.187 -> Temperature: 24.80°C  
00:52:45.187 ->  
00:52:46.882 -> Data sent with success!
```

Fonte: Autoria Própria.

A estação local possui 4 *leds* indicadores. O *led* vermelho e o *led* verde são indicadores de umidade do solo. O *led* vermelho aceso indica que o solo está seco. O *led* verde aceso indica que o solo está úmido. Somente um deles estará aceso em cada instante. O *led* amarelo é o indicador de luz no ambiente. O *led* amarelo aceso indica que existe luz no ambiente e apagado que não existe luz no ambiente. O *led* azul é o indicador da bomba ligada. O *led* azul aceso indica que a bomba está ligada e apagado que a bomba está desligada.

Na figura 30 é exibida uma das situações possíveis. O *led* vermelho aceso indica o solo seco. A princípio, a bomba deveria estar ligada nesta situação (*led* azul ligado) porém como o *led* amarelo está apagado, indica que o ambiente não tem luz e por isso a bomba não é ligada (*led* azul apagado).

Figura 30 – Estação local no modo umidade quando o solo está seco e o ambiente sem luz. A bomba não é acionada.



Fonte: Autoria Própria.

7. Trabalhos Futuros

A partir dos resultados obtidos neste trabalho, sugere-se:

- Utilizar um módulo *RTC* (*real time clock*) para melhorar a temporização do sistema ao invés de usar a função *millis()* do *Arduino*;
- Utilizar um cartão *SD* para armazenamento temporário dos dados em caso de perda de conexão com a *internet*;
- Melhorar a interface gráfica da tela “*Dashboard*”;
- Implementar um botão de *reset* da estação local na tela “*Dashboard*”;
- Implementar uma função que verifica se a estação local está *online* na tela “*Dashboard*”;
- Desenvolvimento da tela “*Library*” para permitir a criação de um banco de parâmetros para diferentes espécies de plantas;

8. Conclusão

O protótipo desenvolvido atingiu os objetivos propostos. Observou-se que a sonda utilizada para obtenção da umidade do solo não se mostrou eficiente para aplicações de longo prazo, devido à rápida oxidação dos seus eletrodos. Para aplicações de longo prazo é necessário utilizar uma sonda mais resistente aos efeitos corrosivos do solo.

Uma limitação encontrada com a utilização do *ThingSpeak* tem relação com o tempo de resposta do sistema aos comandos externos. O *ThingSpeak* necessita de pelo menos 15 segundos entre uma conexão e outra, tanto na função *read* como *write*, o que limita o envio de comandos para uma mudança imediata no sistema. Para aplicações menos robustas o *ThingSpeak* é adequado pois a plataforma é de fácil configuração e possui uma interface amigável com opção de adicionar gráficos e *widgets*.

9. Referências

- ALMEIDA, L. I.; COSTA, C. M.; FERNANDES, I. F. C. **SISCI - Sistema para Controle de Irrigação Através de Dispositivos Celulares**. [S.l.]: [s.n.], 2011.
- ARDUINO. Arduino Mega 2560, 2019. Disponível em: <<https://store.arduino.cc/usa/mega-2560-r3>>. Acesso em: 5 Maio 2019.
- ARUBA. IoT Heading for Mass Adoption by 2019 Driven by Better-Than-Expected Business Results, 2019. Disponível em: <<https://news.arubanetworks.com/press-release/arubanetworks/iot-heading-mass-adoption-2019-driven-better-expected-business-results>>. Acesso em: 14 Fevereiro 2019.
- BASTOS, T. X.; PACHECO, N. A.; FRAZÃO, D. A. C. **Aptidão Climática das Principais Espécies de Fruteiras Tropicais Cultivadas na Amazônia**. [S.l.]: Embrapa, 2002.
- BENNIS, I. et al. **Drip Irrigation System using Wireless Sensor Networks**. [S.l.]: Proceedings of the Federated Conference on Computer Science and Information Systems, 2015.
- CARVALHO, M. S. D. **Sensor para monitoramento de umidade do solo utilizando energia solar**. Quixadá: [s.n.], 2016.
- CORRÊA, M. R. O.; GOMES, E. L. B. **Sensores Aplicados no Controle da Umidade do Solo para o Cultivo da Cenoura**. [S.l.]: [s.n.], 2016.
- DHT 22 Datasheet. Disponível em: <<https://www.sparkfun.com/datasheets/Sensors/Temperature/DHT22.pdf>>. Acesso em: 22 Junho 2019.
- DIAS, L. B. **Água nas Plantas**. [S.l.]: Universidade Federal de Lavras, 2008.
- DINIZ, A. M. **Sistema Automatizado de Aquisição, em Tempo Real, de Umidade e Temperatura do Solo na Irrigação**. Cascavel: [s.n.], 2017.
- EMBARCADOS. Saiba Mais Sobre o Arduino Mega 2560, 2019. Disponível em: <<https://www.embarcados.com.br/arduino-mega-2560/>>. Acesso em: 22 Junho 2019.
- EMBRAPA. Sistema de Produção de Melancia, 2010. Disponível em: <<https://sistemasdeproducao.cnptia.embrapa.br/FontesHTML/Melancia/SistemaProducaoMelancia/clima.htm>>. Acesso em: 01 Julho 2019.
- EMBRAPA. **Hortas em Pequenos Espaços**. [S.l.]: [s.n.], 2012.
- ERICSSON Mobility Report: On the Pulse of Network Society, Junho 2016. Disponível em: <<https://www.ericsson.com/assets/local/mobility-report/documents/2016/Ericsson-mobility-report-june-2016.pdf>>. Acesso em: 17 Fevereiro 2019.
- FERNANDES, D. G.; PREUSS, E.; SILVA, T. L. D. **Sistema Automatizado de Controle de**

Estufas para Cultivo de Hortaliças, 2017.

FERREIRA, A. D. S. et al. **Influências Climáticas nas Culturas Agrícolas**. Capanema: [s.n.], 2014.

FILIPFLOP. Disponível em: <<https://www.filipeflop.com/produto/ethernet-shield-w5100-para-arduino/>>. Acesso em: 22 Junho 2019.

GEWEHR, D. **Tecnologias Digitais de Informação e Comunicação (Tdics) na Escola e em Ambientes Não Escolares (Dissertação de Mestrado)**. Lajeado: [s.n.], 2016.

GIRARDI, G. C. **Automação do Controle e do Monitoramento de Temperatura e Umidificação de Canteiros de Cogumelo Agaricus blazei no Cultivo Familiar em Santa Helena/PR (Dissertação de Mestrado)**. Medianeira: [s.n.], 2018.

GOMES, I. C. **Desenvolvimento de um Sensor Digital de Umidade do Solo e Unidade Remota de Monitoramento Utilizando Comunicação sem Fio**. Pato Branco: [s.n.], 2016.

IDC. Internet of Things Market to Reach \$1.7 Trillion by 2020, 2019. Disponível em: <<http://www.idc.com/infographics/IoT>>. Acesso em: 16 Fevereiro 2019.

IOT Analytics - ThingSpeak Internet of Things. Disponível em: <<https://thingspeak.com/>>.

I-SCOOP. The internet of things (IoT) - essential iot business guide, 2019. Disponível em: <<https://www.i-scoop.eu/internet-of-things-guide/>>. Acesso em: 14 Fevereiro 2019.

KENSKI, V. M. Educação e tecnologias: Um novo ritmo da informação.. **Papirus**, n. 8, p. 15-25, 2012.

KIM, J. Y. et al. **Smart Home Web of Objects-based IoT Management Model and Methods for Home data mining**. [S.l.]: IEICE, 2015.

KLOPFER, E.; SQUIRE, K.; JENKINS, H. **Environmental Detectives: PDAs as a window into a virtual simulated world**. [S.l.]: [s.n.], 2002.

MARINO, D. R. D. M.; VASCONCELOS, D. R.; MORAES, S. G. Jardim Inteligente IoT - IIOT. **Revista Tecnologia Fortaleza**, v. 38, p. 34-59, 2017.

MAUREIRA, M. G.; OLDENHOF, D.; TEERNSTRA, L. ThingSpeak – an API and Web Service for the Internet of Things., 2014. Disponível em: <https://staas.home.xs4all.nl/t/swtr/documents/wt2014_thingspeak.pdf>. Acesso em: 5 Maio 2019.

PATHAK, P. B. Internet of things: a look at paradigm shifting applications and challenges. **International Journal of Advanced Research in Computer Science**, Abril 2016. 49–51.

PAULA, S. E. A. **Estudo do Ganho de Energia Elétrica em Painéis Fotovoltaicos usando Rastreamento Solar Baseado em Sistemas Embarcados**. São Paulo: [s.n.], 2015.

PEÑARANDA, R. E. C. **Sistema de Irrigação Automatizado pela Umidade do Solo**. Curitiba: Universidade Tecnológica Federal do Paraná, 2018.

PEREIRA, F. A. **Explosão dos dados e o conceito de análise de dados relacionados para a geração de informações (Big Data)**. [S.l.]: [s.n.], 2013.

PETRONILHO, L. B. **Aplicativo para auxílio no cultivo de hortas residenciais urbanas**. São Paulo: [s.n.], 2015.

POVO, O. Consumo de smartphones aumenta 33% no primeiro trimestre deste ano, 9 Junho 2015. Disponível em: <<https://www20.opovo.com.br/app/maisnoticias/brasil/2015/06/09/noticiasbrasil,3450881/consumo-de-smartphones-aumenta-33-no-primeiro-trimestre-deste-ano.shtml>>. Acesso em: 12 Fevereiro 2019.

PROSTT, M. E. **Interface Web Utilizando Design Responsivo: Um Estudo De Caso Aplicado A Smartphones, Tablets, Computadores E Televisores**. Curitiba: [s.n.], 2013.

RODRIGUES, K. D. O.; KALIL, F. **Tecnologia e o futuro: internet das coisas, microcontroladores e webservices**. [S.l.]: Núcleo de Estudo e Pesquisa em Computação Aplicada - NEPCA, IMED, 2010.

ROSENMANN, C. H. B. A. et al. **Design e internet das coisas em produtos de linha branca no Brasil**. [S.l.]: 11º Congresso Brasileiro de Inovação e Gestão.

SANTOS, B. et al. **Internet das Coisas: da Teoria à Prática**. Belo Horizonte: [s.n.], 2017.

SANTOS, L. B. **Sistema Automatizado para Controle De Umidade e Temperatura em Cultura de Morangos Aplicados aos Pequenos Produtores (Trabalho de Conclusão de Curso)**. Brasília: [s.n.], 2014.

SEBRAE. **Horas Urbanas - Moradia Urbana com Tecnologia Social**. [S.l.]: [s.n.], 2017.

SILVA, U. L. **Uma Revisão Sistemática da Literatura sobre Desenvolvimento de Aplicativos para Dispositivos Móveis: Tendências e Desafios**. Campina Grande: [s.n.], 2014.

SOMMERVILLE, I. **Engenharia de Software, 9 ed.** [S.l.]: Pearson, 2011.

SOUZA, K. B.; RIBEIRO, K. C.; OCCHI, L. C. M. **O Atual Cenário do Consumo de Alimentos Orgânicos**. Sergipe: [s.n.], 2017.

STATCOUNTER. **Mobile & Tablet Operating System Market Share Worldwide, 2019**. Disponível em: <<http://gs.statcounter.com/os-market-share>>. Acesso em: 10 Março 2019.

SUNROM. **Light Dependant Resistor, 2019**. Disponível em: <<https://www.sunrom.com/get/443700>>. Acesso em: 22 Junho 2019.

THINGSPEAK. **IoT Analytics - ThingSpeak Internet of Things, 2019**. Disponível em:

<<https://thingspeak.com/>>.

VILAÇA, M. L. C.; ARAUJO, E. V. F. **Tecnologia, Sociedade e Educação na Era Digital**.
Duque de Caxias: Universidade Unigranrio, 2016.

ANEXO A

Código fonte da estação local - BeThere_LocalStation.ino

```
#include "ThingSpeak.h"
#include <SPI.h>
#include <Ethernet.h>
#include "DHT.h" //DHT Library
#include <Wire.h>

byte mac[] = {0xDE, 0xAD, 0xBE, 0xEF, 0xFE, 0xED};
EthernetClient client; //ethernet client object
// Inicializa o display no endereço 0x27

//Atribuição dos pinos
#define sensor A0
#define luz A1
#define ledVermelho 9
#define ledAmarelo 8
#define ledVerde 7
#define ledAzul 6
#define bomba 5

//Configuração do DHT
#define pinDHT 4 //DHT communication pin
#define typeDHT DHT22 //DHT sensor type, in this case DHT22

DHT dht(pinDHT, typeDHT); //declaring an DHT object

//variáveis para gravação
int umidade;
int luminosidade;
int luminosidade2;
String nivelUmidade;
String nivelLuz;
int codSoil;
long stopRead = 0;
```

```
long startPump = 0;
unsigned long startTimer = 0UL;

//flags
//int bombaFlag = 0;
int luzFlag;
long int updateTime = 0;
long int recordedTime = 0;
long timerFlag = 0; //flag p/ indicar se o timer está ligado ou desligado
int timeSet = 0; //armazena o tempo do timer lido no ThingSpeak
int tempSet; //valor de temperatura configurado pelo usuario como maximo
int addTime = 0; //tempo de acrescimo da bomba caso a temperatura seja maior que maxTemp e
umidade baixa
unsigned long totalTempo = 0UL;
int pumpTime = 0;
int deltaTemp = 0;

//Campos do ThingSpeak
unsigned long mainChannelNumber = 695672;
const char * myWriteAPIKey = "ZY113X3ZSZG96YC8";

void setup() {

    Serial.begin(9600);
    dht.begin(); //initialize DHT object
    Ethernet.begin(mac);
    ThingSpeak.begin(client);

    //Entradas e saídas

    //portas analógicas e relé
    pinMode(sensor, INPUT); //sensor de umidade
    pinMode(luz, INPUT); //porta conectada ao divisor de tensão com fotoresistor)
    pinMode(bomba, OUTPUT); //acionamento do relé
    //Leds indicadores
    pinMode(ledVerde, OUTPUT);
    pinMode(ledAmarelo, OUTPUT);
```

```

pinMode(ledVermelho, OUTPUT);
pinMode(ledAzul, OUTPUT); //para indicar bomba ligada

//Estados iniciais
digitalWrite(bomba, HIGH); //bomba desligada
//digitalWrite(resetPin, LOW);

Serial.print("----- BE THERE - ONLINE -----\\n");
}

void loop() {

if(timerFlag == 1){

if(totalTempo > recordedTime){
    startPump = millis();
    Serial.print("\\nTimer mode: pump on");

    pumpTime = 20000 + addTime; //tempo padrao da bomba ligada + addTime

    while(millis()<startPump+pumpTime){
        digitalWrite(bomba, LOW);
        digitalWrite(ledAzul, HIGH);
    }
    digitalWrite(bomba, HIGH);
    digitalWrite(ledAzul, LOW);
    Serial.print("\\nYour garden has been watered!\\n");
    startTimer = millis();
    addTime = 0;
}
}

if(millis()-stopRead > 35000){

totalTempo = millis()-startTimer; //calcula quantidade de tempo desde o início do timer

timeSet = ThingSpeak.readFloatField(mainChannelNumber, 6); //ler timeSet: indica se o timer foi

```


ligado

```
//recebe o timeSet do ThingSpeak e verifica se o timer está ligado.
//nesta seção, no funcionamento ideal seria necessário programar as rotinas para 6h, 8h, etc.
//nesta versão o timer ligado seta o tempo do timer para acionar a bomba a cada 2 minutos.
switch (timeSet) {
case 0:
    updateTime = 0; //timer desligado
    break;
case 1:
    updateTime = 120000; //ligar por 2 min
    break;
}

if(updateTime != recordedTime){ //verifica se ocorreu mudança no status do Timer

    recordedTime = updateTime; //grava o novo time setado

    if(recordedTime != 0){
        timerFlag = 1; //timer ligado
        startTimer = millis(); //timer começa contar
        Serial.print("\n:");
        Serial.println(recordedTime);
    } else{
        timerFlag = 0; //timer desligado
    }
} //se não ocorre mudança, o tempo gravado é mantido


//leitura da umidade
umidade = analogRead(sensor);
//leitura da ponta analógica do divisor de tensão com foto resistor
luminosidade = analogRead(luz);


//check do status da bomba no ThingSpeak
//bombaFlag = ThingSpeak.readFloatField(mainChannelNumber, 5);
```

```

//conferencia da luminosidade - atribuição de status
if(luminosidade >= 891){
    luzFlag = 0; //sem luz
    digitalWrite(ledAmarelo, LOW);
}
else {
    luzFlag = 1; //com luz
    digitalWrite(ledAmarelo, HIGH);
}

//read humidity and temperature
float h = dht.readHumidity();
float t = dht.readTemperature();

//sensor DHT read failure check
if (isnan(h) || isnan(t)) {
    Serial.print("\n");
    Serial.print("\nDHT failed! Check Connections!");
} else {
    if(timerFlag == 1){ //verifica modo timer
        tempSet = ThingSpeak.readFloatField(mainChannelNumber, 5); //ler temperatura maxima
        configurada pelo usuário
    }
    if(t > tempSet && h < 30.00){ //verifica se a temperatura lida é maior que a temperatura
        configurada pelo usuário via web

        deltaTemp = t-tempSet;
        if(deltaTemp > 5 && deltaTemp < 10){
            addTime = 15000; //vai somar 15 s no tempo padrao da bomba
        } else if (deltaTemp > 10 && deltaTemp < 20){
            addTime = 25000; //vai somar 25 s no tempo padrao da bomba
        } else if(deltaTemp > 20){
            addTime = 30000; //vai somar 30s no tempo padrao da bomba
        }
    }
}
}

```

```
//classificação do solo
if (umidade <= 500){

    codSoil = 1; //solo úmido

    //Acende led verde
    digitalWrite(ledVerde, HIGH);
    digitalWrite(ledVermelho, LOW);
    digitalWrite(ledAzul, LOW);

    if (timerFlag == 0){
        digitalWrite(bomba, HIGH); //desliga bomba
    }
}

else if (umidade > 500 && umidade < 750){

    codSoil = 2; //solo parcialmente umido

    //Acende led amarelo
    digitalWrite(ledVerde, LOW);
    digitalWrite(ledVermelho, LOW);
    digitalWrite(ledAzul, LOW);

    if (timerFlag == 0){
        digitalWrite(bomba, HIGH); //desliga bomba
    }
}

else if(umidade > 750)
{
    codSoil = 3; //solo seco

    //Acende led vermelho
    digitalWrite(ledVerde, LOW);
    digitalWrite(ledVermelho, HIGH);
```

```

if (timerFlag == 0 && luzFlag == 1){
    digitalWrite(bomba, LOW); //liga bomba
    digitalWrite(ledAzul, HIGH);
}
}

if(codSoil == 3 && luzFlag == 0 && timerFlag == 0){
    digitalWrite(bomba, HIGH); //desliga bomba
    digitalWrite(ledAzul, LOW); //apaga led bomba
}

//ajustar limites para exibir o valor de 0 - 100 (seco-umido/sem luz-com luz)
umidade = map(umidade, 0, 1023, 100, 0);

//imprimir na serial as leituras
Serial.print("\n----- BE THERE - REPORT -----");

if(timerFlag == 0){
    Serial.print("\nTimer Mode Off!");
    Serial.print("\nUmidity Mode On!");
} else {
    Serial.print("\nTimer Mode On!");
    Serial.print("\nUmidity Mode Off!");
}

Serial.print("\n----- Soil Status ----- ");
Serial.print("\nSoil Moisture: ");
Serial.print(umidade);
Serial.print("%");

//status do solo
if (codSoil == 1){
    nivelUmidade = "moist";
} else if (codSoil == 2){
    nivelUmidade = "partially Moist";
} else{
    nivelUmidade = "dry";
}

```

```
}
```

```
if (luzFlag == 1){  
    nivelLuz = "incident light";  
} else {  
    nivelLuz = "no light";  
}
```

```
luminosidade2 = map(luminosidade, 1023, 0, 0, 100);
```

```
Serial.print("\nLuminosity: ");  
Serial.print(luminosidade2);  
Serial.println("%");  
Serial.println("Your soil is " + nivelUmidade + " and your garden has " + nivelLuz);
```

```
//DHT posts
```

```
Serial.print("\n----- Weather Status -----");  
Serial.print("\nHumidity: ");  
Serial.print(h);  
Serial.print("%\n");  
Serial.print("Temperature: ");  
Serial.print(t);  
Serial.print("°C\n");  
Serial.print("\n");
```

```
//set fields
```

```
ThingSpeak.setField(1, umidade);  
ThingSpeak.setField(2, luminosidade);  
ThingSpeak.setField(3, h);  
ThingSpeak.setField(4, t);
```

```
//write fields
```

```
int x = ThingSpeak.writeFields(mainChannelNumber,myWriteAPIKey);
```

```
if (x == 200){  
    Serial.print("Data sent with success!");  
} else{
```

```
    Serial.print("Coneection Error: " + String(x));  
    Serial.print("\n");  
}  
stopRead = millis();  
}  
}
```

ANEXO B

Página web – index.html

```
<!DOCTYPE html>
```

```
<head>
```

```
  <title>BeThere</title>
```

```
  <meta charset="utf-8">
```

```
  <!--<link rel="icon" href="favicon.png">-->
```

```
  <link rel="stylesheet" href="reset.css">
```

```
  <link rel="stylesheet" href="style.css">
```

```
  <link                                rel="stylesheet"
href="http://fonts.googleapis.com/css?family=Open+Sans+Condensed:700">
```

```
  <!--menu-->
```

```
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
```

```
  <meta name="viewport" content="width=device-width, initial-scale=1">
```

```
  <link rel="stylesheet" href="navigation-dark.css">
```

```
  <link rel="stylesheet" href="slicknav.min.css">
```

```
  <link                                rel="stylesheet"
                                href="http://maxcdn.bootstrapcdn.com/font-
awesome/4.2.0/css/font-awesome.min.css">
```

```
  <script src="http://ajax.googleapis.com/ajax/libs/jquery/2.0.0/jquery.min.js"></script>
```

```
</head>
```

```
<body>
```

```
  <main>
```

```
    <div id="dashboard">
```

```
      <h1>Dashboard</h1>
```

```
<div class="container">
```

```
  <div class="soil">
```

```
    <p>Soil Moisture </p>
```

```
    <iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/695672/widgets/39942"></iframe>
```

```
  </div>
```

```
  <div class="luminosity">
```

```
    <p>Luminosity</p>
```

```
    <iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/695672/widgets/63648"></iframe>
```

```
  </div>
```

```
  <aside class="weather-box">
```

```
    <a                                     class="weatherwidget-io"
href="https://forecast7.com/en/n22d01n47d89/sao-carlos/" data-label_1="SÃO CARLOS"
data-label_2="WEATHER" data-theme="gray" >SÃO CARLOS WEATHER</a>
```

```
    <script>
```

```
      !function(d,s,id){var
js,fjs=d.getElementsByTagName(s)[0];if(!d.getElementById(id)){js=d.createElement(s);js.id
=id;js.src='https://weatherwidget.io/js/widget.min.js';fjs.parentNode.insertBefore(js,fjs);}}(do
cument,'script','weatherwidget-io-js');
```

```
    </script>
```

```
  </aside>
```

```
</div>
```

```
</div>
```

```
<div id="charts">
```


<h1>Charts</h1>

<div class="graph1-moisture">

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/695672/charts/1?bgcolor=%23ffffff&color=%23d620
20&dynamic=true&results=60&title=Soil+Moisture&type=line"></iframe>

</div>

<div class="graph2-temperature">

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/695672/charts/4?bgcolor=%23ffffff&color=%23d620
20&dynamic=true&results=60&title=Temperature&type=line"></iframe>

</div>

<div class="graph3-umidity">

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/695672/charts/3?bgcolor=%23ffffff&color=%23d620
20&dynamic=true&results=60&title=Relative+Humidity&type=line"></iframe>

</div>

<div class="graph4-luminosity">

<iframe width="450" height="260" style="border: 1px solid #cccccc;"
src="https://thingspeak.com/channels/695672/charts/2?bgcolor=%23ffffff&color=%23d620
20&dynamic=true&results=60&title=Luminosidade&type=line"></iframe>

</div>

</div>

<div id="books">

<h1>Library</h1>

<div class="container">

<p>Full sun. 6+ hours of direct sunlight.

Partial sun. 4-5 hours of direct sunlight.

Partial shade. 2-4 hours of direct sunlight.

Shade Less than 1 hour of direct sunlight.</p>

</div>

</div>

<div id="settings">

<h1>Settings</h1>

<div class="container modes">

<h2>Operation Mode</h2>

<button type="button" id="timer-on" class="button button3" >Timer</button>

<button type="button" id="umidity-on" class="button button4" >Umidity</button>

</div>

<div class="timers" id="TimerOptions">

<h2>Set timer</h2>

<button type="button" id="6hours" class="button button5" >6h</button>

<button type="button" id="8hours" class="button button6" >8h</button>

<button type="button" id="12hours" class="button button7" >12h</button>

<button type="button" id="16hours" class="button button8" >16h</button>

<button type="button" id="24hours" class="button button9" >24h</button>

<div>

<h3>Temperature: °C

</h3><input type="text" id="tempValue" placeholder="insert the temperature"/input>

<button type="button" id="send" class="button">SEND</button>

</div>

</div>

<div class="container modes" id="UmidityOptions">

<h2>UMIDITY MODE ON!</h2>

```
<p>Running in umidity mode, the timer mode is offline. Click in "Timer"
to switch off umidity mode and use timer. </p>
</div>
</div>
```

```
</main>
```

```
<div class="logo-imagem">
  
</div>
```

```
<!--<footer class="rodape-pagina">
  <!--&copy; Be There-->
```

```
<!--Menu Inferior-->
```

```
<!--&copy; Be There - 2019
</footer>-->
```

```
<nav class="menu-navigation-dark">
```

```
  <a href="#dashboard" class="selected"
onclick="mostrar('dashboard');ocultar('charts');ocultar('books');ocultar('settings');"><i
class="fa fa-th-list"></i><span>Dashboard</span></a>
  <a href="#charts"
onclick="mostrar('charts');ocultar('dashboard');ocultar('books');ocultar('settings');"><i
class="fa fa-bar-chart"></i><span>Charts</span></a>
  <a href="#books"
onclick="mostrar('books');ocultar('charts');ocultar('dashboard');ocultar('settings');"><i
class="fa fa-book"></i><span>Library</span></a>
  <a href="#settings"
onclick="mostrar('settings');ocultar('dashboard');ocultar('books');ocultar('charts');"><i
```

```
class="fa fa-gear"></i><span>Settings</span></a>
```

```
</nav>
```

```
<!--onclick="funcao_a();funcao_b();"-->
```

```
<script src="http://ajax.googleapis.com/ajax/libs/jquery/2.1.1/jquery.min.js"></script>
```

```
<script src="jquery.slicknav.min.js"></script>
```

```
<script src="https://ajax.googleapis.com/ajax/libs/jquery/3.4.1/jquery.min.js"></script>
```

```
<script>
```

```
var umidade = 1;
```

```
var timer = 0;
```

```
$("#timer-on").click(function(){  
    $("#UmidityOptions").hide();  
    $("#TimerOptions").show();  
});
```

```
$("#umidity-on").click(function(){  
    $("#UmidityOptions").show();  
    $("#TimerOptions").hide();  
});
```

```
$(function(){
```

```
    var menu = $('.menu-navigation-dark');
```

```
    menu.slicknav();
```

```
    // Mark the clicked item as selected
```

```

menu.on('click', 'a', function(){
    var a = $(this);

    a.siblings().removeClass('selected');
    a.addClass('selected');
});
});

```

```

function mostrar(ID){
    document.getElementById(ID).style.display = "block";

}

```

```

function ocultar(ID){
    document.getElementById(ID).style.display = "none";
    $("#dive").hide("slow");
}

```

```

function alterMode(){
}

```

```

document.getElementById('umidity-on').addEventListener('click', function() {
    var url =
"http://api.thingspeak.com/update?api_key=ZY113X3ZSZG96YC8&field6=0"
    alert("Umidity Mode ON");
    $.getJSON(url, function(data) {
        console.log(data);
    });
});

```

```

document.getElementById('timer-on').addEventListener('click', function() {
    var url =
"http://api.thingspeak.com/update?api_key=ZY113X3ZSZG96YC8&field6=1"

```

```

    alert("Timer Mode ON");
    $.getJSON(url, function(data) {
        console.log(data);
    });
});

```

// Função para envio de uma nova temperatura

```

document.getElementById('send').addEventListener('click', function() {
    var temperatureSend = document.getElementById('tempValue').value ;
    var url =
"http://api.thingspeak.com/update?api_key=ZY113X3ZSZG96YC8&field5="+temperatureS
end
    alert("Temperature Send: "+temperatureSend);
    $.getJSON(url, function(data) {
        location.reload("#settings");
        console.log(data);
    });
});

```

// Função para leitura da variável de temperatura na API (última temperatura enviada)

```

$.getJSON('https://api.thingspeak.com/channels/695672/field/5/last.json?apikey=QVBH04I
290VRXEOL', function(data) {
    dataField5 = data.field5;
    if (dataField5) {
        // $('#tempValueSpan').val(dataField5);
        document.getElementById('tempValueSpan').innerHTML = dataField5;
    }
});

```

// Leitura da API para receber última "posição" do tipo de modo ativo

```

$.getJSON('https://api.thingspeak.com/channels/695672/field/6/last.json?apikey=QVBH04I
290VRXEOL', function(data) {

```

```
dataField6 = data.field6;
if (dataField6 == 1) {
    timer = 1; // Timer ativa
    umidade = 0; // Umidade desativada
    console.log("Timer Ativo - Umidade Desativada");
    $("#TimerOptions").show();
    $("#UmidityOptions").hide();
} else {
    timer = 0; // Timer desativado
    umidade = 1; // Umidade ativa
    console.log("Umidade Ativada - Timer Desativado");
    $("#UmidityOptions").show();
    $("#TimerOptions").hide();
}
});
```

</script>

</body>

ANEXO C

Página Web – style.css

```
body{
    min-height:100%;
    background-color: ghostwhite;
    color: black;
    font-size: 20px;
}
```

```
img{
    position: absolute;
    top: 0;
    left: 5px;
    background-color: #339999;
    margin: 0;
}
```

```
aside{
    width: 262px;
    margin-bottom: 50px;
    text-align: center;
    border-right: 2px solid black;
    border-bottom: 2px solid black;
    padding: 40px;
}
```

```
.barra-principal a{
    font-family: "Century Gothic";
    font-size: 20px;
    color: #F2FFFC;
```



```
}
```

```
.container{  
  min-height:100%;  
  width: 500px; /*largura do conteúdo*/  
  /*centralizaçao*/  
  margin-top: 10px;  
  margin-left: 200px;  
  margin-right: auto;  
  text-align: left;  
  font-family: "Leelawadee UI" ,sans-serif;  
  line-height: 3.0;  
}
```

```
.button{  
  text-align: center;  
}
```

```
.section-main{  
  
  width: 1000px;  
}
```

```
.timers{  
  min-height:100%;  
  width: 500px; /*largura do conteúdo*/  
  /*centralizaçao*/  
  position: absolute;  
  top: 145px;  
  margin-left: 500px;  
  margin-right: auto;  
  text-align: left;  
  font-family: "Leelawadee UI" ,sans-serif;  
  line-height: 3.0;
```

```
}
```

```
#charts{  
    display: none;  
}
```

```
#settings{  
    display: none;  
}
```

```
#books{  
    display: none;  
}
```

```
.graph1-moisture{  
    padding: 7px;  
    position: absolute;  
    right: 150px;  
}
```

```
.graph2-temperature{  
    padding: 7px;  
    position: absolute;  
    right: 600px;  
}
```

```
.graph3-umidity{  
    padding: 7px;  
    position: absolute;  
    right: 150px;  
    top: 380px;  
}
```

```
.graph4-luminosity{
  padding: 7px;
  position: absolute;
  right: 600px;
  top: 380px;
}
```

```
h1 {
  font-family: "Century Gothic" , sans-serif;
  font-size: 50px;
  margin-left: auto;
  margin-right: auto;
  /*passar o mouse em cima da cor exibe a preview */
  color: white; /*cor do texto*/
  background-color: #339999;
  text-align: center;
  padding: 40px;
  border-bottom: 2px solid black;
}
```

```
h2{
  font-size: 25px;
  padding: 5px;
  font-family: "Century Gothic" , sans-serif;
  color: black;
}
```

```
footer{
  font-family: "Verdana" , sans-serif;
  background-color: #000;
  font-size: 12px;
  position: absolute;
```

```
bottom: 0;
clear: both;
color: #F2FFFC;
width: 100%;
```

```
}
```

```
strong{
    font-weight: bold;
}
```

```
em{
    font-style: italic;
}
```

```
.menu-navigation-dark {
    font-family: "Century Gothic", sans-serif;
    font-size: 0;
    text-align: center;
    position: fixed;
    top: 170px;
    left: 0;
}
```

```
.weather-box{
    position: absolute;
    right: 0;
    bottom: 0;
    width: 344px;
    border: 1px;
}
```

```
.sensors{
```

```

    margin-top: 10px;
}
.menu-navigation-dark a {
    display: block;
    color: #ffffff;
    background-color: #232526;
    font-size: 35px;
    font-weight: bold;
    box-shadow: 1px 3px 5px 0 rgba(0, 0, 0, 0.26);
    text-transform: uppercase;
    text-decoration: none;
    white-space: nowrap;
    border: 1px solid #161718;
    border-top: none;
    width: 136px;
    margin: 0 auto;
    padding: 20px 0;
    box-sizing: border-box;
}

.menu-navigation-dark a:hover {
    background-color: #27292a;
}

.menu-navigation-dark a:first-child{
    border-left: 1px solid #161718;
}

.menu-navigation-dark a:not(.selected) {
    box-shadow:    1px 3px 5px 0 rgba(0, 0, 0, 0.26),
    inset 1px 0 0 #323435,
    inset 0 1px 0 #282a2b,
    inset 0 -1px 0 #282a2b;
}

```

```
.menu-navigation-dark a:last-child {  
    box-shadow:    1px 3px 5px 0 rgba(0, 0, 0, 0.26),  
    inset 1px 0 0 #323435,  
    inset -1px 0 0 #282a2b,  
    inset 0 1px 0 #282a2b,  
    inset 0 -1px 0 #282a2b;  
}
```

```
.menu-navigation-dark a i {  
    display: block;  
    line-height: 1.3;  
}
```

```
.menu-navigation-dark a span {  
    display: block;  
    font-size: 11px;  
    font-weight: bold;  
    line-height: 2;  
    text-shadow: 0 1px 1px rgba(0, 0, 0, 0.1);  
}
```

```
.menu-navigation-dark a.selected {  
    background-color: #1d1e1f;  
    border-bottom: 2px solid #99cc66;  
    pointer-events: none;  
}
```

```
.menu-navigation-dark a.selected i {  
    color: #99cc66;  
}
```

```
/* Make this page responsive */
```

```
.slicknav_menu {  
    display:none;  
}
```

```
@media (max-width: 800px) {  
    .menu-navigation-dark{  
        display:none;  
    }  
}
```

```
.slicknav_nav a i {  
    display: none;  
}
```

```
.slicknav_menu {  
    display:block;  
}  
}
```

/* The switch - the box around the slider */

```
.switch {  
    position: relative;  
    display: inline-block;  
    width: 60px;  
    height: 34px;  
}
```

/* Hide default HTML checkbox */

```
.switch input {  
    opacity: 0;  
    width: 0;  
    height: 0;  
}
```

```
/* The slider */
```

```
.slider {  
  position: absolute;  
  cursor: pointer;  
  top: 0;  
  left: 0;  
  right: 0;  
  bottom: 0;  
  background-color: #ccc;  
  -webkit-transition: .4s;  
  transition: .4s;  
}
```

```
.slider:before {  
  position: absolute;  
  content: "";  
  height: 26px;  
  width: 26px;  
  left: 4px;  
  bottom: 4px;  
  background-color: white;  
  -webkit-transition: .4s;  
  transition: .4s;  
}
```

```
input:checked + .slider {  
  background-color: #2196F3;  
}
```

```
input:focus + .slider {  
  box-shadow: 0 0 1px #2196F3;  
}
```

```
input:checked + .slider:before {
```



```
-webkit-transform: translateX(26px);  
-ms-transform: translateX(26px);  
transform: translateX(26px);  
}
```

```
/* Rounded sliders */  
.slider.round {  
  border-radius: 34px;  
}
```

```
.slider.round:before {  
  border-radius: 50%;  
}
```