**CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA**

**Faculdade de Tecnologia Rubens Lara**

**Curso Superior de Tecnologia em Análise e Desenvolvimento de Sistemas**

**LUIS GUILHERME DA COSTA MARTINS**

**PEDRO PAULO MIRANDA SOUZA**

**SISTEMA DE GERENCIAMENTO DE ENERGIA ELÉTRICA UTILIZANDO ARDUINO**

**Santos, SP**

**2017**

**LUIS GUILHERME DA COSTA MARTINS**

**PEDRO PAULO MIRANDA SOUZA**

**SISTEMA DE GERENCIAMENTO DE ENERGIA ELÉTRICA UTILIZANDO ARDUINO**

Trabalho de Conclusão de Curso apresentada à Faculdade de Tecnologia Rubens Lara, como exigência para a obtenção do Título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

**Orientador: Rui Silverstrin**

**Santos - SP**

**2017**

**LUIS GUILHERME DA COSTA MARTINS**

**PEDRO PAULO MIRANDA SOUZA**

**SISTEMA DE GERENCIAMENTO DE ENERGIA ELÉTRICA UTILIZANDO ARDUINO**

Trabalho de Conclusão de Curso apresentada à Faculdade de Tecnologia da Baixada Santista, como parte dos requisitos para a obtenção do Título de Tecnólogo em Análise e Desenvolvimento de Sistemas.

Aprovado em \_\_\_\_ de \_\_\_\_\_\_\_\_\_\_\_\_ de 20\_\_

BANCA EXAMINADORA:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Nome do examinador:

Titulação:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Nome do examinador:

Titulação:

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Nome do examinador:

Titulação:

**Local:** Faculdade de Tecnologia da Baixada Santista

**DEDICATÓRIA**

Aos meus pais, pelo amor, incentivo, apoio incondicional e qυе apesar dе todas аs dificuldades mе fortaleceu е qυе pаrа mіm foi muito importante.

**AGRADECIMENTOS**

Ao Prof. Rui Silvestrin pela orientação e, mesmo tendo vários outros TCC’s para orientar, teve grande desprendimento em ajudar-nos e aos meus amigos da Fatec pelo incentivo e grande ajuda com o fornecimento de material para a realização deste trabalho.

**EPÍGRAFE**

“A melhor maneira de prever o futuro, é criá-lo”

(Peter Drucker)

**RESUMO**

Este trabalho trata-se do desenvolvimento de um sistema de gerenciamento de energia elétrica utilizando Arduino. O tema abordado é devido ao uso ineficiente de energia elétrica no Brasil, e as tarifas altas presentes no país. O objetivo deste trabalho é desenvolver um software alternativo para o monitoramento de consumo da rede elétrica, utilizando Arduino, capaz de mostrar ao consumidor uma noção dos gastos estimados da energia em sua residência, identificando os possíveis pontos críticos, permitindo-o analisar e a mudar suas rotinas a fim de minimizar os seus gastos. Neste trabalho é abordado conceitos básicos da energia elétrica, o setor de energia elétrica no Brasil, as ferramentas necessárias e o processo de desenvolvimento do projeto.

**Palavras-Chave**: Energia Elétrica; Gasto; Arduino.

**ABSTRACT**

This work deals with the development of an electrical power management system using Arduino. The issue addressed is due to the inefficient use of electric power in Brazil, and the high tariffs present in the country. The objective of this work is to develop an alternative software for the monitoring of power consumption using Arduino, able to show the consumer a notion of the estimated energy expenditures in his residence, identifying the possible critical points, allowing you to analyze and change your routines in order to minimize your expenses. In this work is addressed the basic concepts of electric energy, the electric energy sector in Brazil, the necessary tools and the project development process.

**Keywords**: Electric Power; Spent; Arduino.

**LISTA DE ILUSTRAÇÕES**

[Figura 1: Valor energia elétrica 21](#_Toc483840903)

[Figura 2: Sensor não-invasivo (100A SCT-013) 22](#_Toc483840904)

[Figura 3: Ethernet Shield W5100 23](#_Toc483840905)

[Figura 4: Arduino Mega 25](#_Toc483840906)

[Figura 5: Codenvy 30](#_Toc483840907)

[Figura 6: IDE Arduino 30](#_Toc483840908)

[Figura 7: Fritzing versão 0.9.3 33](#_Toc483840909)

[Figura 8: ArgoUML versão 0.34 35](#_Toc483840910)

[Figura 9: brModelo versão 3.0 36](#_Toc483840911)

[Figura 10: Caso de uso Usuário 37](#_Toc483840912)

[Figura 11: Caso de uso Proprietário 38](#_Toc483840913)

[Figura 12: Caso de uso Autorizado 38](#_Toc483840914)

[Figura 13: Modelo DER 43](#_Toc483840915)

[Figura 14: Modelo lógico 44](#_Toc483840916)

[Figura 15: Circuito eletrônico 47](#_Toc483840917)

[Figura 16: Regulagem do sensor 48](#_Toc483840918)

[Figura 18: Tela Principal 49](#_Toc483840919)

[Figura 19: Tela Consumo 50](#_Toc483840920)

[Figura 20: Tela Ambientes 50](#_Toc483840921)

[Figura 21: Tela Visualização do Ambiente 51](#_Toc483840922)

[Figura 22: Tela Usuários 52](#_Toc483840923)

[Figura 23: Tela Relatório 53](#_Toc483840924)

**LISTA DE TABELAS**

[Tabela 1: Regras de Negócio 37](#_Toc483839592)

[Tabela 2: Requisitos funcionais 38](#_Toc483839593)

**LISTA DE ABREVIATURAS E SIGLAS**

A Ampere

ACEEE American council for an energy efficient economy

ABESCO Associação brasileira das empresas de serviços de conservação de energia

AREF Analogue reference

CCEE Câmara de comercialização de energia elétrica

COFINS Contribuição para o financiamento da seguridade social

CPP Corrente de pico primária

CPS Corrente de pico secundário

CSS Cascading style sheets

DER Diagrama entidade relacionamento

E Tensão

GHC Glasgow haskell compiler

HTML Hypertext markup language

I Corrente

ICMS Imposto sobre circulação de mercadorias e serviços

ICSP In-Circuit Serial Programming

IDE Integrated development environment

KWh Quilowatt-hora

MWh Megawatt-hora

p. Página

P Potência

PIS Programa de integração social

RCI Resistência de carga ideal

REST Representational state transfer

RF Requisito funcional

RMS Root Mean Square

RN Regras de negocio

SGBD Sistema gerenciador de banco de dados

SGBDOR Sistema gerenciador de banco de dados objeto relacional

SPI Serial peripheral interface bus

SQL Structured query language

TCP Transmission control protocol

UART Universal asynchronous receiver/transmitter

UDP User datagram protocol

UFSC Universidade federal de Santa Catarina

UML Unified Modeling language

USB Universal serial bus

w Watt

Wh Watt-hora

SUMÁRIO

[1 INTRODUÇÃO 9](#_Toc483828019)

[1.1 OBJETIVO 9](#_Toc483828020)

[1.2 Objetivos específicos 10](#_Toc483828021)

[1.3 Organização do trabalho 10](#_Toc483828022)

[2 REVISÃO BIBLIOGRÁFICA 11](#_Toc483828023)

[2.1 Energia elétrica 11](#_Toc483828024)

[2.1.1 Tensão Elétrica 11](#_Toc483828025)

[2.1.2 Corrente Elétrica 11](#_Toc483828026)

[2.1.3 Potência Elétrica 12](#_Toc483828027)

[2.1.4 Cálculo de energia elétrica 13](#_Toc483828028)

[2.1.5 Energia elétrica brasileira 14](#_Toc483828029)

[2.2 Hardware 17](#_Toc483828030)

[2.2.1 Sensor não-invasivo (100A SCT-013) 17](#_Toc483828031)

[2.2.2 Ethernet shield W5100 18](#_Toc483828032)

[2.2.3 Arduino Mega 19](#_Toc483828033)

[2.3 Software 20](#_Toc483828034)

[2.3.1 Haskell 21](#_Toc483828035)

[2.3.2 GHCI 21](#_Toc483828036)

[2.3.3 Modelo REST 22](#_Toc483828037)

[2.3.4 Yesod 22](#_Toc483828038)

[2.3.5 HTML 23](#_Toc483828039)

[2.3.6 CSS 23](#_Toc483828040)

[2.3.7 JavaScript 24](#_Toc483828041)

[2.3.8 Bootstrap 24](#_Toc483828042)

[2.3.9 Code Envy 24](#_Toc483828043)

[2.3.10 Arduino IDE 25](#_Toc483828044)

[2.3.11 OpenEnergyMonitor 26](#_Toc483828045)

[2.3.12 PostgreSQL 27](#_Toc483828046)

[2.3.13 Heroku Postgres 27](#_Toc483828047)

[2.3.14 Fritzing 28](#_Toc483828048)

[2.3.15 UML 29](#_Toc483828049)

[2.3.16 ArgoUML 29](#_Toc483828050)

[2.3.17 BrModelo 30](#_Toc483828051)

[3 METODOLOGIA 31](#_Toc483828052)

[3.1 Engenharia de Software 31](#_Toc483828053)

[3.1.1 Diagrama de caso de uso 32](#_Toc483828054)

[3.1.2 Regras de negócio 34](#_Toc483828055)

[3.1.3 Requisitos funcionais 35](#_Toc483828056)

[3.2 Banco de dados 36](#_Toc483828057)

[3.2.1 Modelo DER 36](#_Toc483828058)

[3.2.2 Modelo lógico 38](#_Toc483828059)

[3.3 Circuito eletrônico 39](#_Toc483828060)

[3.4 Regulagem do sensor 100A SCT-013 42](#_Toc483828061)

[3.5 Calculo do consumo 43](#_Toc483828062)

[3.6 Interface do web-app 43](#_Toc483828063)

[4 CONSIDERAÇÕES FINAIS 48](#_Toc483828064)

[REFERÊNCIAS BIBLIOGRÁFICAS 51](#_Toc483828065)

# 1 INTRODUÇÃO

No Brasil, o custo de energia elétrica é alto, e existe um grande índice de consumo ineficiente. Segundo levantamento realizado pela ABESCO (Associação Brasileira das Empresas de Serviços de Conservação de Energia), o desperdício de energia no país chegou a mais de 10% em seis anos, com metade correspondendo aos consumidores residenciais, através dos eletrodomésticos. No âmbito internacional, entre as dezesseis maiores economias do mundo, o Brasil é o penúltimo em ranking de eficiência energética elaborado pela entidade internacional ACEEE (*American Council for an Energy-Efficient Economy*).

Observando as estatísticas e sabendo que as fontes renováveis e limpas ainda não possuem o mesmo grau de produtividade que as demais, a economia de energia, além de proporcionar a redução de gastos, também é um fator importante pela consciência sobre os danosos impactos ambientais resultantes do desperdício.

O projeto proposto neste trabalho consiste em um sistema de gerenciamento de energia elétrica utilizando Arduino, que é responsável pelo controle do sensor não invasivo (100A SCT-013), que realiza a captação da corrente elétrica passada pelo fio condutor ligado a um cômodo da residência. Com a captação da energia, o sistema envia o consumo para um web-server através do módulo Ethernet Shield, para que então, o usuário possa visualizar os gastos e o consumo estimado da sua casa através da internet, pelo web-app. O gasto será calculado a partir do preço da energia por kWh vigente.

Com o consumo sendo avaliado pelos residentes da casa, pode-se melhorar a eficiência do uso da energia elétrica, a fim de diminuir desperdícios e gastos. De modo que o gerenciamento possa ser feito com mais facilidade e acessibilidade pelos os usuários com um sistema simples e de baixo custo, mesmo que não seja totalmente preciso, proporcionando uma noção do consumo estimado da casa., resultando em mais controle sobre os gastos, e não apenas restrito às empresas que administram o serviço de energia elétrica no país.

## 

## 1.1 OBJETIVO

Desenvolver um sistema de gerenciamento de energia elétrica utilizando ferramentas de baixo custo, oferecendo uma alternativa para a consulta dos gastos para que se tenha uma noção sobre o consumo e economia na energia elétrica.

## 1.2 Objetivos específicos

* Coletar dados e informações sobre o funcionamento do sistema de energia elétrica no Brasil;
* Montar o sistema físico, com as ferramentas necessárias para a captação de energia, para a realização do cálculo com base nos dados e informações coletadas;
* Elaborar os requisitos necessários para a construção do sistema,
* Desenvolver as funcionalidades e a interface do sistema;
* Testar o sistema e observar os resultados de eficiência do projeto;

## 1.3 Organização do trabalho

Neste primeiro capitulo foi apresentado uma breve introdução sobre o que é o projeto, e quais são seus objetivos. Nos capítulos seguintes é abordado como foi realizado a execução destes objetivos.

O capitulo dois, Revisão Bibliográfica, é estruturado de forma para introduzir o leitor a compreensão dos conceitos, ferramentas, componentes e tecnologias utilizadas no projeto.

No capitulo três, Metodologia, é abordado o processo de desenvolvimento do sistema, com base nas informações e tecnologias apresentadas no capitulo anterior.

O capitulo quatro, Considerações finais, é apresentado os resultados do projeto.

# 2 REVISÃO BIBLIOGRÁFICA

Este capítulo descreve todas as tecnologias usadas, como funciona e como é aplicada, para resolver o problema apresentado no capítulo anterior. Assim como dizem Prodanov e Freitas, (2013, p. 131) serão analisadas as obras científicas que tratem do assunto e que deem embasamento teórico e metodológico para o desenvolvimento do projeto de pesquisa.

Nesta seção é descrito uma breve explicação de conceitos básicos sobre a energia elétrica e de como é calculado o consumo no Brasil. Também é introduzido, de forma simplificada, como funciona o setor do sistema de energia elétrica no país. E por fim, os componentes e as ferramentas necessárias para a construção do projeto, que foram divididas em duas categorias: hardware e software.

## 2.1 Energia elétrica

Nos tópicos seguintes, são apresentados conceitos básicos sobre a energia elétrica: tensão, corrente e potência. Também está descrito como é realizado o cálculo de consumo, e uma breve introdução sobre a energia elétrica e o seu funcionamento no Brasil.

### 2.1.1 Tensão Elétrica

Tensão elétrica é a força ou pressão elétrica, capaz de movimentar elétrons ordenamento num condutor. “Pode-se lembrar inclusive de uma analogia feita com um sistema hidráulico, onde observa-se que a água fluirá, através do cano, até que as “pressões” dos dois reservatórios se igualem”. (Comissão tripartite permanente de negociação do setor elétrico no estado de São Paulo, 2005, p. 14).

A tensão elétrica é representada por volts (V), conhecida por alguns como voltagem. No Brasil, o padrão de tensão é 230V e 110V.

### 

### 2.1.2 Corrente Elétrica

Corrente elétrica é o nome do movimento ordenado de cargas elétricas pelo condutor, é um fluxo de cargas elétricas. Por exemplo, de acordo com Cavalcante:

Alguns equipamentos elétricos, que chamamos de geradores de eletricidade, como as pilhas e as baterias, apresentam duas regiões que chamamos de polos. Um polo de maior potencial (+) e um polo de menor potencial (-), existindo, então, uma diferença de potencial.

Nos condutores de eletricidade, os elétrons da última camada, elétrons livres, estão fracamente ligados ao núcleo. Dessa forma, quando conectamos um fio condutor a uma bateria (gerador), os elétrons livres iniciam um movimento através do condutor, indo do polo de menor potencial (-) para o de maior potencial (+). (CAVALCANTE, 2016)

Para medir a corrente a unidade usada é o ampère (A).

De uma forma simplificada, podemos medir a corrente exigida por um aparelho utilizando a seguinte fórmula:

|  |  |  |
| --- | --- | --- |
|  |  | (1) |

Onde I é a corrente (em Ampères), P é a potência (em Watts) e E é a tensão (em Volts). Aplicando a fórmula em um aparelho cuja potência é de 2200 Watts, ligado à rede elétrica de 220V, tem-se uma corrente de 10A (amperes).

### 2.1.3 Potência Elétrica

De acordo com a comissão tripartite permanente de negociação do setor elétrico no estado de São Paulo (2005, p. 32) “Potência elétrica ou mecânica é a rapidez com que se faz o trabalho. Pode-se considerar, para facilitar o entendimento, como capacidade de produzir que trabalho que uma carga possui”.

A potência de uma carga depende de outras grandezas que são: R (resistência) e V (tensão aplicada). Uma vez aplicada uma tensão a resistência, ter-se a corrente I, ou seja, a potência também depende da corrente. Temos:

|  |  |  |
| --- | --- | --- |
|  |  | (2) |
|  |  | (3) |

Nós prenderemos mais à segunda equação P = V x I onde:

1. V: volts;
2. I: amperes.

A unidade de medida da potência elétrica é o WATT (w). Vale ressaltar que o valor da corrente em amperes (I), é o valor que o sensor capta do fio elétrico. E os volts (V), é a voltagem regulada pela fonte AC.

### 2.1.4 Cálculo de energia elétrica

Para calcular a energia elétrica, usa-se a equação:

|  |  |  |
| --- | --- | --- |
|  |  | (4) |

Onde:

1. E: energia elétrica;
2. P: potência;
3. ∆t: variação do tempo.

A energia elétrica pode ser medida em Watt-hora (Wh), quilowatt-hora (kWh) ou em megawatt-hora (MWh). No Brasil o valor padrão é o kWh (equivalente a mil watts), e o aparelho que mede a energia elétrica convencionalmente é o medidor de energia elétrica, que depende da potência elétrica da carga (P), explicado no tópico anterior, e da variação do tempo (∆t) em que a mesma foi ligada, exemplo: um computador foi mantido ligado por 10 horas seguidas.

Como mostrado na formula acima, o consumo é o resultado da potência multiplicada pela variação do tempo, que então é convertido para kWh dividindo o valor por mil (1000). Para calcular o custo é utilizado a formula:

|  |  |  |
| --- | --- | --- |
|  |  | **(5)** |

Onde:

1. C: custo em reais do consumo;
2. kWh: valor do consumo;
3. R$: valor em reais por kWh.

Sendo assim, o custo é o resultado do consumo (kWh) multiplicado pelo preço vigente da companhia que fornece o serviço, o processo de distribuição de energia e explicação das tarifas é abordado no tópico seguinte. Explica Cavalcante:

As companhias de energia utilizam o kWh para medir o consumo de energia elétrica. Se o consumo no mês de maio foi de 120 kWh e o custo de 1kWh é de R$ 0,48, a conta de energia referente a esse mês será de:

C = 120 x 0,48

C = R$ 57,60. (CAVALCANTE, 2016)

### 2.1.5 Energia elétrica brasileira

O sistema de distribuição de energia elétrica no Brasil é regulamentado pela Agência Nacional de Energia Elétrica (ANEEL), que é supervisionada por leis aprovadas no Congresso Nacional e nos decretos estabelecidos pelo Executivo Federal. A tarifa é composta por três custos principais:

1. Energia gerada;
2. Transporte de energia elétrica até as unidades consumidoras: dividido em transmissão e distribuição. Nesse setor não há competição, pois não há retorno financeiro. Por essa razão, fica a cargo da Aneel intervir sobre as tarifas dos serviços prestados;
3. Encargos setoriais: são encargos instituídos por lei, portanto não são criados pela ANEEL. Podendo variar entre o curso da distribuição, geração e transmissão.

Além desse valor, o Governo Federal, Estadual e Municipal cobram junto a tarifa o PIS/COFINS, o ICMS e a Contribuição para iluminação pública.

Com o novo modelo proposto em 2004, o valor da tarifa também está ligado às competições dos leilões públicos. Foi implantando, com o objetivo de alcançar a modicidade tarifária, e funcionam como instrumento de compra de energia elétrica pelas distribuidoras no ambiente regulado. Os leilões são realizados pela Câmara de Comercialização de Energia Elétrica (CCEE), por delegação da ANEEL, e utilizam o critério de menor tarifa, visando a redução de custo de aquisição da energia elétrica a ser repassada aos consumidores cativos. De acordo com CCEE:

O modelo em vigor exige a contratação de totalidade da demanda por parte das distribuidoras e dos consumidores livres; nova metodologia de cálculo do lastro para venda de geração; contratação de usinas hidrelétricas e termelétricas em proporções que assegurem melhor equilíbrio entre garantia e custo de suprimento, bem como o monitoramento permanente da segurança de suprimento. Este modelo foi implantado por meio das Leis nº 10.847 e 10.848, de 15 de março de 2004, e pelo Decreto nº 5.163, de 30 de julho de 2004. (CÂMARA DE COMERCIALIZAÇÃO DE ENERGIA ELÉTRICA, 2016).

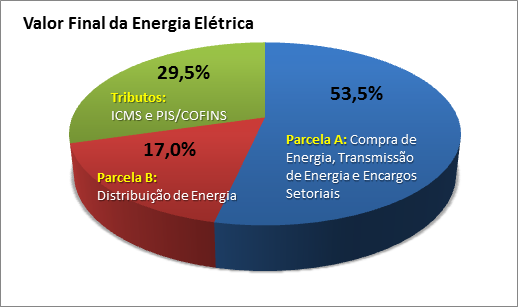
Desse modo, a tarifa está dividida em duas parcelas. A Parcela A, que representa a Compra de energia, Transmissão e Encargos e, a Parcela B, que representa os custos diretamente gerenciáveis pela distribuidora: Custos Operacionais, Receitas Irrecuperáveis, Remuneração de Capital e Cota de Depreciação. A revisão tarifária periódica é realizada a cada quatro anos em média e atualmente cada componente pode ser revista separadamente, desse modo, o processo de revisão se torna mais efetivo e eficaz.

Está também em vigor desde 2015, o Sistema de Bandeiras Tarifárias, que é responsável por sinalizar os consumidores os custos reais da geração de energia elétrica, e são divididos em três cores:

1. bandeira verde: Condições favoráveis de geração de energia. O valor da conta terá a média que o consumidor está habituado a pagar, não sofre nenhum acréscimo;
2. bandeira amarela: Condições de produção de energia um pouco mais caras, sofre acréscimo de R$ 0,0015 para cada quilowatt-hora (KWh) consumidos;
3. bandeira vermelha: Condição desfavorável, energia mais cara;
4. patamar 1: Sofre um acréscimo de R$ 0,030 para cada quilowatt-hora kWh consumido;
5. patamar 2: Sofre um acréscimo de R$ 0,045 para cada quilowatt-hora kWh consumido.

Sendo assim, o consumidor final paga pela compra da energia, transmissão (custos da transmissora) e pela distribuição (serviços prestados pela distribuidora), além dos encargos setoriais e tributos.

Figura : Valor energia elétrica



Fonte: ANEEL (2016).

Segundo a CCEE, o Brasil tem um modelo diferenciado de operação comparado ao restante do mundo, e no país há um enorme potencial de fontes de energia renováveis a ser explorado.

De acordo com a ANEEL, a grande maioria das unidades geradoras de energia são sistemas hidrelétricos, que correspondem cerca de 70% da capacidade do Brasil. Em segundo é a energia termelétrica, com 28% e logo em seguida a energia eólica, com 0,97%.

Fontes renováveis são consideradas soluções apropriadas e nosso país tem o privilégio de tê-las em quantidade, porém não são utilizadas adequadamente. José Goldemberg e José Roberto Moreira (2005, p. 225) afirmam que “O Brasil dispõe de grandes recursos, porém, as reservas utilizáveis são relativamente modestas”.

## 2.2 Hardware

O hardware, representa a parte física do computador. Segundo Tanembaum (2010, p.5), “O hardware consiste em objetos tangíveis - Circuitos integrados, placas de circuito impresso, cabos, fontes de energia, memórias e impressoras - em vez de ideias abstratas, algoritmos ou instruções”.

Os equipamentos necessários para a construção do projeto são: sensor não invasivo (100A SCT-013), ethernet shield W5100 e o Arduino Mega. Além desses três componentes principais, também é necessário para o funcionamento do circuito elétrico componentes secundários para a sua ligação: jumpers, resistores, capacitores e protoboard. A seguir, está a descrição de cada um dos três componentes principais e o motivo da escolha para o projeto.

### 2.2.1 Sensor não-invasivo (100A SCT-013)

O sensor não invasivo é o componente responsável pela captação da corrente elétrica que é enviada para o Arduino, sendo assim, ele é ligado ao Arduino e também acoplado ao fio da rede elétrica desejado para realizar a captação. O modelo utilizado é o SCT013, representado na figura 2.

As especificações seguintes foram coletadas no próprio site do fabricante (YHDC). O Sensor de Corrente SCT013 é um componente eletrônico desenvolvido pela empresa chinesa YHDC (Yaohua Dechang) para aplicação e construção de diversos circuitos elétricos. Utilizado comumente por hobistas e projetistas para o desenvolvimento de projetos, o SCT013 é capaz de realizar o monitoramento de corrente de até 100A, proteção de motores AC, iluminação e diversos outros sistemas elétricos por meio de ferramentas de prototipagem como: Arduino, microcontroladores AVR, microcontroladores PIC e Raspberry PI.

O fator determinante para a escolha do sensor foi pelo fato dele não ser invasivo, ou seja, não há necessidade de alterar nenhuma característica do fio, apenas enganchar o sensor sobre ele, possibilitando ao projeto mais flexibilidade e facilidade em testes e na instalação do equipamento no ambiente desejado.

Figura : Sensor não-invasivo (100A SCT-013)



Fonte: YHDC

### 2.2.2 Ethernet shield W5100

O módulo Ethernet Shield é um dispositivo eletrônico, que é encaixado ao Arduino, que permite a conexão com a internet através de um cabo de rede com conector RJ45. O modelo escolhido é o W5100, representado na figura 3.

As especificações seguintes foram coletadas na página de referências do Arduino. O W5100 é baseado no chip de internet da empresa Wiznet, suporta até quatro conexões simultâneas através de uma conexão UDP (User Datagram Protocol) ou TCP (Transmission Control Protocol), que pode atuar como cliente ou servidor. Comunica-se com o Arduino através de *Serial Peripheral Interface bus* (SPI), e no caso do Mega, são usados os pinos digitais 50, 51,52 e o 10.

No projeto, o módulo é responsável em enviar os dados do consumo, do Arduino para o web *server*, onde os dados ficam armazenados de forma centralizada, através das funcionalidades disponíveis em sua biblioteca para o Arduino, a ethernet Library (Ethernet.h).

O modelo foi escolhido por ser compatível com o Arduino Mega e ser de baixo custo.

Figura : Ethernet Shield W5100



Fonte: W5100

### 2.2.3 Arduino Mega

Arduino é uma plataforma de prototipagem eletrônica de hardware livre e placa única, projetada por um microcontrolador Atmel AVR com suporte de entrada/saída embutido. Evans, Noble e Hochenbaum descrevem:

Originalmente concebido para ser utilizado como uma ferramenta para projetos de computação por designers e estudantes de arte, o Arduino tem sido adotado como ferramenta preferida das comunidades de desenvolvedores e fabricantes interessados na construção e prototipagem de seus próprios projetos. (EVANS, NOBLE E HOCHENBAUM, 2013, p. 24)

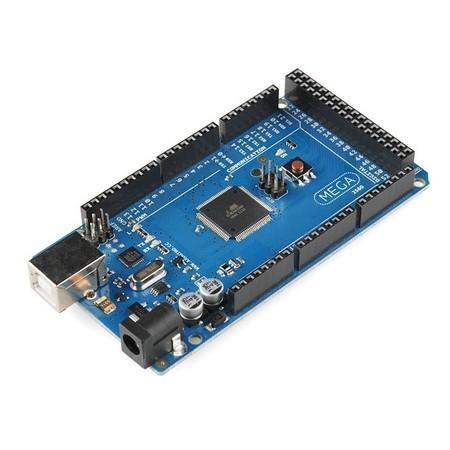
O modelo escolhido foi o Arduino Mega 2560, representado na figura 4.

As especificações seguintes foram coletadas na página de referências do Arduino. O Arduino Mega é uma placa microcontroladora baseada no chip Atmega2560. Possui 54 pinos digitais de entrada e saída, 16 pinos de entradas analógicas, 4 UARTS (Universal Asynchronous Receiver/Transmitter), conexão USB (Universal Serial Bus), uma entrada para fonte, um ICSP (In-Circuit Serial Programming) header e um botão de reset. Além de ser compatível com a maioria dos Shields desenhados para o modelo uno.

No projeto é utilizado como hardware principal, responsável pelo controle do módulo Ethernet Shield e do Sensor de Corrente SCT013. É instalado no ambiente desejado para realizar o monitoramento de consumo de energia.

Foi escolhida por ser uma placa acessível, em questão de preço e disponibilidade de componentes.

Figura : Arduino Mega



Fonte: Arduino

## 2.3 Software

Software são programas escritos para serem interpretados por um computador. Em seu livro, Tanembaum (2010, p.5) descreve software como: “O software, entretanto, consiste em algoritmos (instruções detalhadas que dizem como fazer algo) e suas representações no computador, o que chamamos de programar”.

Os softwares utilizados para a criação do projeto são: IDE Code Envy, Arduino IDE, biblioteca OpenEnergyMonitor, banco de dados PostgreeSQL, Heroku Postgre, Fritzing para a modelagem do circuito eletrônico, ArgoUML para a modelagem da UML e brModelo para a modelagem do banco de dados. Também é abordado as linguagens utilizadas: Haskell e seu compilador GHCI, Framework Web Yesod, HTML, CSS, JavaScript, Bootstrap e UML.

A seguir está a descrição de cada ferramenta e o motivo da escolha para o projeto.

### 2.3.1 Haskell

Haskell é uma linguagem de proposito geral baseada nas observações de Haskell Curry, sua estrutura é composta por funções, e possui seu foco voltado para o que deve ser feito, e não para como é feito. De acordo com O’Sullivan, Goerzen e Stewart, Haskell é uma linguagem de programação de propósito geral. Ela foi designada sem nenhum nicho de aplicação em mente. Embora leve um grande entendimento em como os programas devem ser escritos, ele não favorece um domínio do problema sobre outro.

Haskell é uma linguagem puramente funcional, *lazy* (não executa funções até que seja forçado a mostrar o resultado), fortemente tipada (quando executado, o compilador sabe exatamente qual o tipo de cada parte do código), e seu código é considerado elegante (sua sintaxe possui clareza, e se comparado a outras linguagens, é utilizado menos linhas de código para a execução de uma mesma tarefa).

LIPOVAČA (2011) diz que na programação puramente funcional você não diz para o computador o que fazer, porém muitas vezes você diz em qual coisa está. O fatorial de um número é o produto de todos os números sobre 1 deste número, a soma de uma lista de números é o primeiro mais a soma de todos os outros números e assim por diante. Você expressa isto na forma de funções

Para começar a desenvolver em Haskell, é necessário apenas de um compilador da linguagem, o mais utilizado é o GHCI, e de um editor de texto.   
 No projeto, Haskell é a linguagem de programação utilizada para o desenvolvimento do sistema através do framework web Yesod.

### 2.3.2 GHCI

GHC ([Glasgow Haskell Compiler](https://www.haskell.org/ghc/docs/latest/html/users_guide/ghci.html)) é um compilador e ambiente interativo open-source para a linguagem de programação Haskell. Segundo LIPOVAČA (2011), GHC pode pegar um script Haskell (que normalmente tem uma extensão .hs) e compilá-lo mas ele também tem um modo interativo que permite que você interativamente interaja com os scripts. Interativamente. Pode-se chamar funções de scripts que foram carregadas e os resultados são exibidos imediatamente.   
 O GHC é utilizado no projeto para compilar o programa escrito em Haskell que possui a extensão (.hs).

### 2.3.3 Modelo REST

REST (Representational State Transfer) é um design de arquitetura construído para servir aplicações em rede. De acordo com Richards.

Os componentes nesta arquitetura executam ações no recurso usando sua representação. Por exemplo, usando um navegador para solicitar o recurso, um servidor transfere o estado atual. De uma página da Web para o navegador. O navegador pode então executar a ação de renderizar a representação. Simplificando, o servidor envia os dados para a página solicitada na sua em seu estado atual, e os dados são processados. Navegar para qualquer um dos links dentro da página renderizado causa uma transição de estado por causa da próxima página, que poderia ser considerado outro estado de uma aplicação, é transferido para o navegador para renderização. (Richards, 2006, p. 633)

### 2.3.4 Yesod

Yesod é um framework *open-source* baseado em Haskell desenvolvido por Michael Snoyman. Em seu artigo, Snoyman (2014) descreve Yesod como um framework de aplicação web gratuita para desenvolvimento produtivo de *type-safe* (consegue especificar a entrada exata que se espera), baseado em modelo REST (onde as URL’s identificam recursos, e os métodos HTTP identificam transações) e aplicações web de alta performance. De acordo com Snoyman, Yesod:

Yesod tenta facilitar o processo de desenvolvimento web usando os pontos fortes da linguagem de programação Haskell. O forte tempo de compilação de haskell provém garantias de correção não só acerca de tipos; Transparencia referencial garante que não tenhamos qualquer efeito colateral não intencional. *Pattern matching* em tipos de dados algébricos podem ajudar a garantir que tenhamos levado em conta todos os casos possíveis. Ao construir em Haskell, classes inteiras de bugs desaparecem. (SNOYMAN, 2012, p. 3)

Para a criação das interfaces, o Yesod utiliza o Shakespearean *templates* para a criação de HTML, CSS e JavaScript. De acordo com Hackage, Shakespeare é uma família de *type-safe* e eficientes *templates* de linguagens. Shakespeare templates são expandidos em tempo de compilação, garantindo que todas as variáveis interpoladas estejam no escopo. Variáveis são interpoladas de acordo com seus tipos através de *typeclass*.

O Shakespearean é divido em quatro *templates* linguagens: Hamlet, Julius, Celsius e Lucius. Hamlet é feito para a criação de HTML. Julius para criação de JavaScript. E Cassius e Lucius para criação do CSS. Snoyman explica.

Hamlet e Cassius são ambos formatos sensíveis a espaço, usando indentação para indicar o aninhamento. Em contraste, Lucius é um superconjunto de CSS, mantendo o suporte CSS para denotar aninhamento. Julius é uma simples linguagem de passagem para a produção de Javascript; O único recurso adicionado é a interpolação de variáveis. (SNOYMAN, 2012, p. 23)

No projeto, o framework é utilizado como o *back-end*, responsável pelo *web-serve*, que recebe e armazena os dados vindos do Arduino, e também pelas funções do *web-app*. O framework também é utilizado para o *front-end*, utilizando o Shakespearean *templates*, responsável pela interface de usuário do *web-app*. Yesod foi escolhido por ser um *framework* web de alta performance utilizado na linguagem Haskell, e por ser a mais utilizada, que resulta em mais facilidade para se obter informações.

### 2.3.5 HTML

HTML (*Hypertext Markup Language* /Linguagem de Marcação de Hipertexto) é uma linguagem que descreve como o texto e todas as outras informações da página web são organizados, estruturados e linkados em sua construção.

MELONI (2015, p. 3) fala que código HTML que informa o navegador como exibir o texto - como um cabeçalho, como um parágrafo, em uma fonte vermelha e assim por diante. Algumas marcações HTML dizem o navegador para exibir uma imagem ou vídeo em vez de texto sem formatação.

No projeto, o HTML é utilizado para realizar a criação da estrutura da interface do *web-app, através do* Shakespearean *templates* Hamlet.

### 2.3.6 CSS

CSS (Cascading Style Sheets) é uma linguagem  [*style sheet*](https://en.wikipedia.org/wiki/Style_sheet_language)que define a aparência de uma página *web*. De acordo com MELONI (2015, p. 45), o conceito por trás de folha de estilos é simples. Você cria um documento de folha de estilo que especifica as fontes, cores, espaçamento e outras características que estabelecem uma aparência única para o seu website.

No projeto, o CSS é utilizado para criar a aparência da interface do *web-app*, junto com o *framework* bootstrap, através do Shakespearean *templates* Lucius.

### 2.3.7 JavaScript

JavaScript é uma linguagem de programação de alto-nível, dinâmica, não tipada e interpretada. A linguagem é baseada em objetos, e é bastante utilizada para scripts em páginas web. CIMO explica:

Ao lado de HTML e CSS, é uma das três tecnologias essenciais da produção de conteúdo de World Wide Web; A maioria dos sites o empregam e é suportado por todos os navegadores dos modernos sem plug-ins ou qualquer tipo de outra extensão. *(CIMO, 2016, p. 1)*

No projeto o JavaScript é utilizado para realizar as interações da página, através do Shakespearean *templates* Julius.

### 2.3.8 Bootstrap

Bootstap é um framework HTML, CSS e JS para desenvolvimento de projetos responsivo na *web* baseado para dispositivos móveis. Conforme CIMO (2015, p. 1) Bootstrap inclui um responsivo, primeiro sistema *fluid grid* móvel que dimensiona adequadamente até 12 colunas enquanto o dispositivo ou a janela de exibição aumenta de tamanho.

No projeto, o bootstrap é utilizado para deixar a interface do *web-app* responsiva e acessível em diferentes tamanhos de telas, celulares, tabletes e computadores.

### 2.3.9 Codenvy

Codenvy, representado pela figura 5, é uma IDE (*Integrated Development Environment*) na nuvem que permite o desenvolvimento de softwares em diversas linguagens de programação. Segundo a própria empresa, ele torna o desenvolvimento mais ágil ao fornecer espaços de trabalho que são *on-demand*, colaborativos e livres de restrições

É utilizada no projeto para o desenvolvimento do programa com o Yesod, e também responsável por efetuar o papel de servidor de aplicação.

Figura : Codenvy



Fonte: Codenvy, 2017

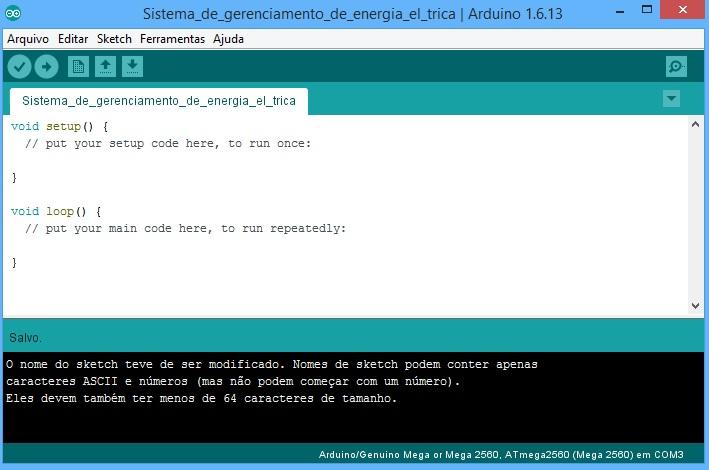
### 2.3.10 Arduino IDE

Para realizar a comunicação com o Arduino, é necessário um software de IDE (*Integrated Development Environment*). Por padrão, a IDE utilizada para o projeto é a do próprio Arduino, Arduino Software (IDE), que é uma IDE de código aberto e contém características básicas, como um editor de texto para escrever os códigos, uma área de mensagem e console. O ambiente é escrito em Java e baseado em outros softwares de código aberto. Programas escritos no Arduino Software IDE são chamados de sketches, e possuem extensão .ino.

O software IDE Arduino fornece tudo o que é necessário para programar o Arduino, incluindo vários programas ou esboços de exemplo que demonstram como conectá-lo e comunicar com alguns dispositivos comuns, como LEDs, LCDs e alguns sensores. (EVANS, NOBLE E HOCHENBAUM, 2013, p.10)

No projeto, a IDE é utilizada para a comunicação com o Arduino, para controlar o sensor não invasivo através código de captação da corrente elétrica utilizando a biblioteca EmonLib.h, e o Ethernet Shield, para enviar os dados para o *web-server*, utilizando a biblioteca Ethernet.h. A versão utilizada é a 1.6.13, representada na figura 6.

Figura : IDE Arduino



Fonte: Arduino, 2017

### 2.3.11 OpenEnergyMonitor

OpenEnergyMonitor é um projeto *open-source* de monitoramento de energia elétrica. Ele é descrito em sua documentação como um projeto para desenvolver ferramentas de monitoramento de energia de fonte aberta, que nos ajudem a relacionarmos com nosso uso de energia, nossos sistemas de energia e o desafio de energia sustentável.

O projeto é destinado a pessoas que queiram desenvolver alternativas para o monitoramento do consumo, em busca de um desenvolvimento sustentável. De acordo com eles, o projeto teve início em primeiro lugar para a nossa própria curiosidade e compreensão.

O site do projeto possui guias de como montar circuitos eletrônicos de características variadas. Possui uma área de aprendizado, onde é apresentada toda parte teórica para a elaboração dos seus projetos, como por exemplo, noções e conceitos sobre cálculo para a utilização de resistores. Há também um fórum, onde qualquer pessoa pode deixar sua contribuição para a comunidade, com dúvidas, respostas, tutoriais, debates, etc. O OpenEnergyMonitor também possui seu próprio sistema de gerenciamento de energia elétrica e componentes à venda em sua loja virtual que ajudam a manter o projeto.

Em nosso projeto é utilizada a biblioteca do OpenEnergyMonitor (EmonLib.h) como base para escrever o código de captação de corrente elétrica e a realização do cálculo de consumo. Também é utilizado o guia como base para a construção do circuito eletrônico com o Arduino.

### 2.3.12 PostgreSQL

PostgreSQL é um sistema gerenciador de banco de dados objeto relacional (SGBDOR) *open-source* baseado no Postgres e coordenado pelo PostgreSQL Global Development Group. Descrito por eles, com informações apresentadas em seu próprio site, como uma ferramenta poderosa, capaz de executar nos sistemas operacionais mais importantes, como Linux, UNIX e Windows, com mais de 15 anos ativo de desenvolvimento, responsável por atribuir a eles uma grande reputação. De acordo com sua documentação, ele foi desenvolvido na Universidade da Califórnia no departamento de ciência da computação de Berkeley. A POSTGRES foi pioneira em muitos conceitos que só mais tarde ficaram disponíveis em alguns sistemas de banco de dados comerciais. PostgreSQL é um descendente de código aberto deste código original de Berkeley. Ele suporta uma grande parte do padrão SQL e oferece muitos recursos modernos

No projeto o PostgreSQL é usado para o armazenamento do consumo e de todas as informações do sistema. Foi escolhido por ter integração com o serviço do Heroku Postgres.

### 2.3.13 Heroku Postgres

Heroku Postgres é um serviço de banco de dados SQL, gerenciado pelo Heroku, que é provisionado e gerenciado como um *add-on* (por definição é um artifício que pode modificar e adicionar novas funções de maneira significativa à interface principal). Heroku Postgres está disponível para qualquer linguagem de programação e framework desde que se tenha o driver PostgreSQL instalado.

Espake (2015, p. 13) cita que o PostgreSQL é fornecido no Heroku através de um *add-on*. Este add-on tem um plano inicial gratuito, apenas para pequenas aplicações, como sites ou blogs e, também está disponível em uma série de planos pagos, de acordo com sua necessidade.

No projeto, o serviço é utilizado para guardar os dados do sistema. Foi escolhido, pois suas informações ficam armazenadas na nuvem.

De acordo com Espake (2015, p. 1), Heroku Postgres permite que você gerencie seus bancos de dados PostgreSQL de uma maneira simples, sem preocupações e de qualquer lugar. Através dos recursos oferecidos, você pode facilmente escalar seu banco de dados e estender funcionalidades.

### 2.3.14 Fritzing

Fritzing é um software de código aberto para a prototipagem e modelagem de circuitos eletrônicos. Como descrito em seu site, a ferramenta permite aos usuários documentar os protótipos, compartilhar com outras pessoas e até planejar e fabricar dispositivos profissionais. Monk (2015) explica que Fritzing foi projetado e desenvolvido para hobbistas e inventores, não para profissionais da eletrônica.

Monk (2015) também completa, Fritizing é tão fácil de usar que muitas pessoas usam para esboçar *layouts* de *breadboard* ou desenhar diagramas esquemáticos, por ser tão fácil de se fazer quanto usar caneta e papel.

Fritzing é utilizado no projeto para mostrar a montagem do circuito eletrônico utilizado e explicar a ligação do Arduino e seus componentes. Foi escolhido por ser simples e prático. A versão utilizada é a 0.9.3, representado pela figura 7.

Figura : Fritzing versão 0.9.3



Fonte: Autores, 2017

### 2.3.15 UML

UML (*Unified* *Modeling* *Language*) é uma linguagem para estruturação de um sistema de software, usada para especificação, construção, visualização e documentação de artefatos, sem depender do seu processo de desenvolvimento, como por exemplo, da linguagem de programação utilizada para executar o projeto. SILVA e VIDEIRA explicam:

A ênfase do UML é na definição de uma linguagem de modelação standard, e, por conseguinte, o UML é independente das linguagens de programação, das ferramentas CASE, bem como dos processos de desenvolvimento. O objetivo do UML é que, dependendo do tipo de projeto, da ferramenta de suporte, ou da organização envolvida, devem ser adoptados diferentes processos/metodologias, mantendo-se, contudo a utilização da mesma linguagem de modelação. (SILVA e VIDEIRA, 2001, p. 118)

A linguagem UML é utilizada no projeto para a modelagem dos casos de uso, através do ArgoUML.

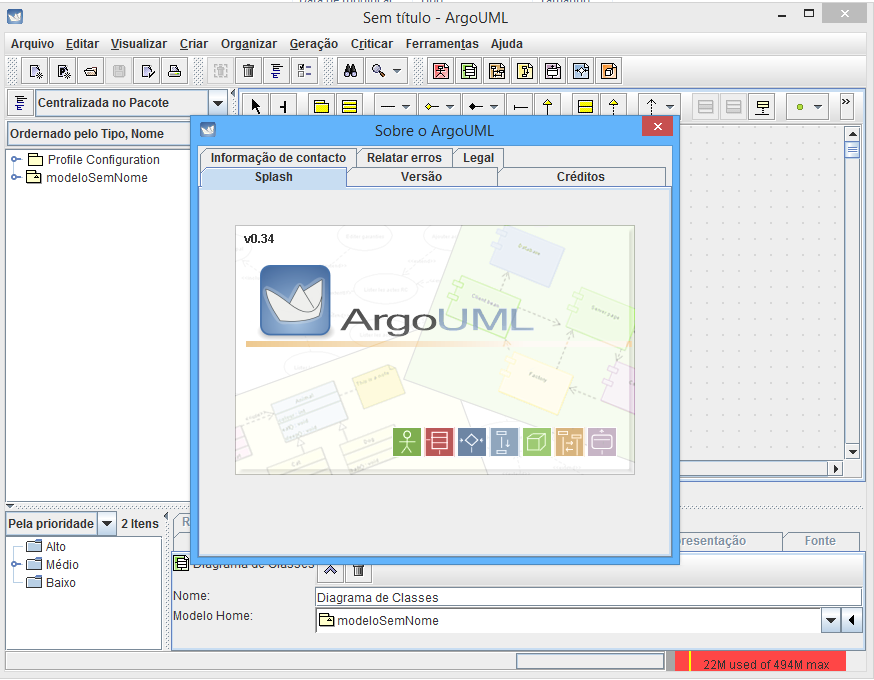
### 2.3.16 ArgoUML

ArgoUML é um programa *open-source* para a modelagem da UML. Assim como descrito no manual de usuário deles, o ArgoUML é:

ArgoUML é um ambiente de design orientado a domínios que fornece suporte cognitivo ao design orientado a objetos, ArgoUML fornece alguns dos mesmos recursos de automação de uma ferramenta CASE comercial, mas se concentra em recursos que suportam as necessidades cognitivas dos designers. (ARGOUML, [200?], p. 17).

ArgoUML é utilizado no projeto para a modelagem dos casos de uso. A versão utilizada é a 0.34, representada na figura 8.

Figura : ArgoUML versão 0.34



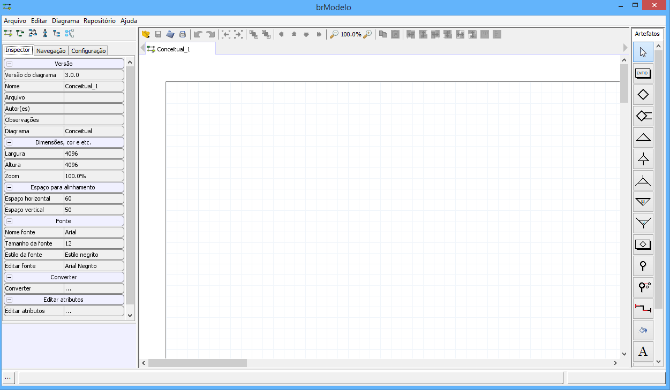
Fonte: Autores, 2017

### 2.3.17 BrModelo

BrModelo é uma ferramenta de modelagem de banco de dados *freeware*, com base na metodologia de Carlos A. Heuser. A ferramenta foi desenvolvida por Carlos Henrique Cândido sob a orientação do Prof. Dr. Ronaldo dos Santos Mello da Universidade Federal de Santa Catarina (UFSC), como trabalho de conclusão do curso de pós-graduação em banco de dados.

No projeto, o brModelo é utilizado para a modelagem do modelo DER e do modelo lógico do banco de dados. A versão utilizada é a 3.0, representada na figura 9.

Figura : brModelo versão 3.0



Fonte: Autores, 2017

# 3 METODOLOGIA

Apresentado o contexto do projeto, as ferramentas e os componentes necessários, neste capitulo é explicado a metodologia aplicada para a construção do projeto.

O processo de desenvolvimento do projeto segue a sequência: engenharia de software, montagem do circuito eletrônico, regulagem do sensor, calculo do consumo e a criação da interface do *web-app*.

## 3.1 Engenharia de Software

Pressman define engenharia de software como:

Ela abrange um conjunto de três elementos fundamentais – métodos, ferramentas e procedimentos – que possibilita ao gerente o controle do processo de desenvolvimento do software e oferece ao profissional uma base para a construção de software de alta qualidade produtivamente. (PRESSMAN, 1995, p. 31)

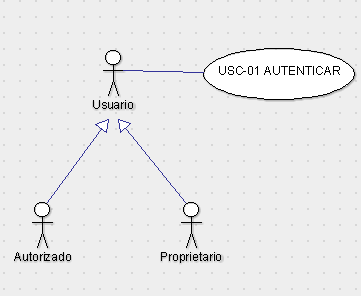
Aqui está especificado como o sistema foi construído e o seu processo de desenvolvimento.  Como a criação do banco de dados, diagramas de caso de uso, regras de negócio e requisitos funcionais.

### 3.1.1 Diagrama de caso de uso

Para melhor visualização, o caso de uso foi dividido em três imagens: usuário, proprietário e autorizado.

A primeira imagem, mostrada na figura 10, é a representação do caso de uso do ator usuário, que pode ser autorizado ou proprietário. Proprietário é o usuário que fez o cadastro no sistema, é o administrador da casa em que será realizado o consumo. O autorizado é o usuário que o proprietário concede permissão para também realizar consulta no consumo da sua casa. O usuário possui um relacionamento de associação ao caso de uso autenticar, portanto, para acessar as funcionalidades do sistema, é necessário realizar a autenticação.

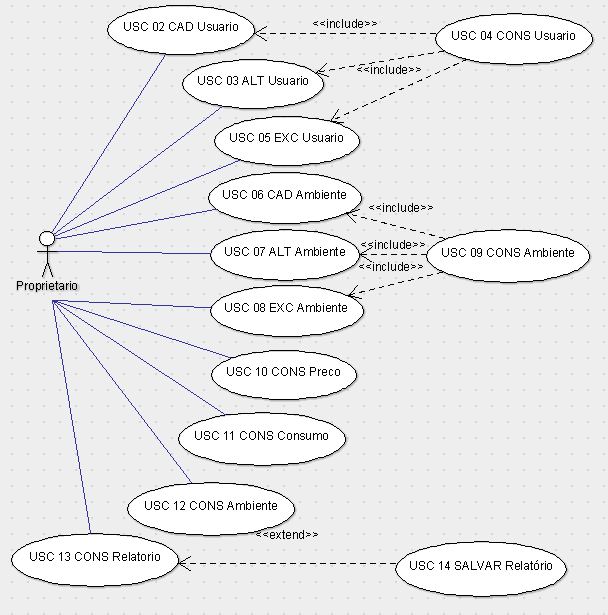
Figura : Caso de uso Usuário



Fonte: Autores, 2017

O caso de uso dois, mostrado na figura 11, é a representação dos relacionamentos do ator proprietário. O ator proprietário é o administrador do sistema, que pode cadastrar, excluir e alterar um usuário do tipo autorizado. Cadastrar, alterar ou deletar um ambiente da casa. Consultar o consumo e o relatório, podendo salva-lo.

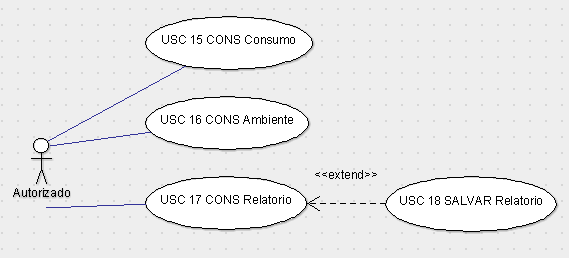
Figura : Caso de uso Proprietário



Fonte: Autores, 2017

O caso de uso três, representado pela figura 12, é a representação dos relacionamentos do ator autorizado. O ator autorizado é um tipo de usuário cadastrado pelo proprietário do sistema para obter acesso ao consumo da casa. O autorizado pode consultar o consumo, o ambiente, o relatório e salvar o relatório.

Figura : Caso de uso Autorizado

****

Fonte: Autores, 2017

### 3.1.2 Regras de negócio

A partir do caso de uso e da modelagem do banco de dados foram geradas as seguintes regras de negócio descritas na tabela 1.

Tabela : Regras de Negócio

|  |  |
| --- | --- |
| RN-001: | O usuário é uma pessoa cadastrada no sistema, que pode ser um proprietário ou um autorizado. |
| RN-002: | O proprietário é um tipo de usuário que tem controle sobre todas funcionalidades do sistema |
| RN-003: | O autorizado é um tipo de usuário que tem autorização para consultar o consumo da casa, e deve ser cadastrado pelo proprietário |
| RN-004: | Antes de realizar qualquer função do sistema, o usuário precisa ser autenticado |
| RN-005: | A casa é a representação da residência do usuário, onde são cadastrados os ambientes para o gerenciamento do consumo. |
| RN-006: | Um ambiente é a representação de um cômodo da casa em que está instalado um arduino para o gerenciamento de consumo, e dever ser cadastrado pelo proprietário. |
| RN-007: | Os dispositivos são eletrônicos cadastrados no ambiente para que se possa obter um controle de gerenciamento de quais dispositivos estão vinculados a aquele cômodo. O dispositivo deve ser cadastrado pelo administrador, e devem pertencer obrigatoriamente a um ambiente |
| RN-008: | No ato de cadastro (autorizado, ambiente), todos os campos são obrigatórios. |
| RN-009: | No ato de excluir e alterar (autorizado, ambiente), deve se realizar uma consulta que verifique se o autorizado ou ambiente existe no sistema. |
| RN-010: | O preço da energia por kWh é o valor da energia elétrica que o sistema usa como base para calcular os gastos, e é definida pelo proprietário. |
| RN-011: | Todo histórico de consumo é salvo, e pode ser consultado a qualquer momento pelo usuário. |
| RN-012: | No ato de consultar o consumo, o usuário pode apenas visualizar o gasto estimado na tela ou salvar relatório. |
| RN-013: | O relatório é o arquivo com os gastos estimados, pode ser consultado e salvo. |

Fonte: Autores, 2017

### 3.1.3 Requisitos funcionais

A partir do caso de uso e da modelagem do banco de dados foram gerados os seguintes requisitos funcionais descritos na tabela 2:

Tabela : Requisitos funcionais

|  |  |
| --- | --- |
| RF-001: | **Autenticar usuário**  Este requisito faz a autenticação do usuário através de seu login e senha e, em seguida, exibe um menu de opções com as operações disponíveis no sistema. |
| RF-002: | **Cadastrar Ambiente**  O proprietário pode cadastrar Ambientes. |
| RF-003: | **Alterar nome do Ambiente**  O proprietário pode alterar o nome de um ambiente |
| RF-004: | **Deletar Ambiente**  O proprietário pode deletar um ambiente |
| RF-005: | **Consultar consumo total**  O usuário pode consultar o consumo de todos os dispositivos durante um período pré-determinado por ele |
| RF-006: | **Consultar consumo por ambiente**  O usuário pode consultar o consumo de um ambiente de dispositivos durante um período pré-determinado por ele |
| RF-007: | **Gerar relatório**  O usuário pode salvar o consumo estimado |
| RF-008: | **Alterar preço do Quilowatts/hora**  O preço da energia por kWh é definido automaticamente pelo sistema |
| RF-009: | **Calcular gastos**  O sistema calcula os gastos de acordo com preço da emergia por kWh. |
| RF-010: | **Cadastrar usuário**  O proprietário pode cadastrar usuário |
| RF-011: | **Excluir usuário**  O proprietário pode excluir usuário |
| RF-012: | **Alterar usuário**  O proprietário pode alterar dados do usuário |
| RF-013: | **Consultar informações**  O proprietário pode consultar usuário, ambiente e consumo cadastrados no sistema. |

Fonte: Autores, 2017

## 3.2 Banco de dados

O banco de dados, baseado no PostgreSQL e hospedado no serviço do Heroku Postgres, é a base em que é armazenado o consumo da energia elétrica, e todas as informações do sistema. Nos tópicos seguintes é explicado o modelo conceitual e lógico do banco de dados do projeto. Heuser (1998, p. 5) explica que um modelo de (banco de) dados é uma descrição dos tipos de informações que estão armazenadas em um banco de dados.

### 3.2.1 Modelo DER

Neste tópico está explicada a estrutura do modelo conceitual do banco de dados representado através do diagrama de entidade-relacionamento (modelo DER). Heuser define modelo conceitual como:

Um modelo conceitual é uma descrição do banco de dados de forma independente de implementação em um SGBD. O modelo conceitual registra que dados podem aparecer no banco de dados, mas não registra como estes dados estão armazenados a nível de SGBD. (HEUSER, 1998, p. 5).

A estrutura do banco de dados é composta por cinco tabelas, representada pela figura 13.

A tabela usuário representa a pessoa que utiliza o sistema. Composta pelo número de identificação do usuário (cdUsuario), o nome de usuário (nmUsuario), a senha (cdPassword), e o tipo (tipoUsuario), que pode ser autorizado ou proprietário.

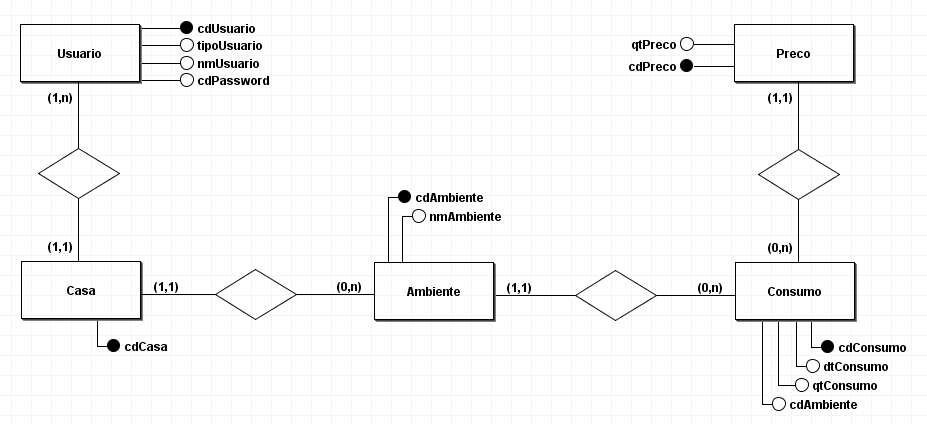
A tabela casa representa o local físico em que são gerenciados os ambientes monitorados, e é composta pelo número de identificação (cdCasa). A casa é ligada ao usuário proprietário, gerada através do cadastro, porém, ela pode ter vários usuários pelo fato de o proprietário poder cadastrar usuários autorizados para realizarem consultas no consumo da sua casa.

A tabela ambiente representa o local físico em que o Arduino está instalado. Composta pelo número de identificação (cdAmbiente) e pelo nome do ambiente (nmAmbiente).

A tabela consumo representa o consumo armazenado que é gerado através do Arduino e o sensor-não invasivo, e enviada para o *web-serve,* através do Ethernet Shield. Composta pelo código de identificação do consumo (cdConsumo), data em que foi realizado o consumo (dtConsumo) e pelo o valor do consumo, gerado no período em um período de vinte e quatro horas, em quilowatts (qtConsumo).

A tabela preço representa o preço em que é calculado os gastos do consumo, é composta pelo número de identificação (cdPreco), e o valor do preço (qtPreco).

Figura : Modelo DER



Fonte: Autores, 2017

### 3.2.2 Modelo lógico

A partir do modelo conceitual, foi gerado o modelo lógico do banco de dados, representado pela figura 14.

“Um modelo lógico é uma descrição de um banco de dados no nível de abstração visto pelo usuário do SGBD. Assim, o modelo lógico é dependente do tipo particular de SGBD que está sendo usado”. (HEUSER, 1998, Página 6)

Para cada novo cadastro de usuário no sistema, o banco de dados armazena o número de identificação (cdUsuario), o nome do usuário (nmUsuário), sua senha (cdPassword), e o tipo de usuário (tipoUsuario). Também é referenciado, através da chave estrangeira (FK\_Casa\_cdCasa), a casa do usuário, que pode ser um novo registro (em caso do cadastro ser realizado direto na página inicial do *web-app,* tornando o usuário proprietário) ou existente (usuário proprietário cadastrando usuário autorizado, a casa é a mesma).

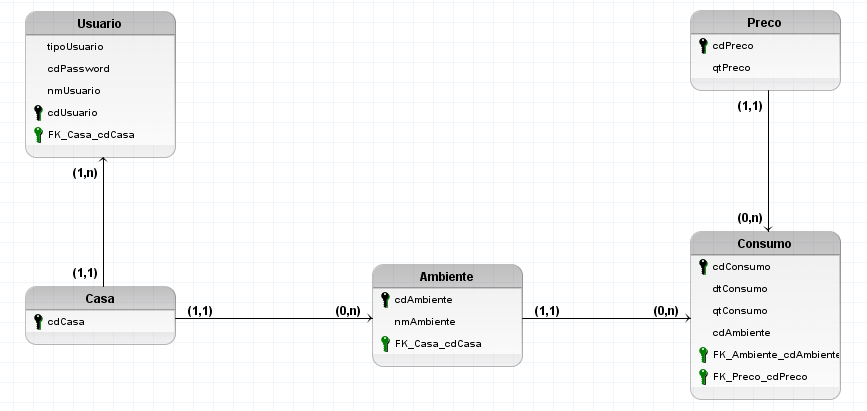
Para cada nova casa, o banco de dados armazena o seu número de identificação (cdCasa).

Para cada novo ambiente na casa, o banco de dados armazena o número de identificação (cdAmbiente), o nome do ambiente (nmAmbiente) e a casa em qual está sendo cadastrado, representado pela chave estrangeira FK\_Casa\_cdCasa.

Para cada novo registro de consumo, o banco de dados armazena o código de identificação do consumo (cdConsumo), a data em que foi realizado o consumo (dtConsumo), o valor do consumo em quilowatts (qtConsumo), o código de identificação do preço correspondente para a realização do cálculo dos gastos do consumo (FK\_Preco\_cdPreco) e o código de identificação do ambiente em que o consumo está sendo gerado, representado pela chave estrangeira FK\_Ambiente\_ambiente.

Para cada novo registro de preço, o banco de dados armazena o código de identificação do preço (cdPreco) e o valor (qtPreco).

Figura : Modelo lógico



Fonte: Autores, 2017

## 3.3 Circuito eletrônico

O circuito eletrônico foi moldado a partir do Fritizing, representado na figura 15. O circuito é composto por: um resistor de 100k Ohm (resistor para divisão de voltagem), um resistor de 10K, um resistor de 33 Ω (para o resistor de carga), quatro resistores de 470kOhm, dois capacitores de 10 µF, um sensor não-invasivo, uma fonte AC-AC, um Arduino Mega e um Ethernet Shield.

Foi moldado com base no tutorial do projeto OpenEnergyMonitor, de modo que além da comunicação do Arduino com o sensor-não invasivo, também tenha a medição da variação da tensão elétrica, realizada através da fonte AC-AC.

Sem a fonte para medir a variação, a voltagem elétrica ficaria fixa (110V ou 230V nos padrões do Brasil) e o consumo não seria muito aproximado ao valor real. A medição de tensão AC é necessária para calcular a potência real, a potência aparente e o fator de potência. Com a fonte, o monitor de energia pode calcular a potência real, a potência aparente, o fator de potência, a tensão RMS e a corrente RMS.

Considerando que o sensor-não invasivo SCT-013-000 (100A) tem na saída uma variação de corrente, o sinal de corrente precisa ser convertido em um sinal de tensão com uma resistência de carga.

O YHDC SCT-013-000 CT tem uma faixa de corrente de 0 a 100 A. Por isso, escolhemos 100 A como nossa corrente máxima. Após isso, é necessário converter a corrente máxima RMS (Root Mean Square) em corrente de pico, multiplicando por √2.

|  |  |  |
| --- | --- | --- |
|  |  | **(6)** |

Onde:

1. CPP = Corrente de pico primária;
2. RMS = Root Mean Square.

Ao substituir os valores temos:

|  |  |  |
| --- | --- | --- |
|  |  | **(7)** |

Após isso, é feito a divisão da corrente de pico pelo número de voltas do CT para dar a corrente de pico na bobina secundária.

O YHDC SCT-013-000 CT possui 2000 voltas, então a corrente de pico secundária será:

|  |  |  |
| --- | --- | --- |
|  |  | **(8)** |

Onde:

1. CPS = Corrente de pico secundário

Resultando em:

|  |  |  |
| --- | --- | --- |
|  |  | **(9)** |

Para maximizar a resolução da medição, a tensão entre a resistência de carga na corrente de pico secundária deve ser igual a metade da tensão de referência analógica do Arduino (AREF).

|  |  |  |
| --- | --- | --- |
|  |  | **(10)** |

Como estamos usando um Arduino rodando em 5V, temos

|  |  |  |
| --- | --- | --- |
|  |  | **(11)** |

Assim, a resistência à carga ideal será:

|  |  |  |
| --- | --- | --- |
|  |  | **(12)** |

Onde:

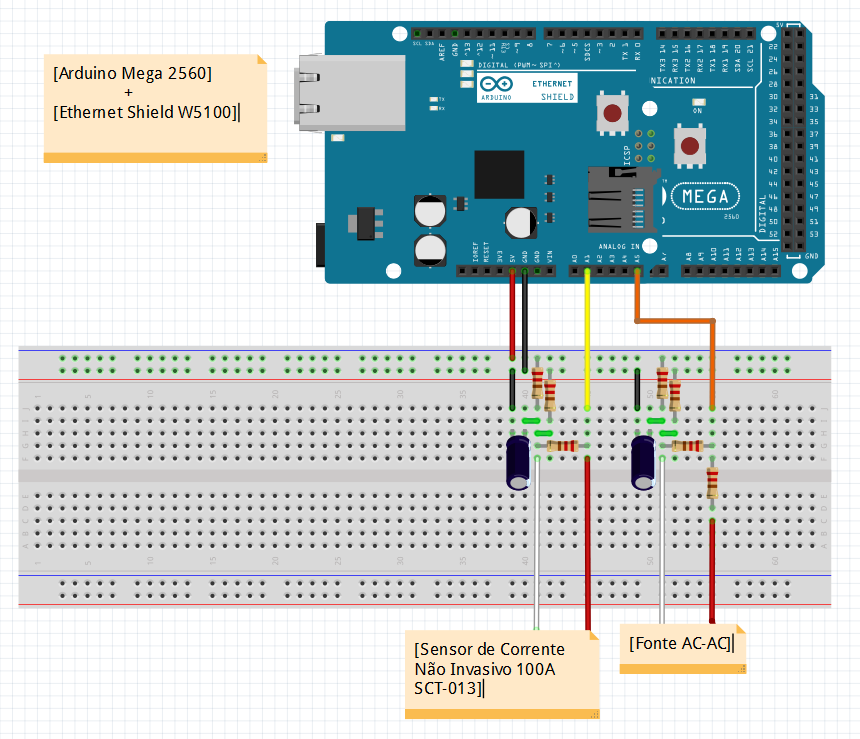
1. RCI representa a Resistência de carga ideal.

Ao calcular temos:

|  |  |  |
| --- | --- | --- |
|  |  | **(13)** |

35 Ω não é um valor de resistência comum. Os valores mais próximos de 35 Ω são 39 e 33 Ω. Foi escolhido o valor menor, ou a corrente de carga máxima criará uma tensão maior do que o AREF.

Figura : Circuito eletrônico



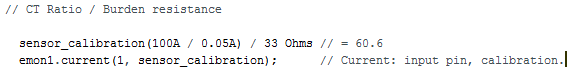
Fonte: Autores, 2017

## 3.4 Regulagem do sensor 100A SCT-013

A regulagem do sensor, mostrada na figura 16, foi feita com base no tutorial do projeto OpenEnergyMonitor. De acordo com OpenEnergyMonitor, é impossível fabricar qualquer coisa com absoluta precisão. Em termos gerais, quanto mais precisamente algo é fabricado, mais caro se torna.

O sensor 100A SCT-013 é calibrado a partir do seu número de voltas dividido pelo o resistor ao qual está conectado. O número de voltas é calculado pela sua faixa de corrente, que varia de 0 a 100 A, dividida por sua faixa secundária 50mA. O resistor utilizado, explicado no tópico anterior, é um resistor de 33 Ohms.

Figura : Regulagem do sensor



Fonte: Autores, 2017

## 3.5 Calculo do consumo

O calculo do consumo é realizado no Arduino no momento da captação da corrente elétrica. Para calcular o consumo é utilizada a formula 4 citada no capitulo 2: E = P × ∆t.

Em um cálculo simples, P representa a potência, que é calculado a partir da corrente multiplicada pela tensão, porém, esse resultado será da potência aparente. Para conseguirmos pegar o valor da potência ativa (watts) é utilizada funções da biblioteca EmonLib.h, que calcula o valor a partir da corrente e dos valores gerados pela fonte AC-AC. A corrente é captada em amperes pelo sensor não invasivo, e a tensão é fornecida através da fonte AC-AC, que é responsável por regular a variação dos valores da tensão elétrica, como explicado no tópico anterior.

O ∆t representa a variação do tempo que se refere ao valor de potência calculado. Como a captação é feita por segundo, a potência gerada sempre será referente àquele segundo. Sendo assim, o consumo final de energia elétrica (E), sempre será a soma de todas as potências em Watts. Por padrão, como é utilizado o kWh para medir o consumo de energia elétrica no Brasil, o valor de E é dividido por 1000, pois o kWh se equivale a mil watts.

Feito o cálculo, o consumo é enviado do Arduino para o web-server, que realiza o cálculo dos custos. O custo é o resultado do consumo multiplicado pelo valor do kWh vigente, representado pela formula 5, também citada no capitulo 2.

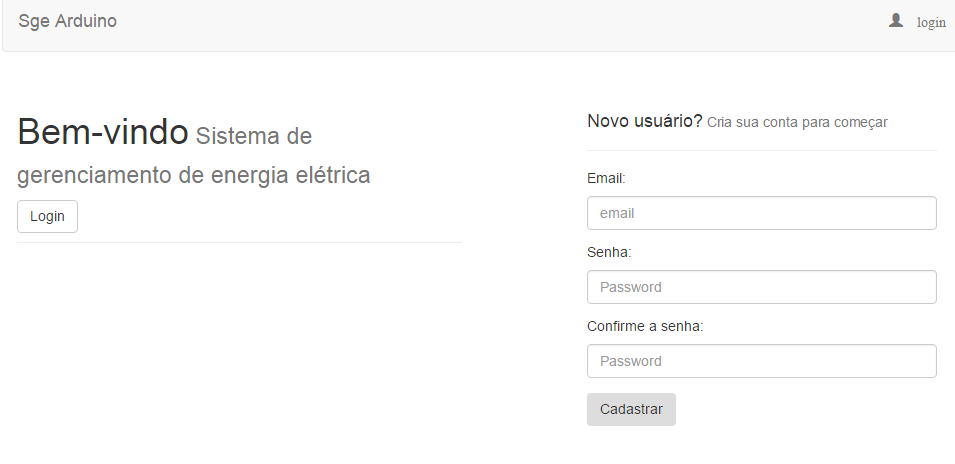
## 3.6 Interface do *web-app*

A interface do *web-app* foi criada para ter um design responsivo, através do framework bootstrap, tornando seu uso acessível e disponível em diferentes tamanhos de telas (computadores, celulares e tabletes).

A seguir estão descritas as principais telas do sistema: a tela Principal, tela de Consumo, tela de Ambientes, tela de Visualização de ambiente, tela de Usuários, tela de Relatório e a tela Sobre. Cada botão da interface representa uma função disponível no *back-end*, e são através delas que o programa realiza suas operações.

A tela principal, mostrada na figura 18, é uma tela de boas-vindas para novos usuários utilizarem o sistema de monitoramento de energia em suas casas, onde um novo usuário pode realizar o cadastro da sua casa. Há também, um botão de acesso (Login) que redireciona para outra página, através da função de autenticação, para usuários já cadastrados realizarem a autenticação no sistema para a sua utilização.

Figura : Tela Principal

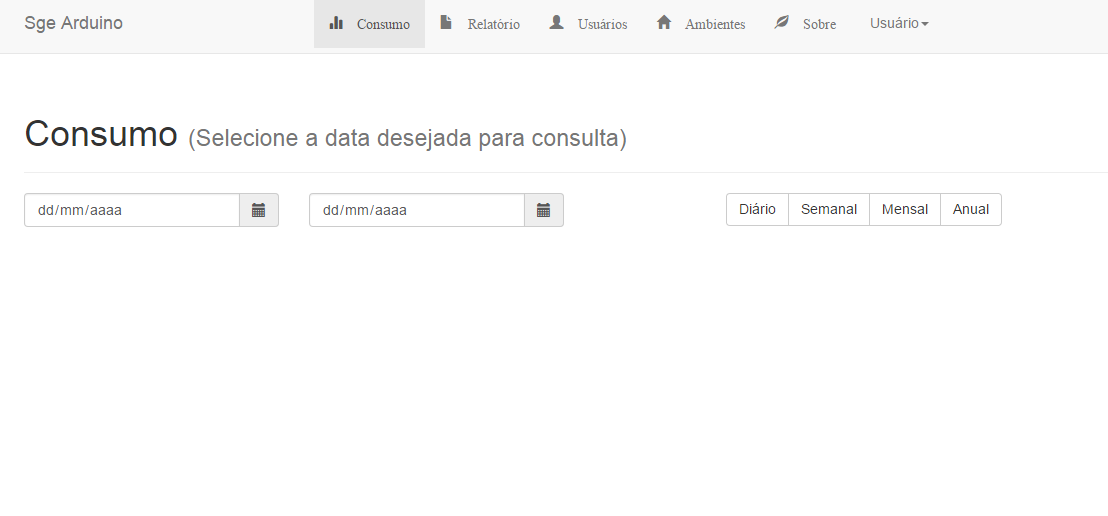


Fonte: Autores, 2017

A tela Consumo, mostrada na figura 19, é a tela exibida logo após o login do usuário ao sistema. A tela representa o consumo da casa, na data selecionada pelo usuário, representado através de um gráfico. O usuário pode selecionar a data especifica para a consulta ou também pode escolher entre: consumo diário, semanal, mensal ou anual. Exemplo: semanal é realizado uma consulta dos últimos sete dias. Cada botão (diário, semanal, mensal e anual) representa uma função do *back-end* que retorna os valores necessários para gerar o gráfico. Quando selecionado uma data especifica, é acionada uma função que recebe as duas datas, e retorna os dados da data requisitada para gerar o gráfico.

Por padrão o consumo selecionado é o mensal, e neste modo de consulta é computado o consumo de todos os ambientes cadastrados na casa.

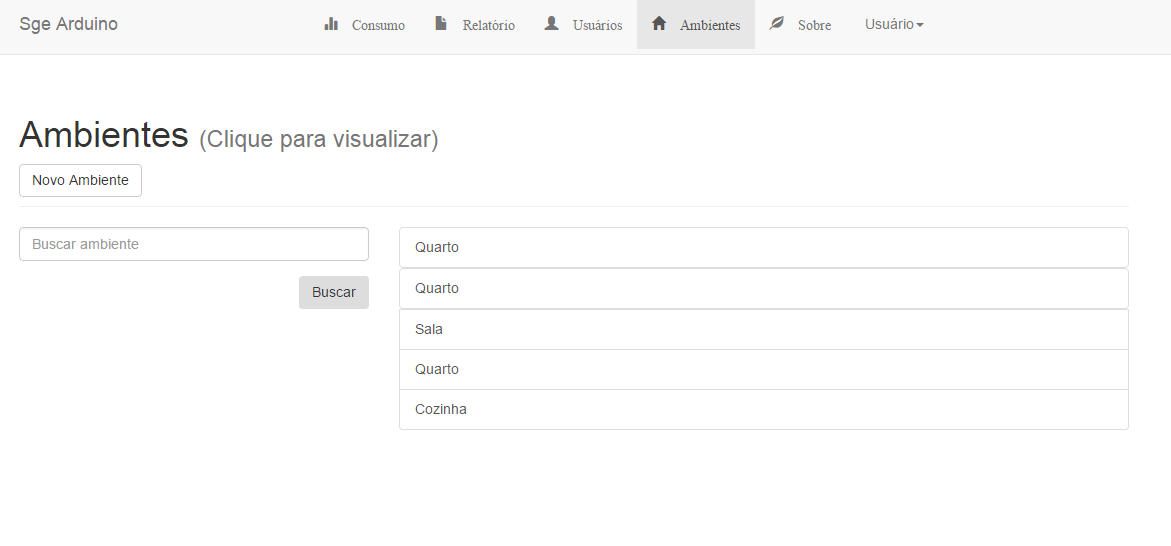
Figura : Tela Consumo



Fonte: Autores, 2017

A tela Ambientes, mostrada na figura 20, é a tela onde o usuário pode pesquisar e consultar os ambientes cadastrados na casa. Há também um botão (Novo Ambiente), que redireciona o usuário para o formulário de cadastro de um novo ambiente na casa, visível apenas para usuários proprietários. Quando o botão de busca é acionado, é enviado o campo de texto para a função de busca que retorna os resultados combináveis com o que está sendo requisitado.

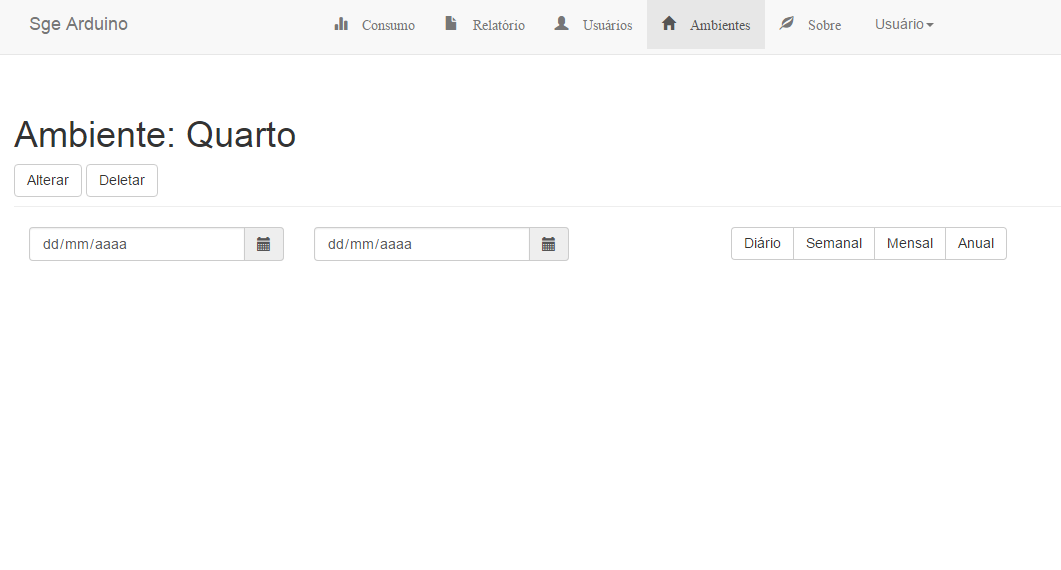
Figura : Tela Ambientes



Fonte: Autores, 2017

Selecionando um ambiente em especifico, o usuário é redirecionado para uma tela de consulta de consumo daquele ambiente, mostrada na figura 21, a tela possui as mesmas características da tela de consumo. O consumo é representado em gráfico, e o usuário pode selecionar a data especifica para a consulta ou também pode escolher entre: consumo diário, semanal, mensal ou anual. Exemplo: diário é realizado uma consulta das últimas vinte e quatro horas. Cada botão (diário, semanal, mensal e anual) representa uma função do *back-end* que retorna os valores necessários para gerar o gráfico. Quando selecionado uma data especifica, é acionada uma função que recebe as duas datas, e retorna os dados da data requisitados para gerar o gráfico. Por padrão o consumo selecionado é o mensal, e neste modo de consulta é computado o consumo apenas do ambiente selecionado. Há também dois botões: Alterar (alterar nome do ambiente), que pega o id do ambiente e redireciona para a página de alteração de nome do ambiente, e Deletar (deletar ambiente), que pega o id e deleta o ambiente correspondente através da função de deletar ambiente.

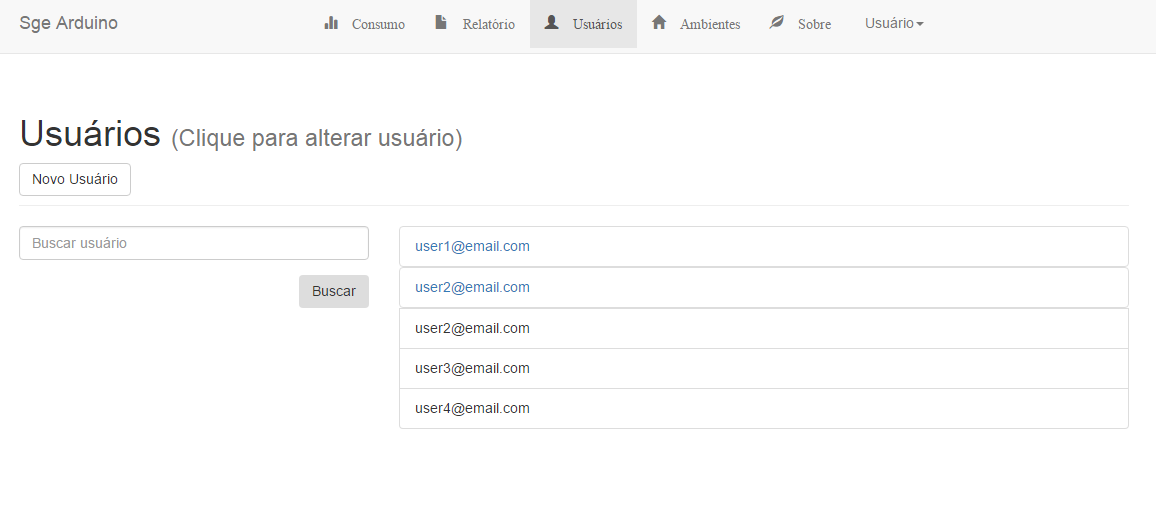
Figura : Tela Visualização do Ambiente



Fonte: Autores, 2017

A tela Usuários, mostrado na figura 22, é a tela onde o proprietário pode cadastrar um autorizado ao seu sistema, para realizar consultas do consumo da sua casa. Semelhante a tela Ambientes, o usuário proprietário pode pesquisar e consultar os usuários autorizados cadastrados em sua casa. Há também um botão (Novo Usuário) que redireciona o usuário proprietário para o formulário de cadastro de um novo usuário autorizado. Esta tela é visível apenas para usuários proprietários. Quando o botão de busca é acionado, é enviado o campo de texto para uma função de busca no back-end que retorna os resultados combináveis com o que está sendo requisitado.

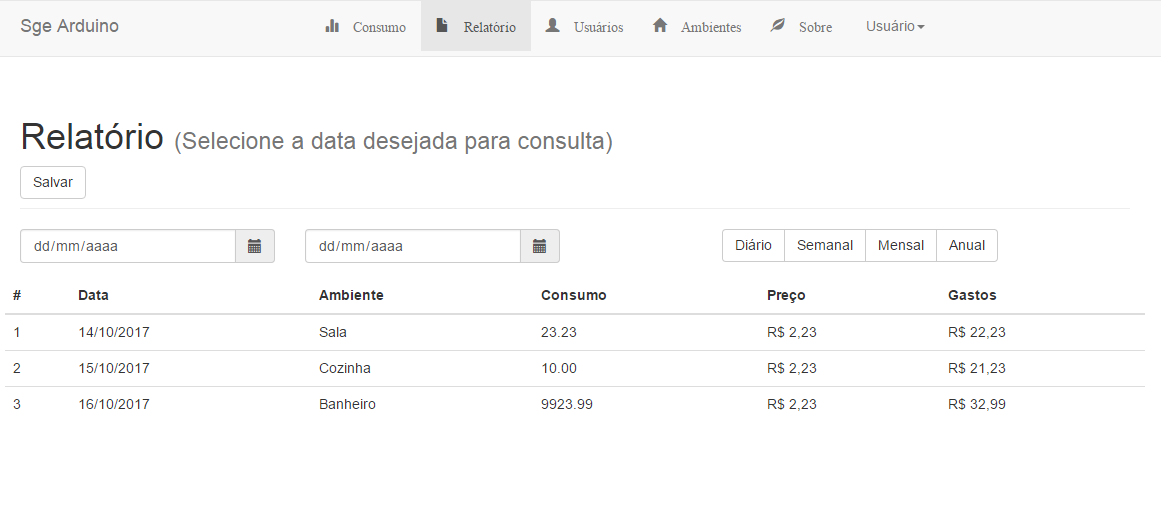
Figura : Tela Usuários



Fonte: Autores, 2017

A tela Relatório, mostrada na figura 23, é a tela onde o usuário pode gerar um relatório de consumo. Semelhante a tela de consumo, porém, ao invés de gerar um gráfico, o relatório é gerado em forma de uma tabela. O usuário pode selecionar uma data especifica para a consulta ou também pode escolher entre: consumo diário, semanal, mensal ou anual. Exemplo: mensal é realizado uma consulta dos últimos trinta dias. Cada botão (diário, semanal, mensal e anual) representa uma função do *back-end* que retorna os valores necessários para gerar a tabela. Quando selecionado uma data especifica, é acionada uma função que recebe as duas datas, e retorna os dados da data requisitados para gerar a tabela. Por padrão o consumo selecionado é o mensal, e neste modo de consulta é computado todos os ambientes cadastrados na casa. Há também um botão (Salvar) onde o usuário pode salvar em seu computador ou dispositivo móvel, o relatório no formato .xls, comando feito através da função salvar que realiza todas as conversões.

Figura : Tela Relatório



Fonte: Autores, 2017

# 4 CONSIDERAÇÕES FINAIS

No inicio da elaboração do trabalho, a ideia era construir um sistema de gerenciamento de energia elétrica tão eficaz quando o fornecido pelas companhias de distribuição de energia elétrica, porém, visto que o projeto também carregava a ideia de ser de baixo custo, o objetivo foi mudado, e se transformou em um sistema de gerenciamento para os usuários terem uma noção do consumo e gastos estimados em sua casa para a redução de desperdícios e redução de gastos.

O projeto pode ser mais explorado posteriormente. O essencial seria implementar uma melhoria no algoritmo do Arduino e tornar a calibração do sensor não-invasivo e da fonte AC-AC mais precisa. Se tratando de novos componentes, o projeto utiliza o ethernet shield para a conexão com a internet, e isso faz com que o Arduino esteja limitado ao cabo de rede, uma alternativa seria colocar um sensor Wi-Fi, que traria mais liberdade para a instalação do Arduino, porém é preciso ter em mente que quanto mais dispositivos tiver, mais caro o projeto fica. Pode-se também realizar alterações e melhorias na interface do usuário, que pode possuir gráficos mais elaborados e com novas funções, como por exemplo: gráfico com a quantidade de ambientes escolhida pela usuário, que atualmente, só é possível realizar a consulta de todos ou de apenas um.

Um ponto que precisa ser mencionado, é que a escolha do Arduino ser instalado em um ambiente foi devido ao fato de que se fosse ligado em uma casa inteira ficaria de certa forma, um sistema limitado, por outro lado, se fosse ligado a cada dispositivo, teria que ter muitos Arduinos e sensores para a captação, tornando-se inviável para a proposta do projeto. Por essa razão foi escolhido a utilização de um cômodo da casa (ambiente). O problema da instalação em um ambiente são as péssimas condições das estruturas presente nas casas brasileiras. Outro fator é o sensor-não invasivo, que se limita apenas para correntes continuas, e sistemas monofásicos para outros tipos de instalações seria necessárias adaptações.

Vale frisar que o projeto é um protótipo e não é a melhor tecnologia para o monitoramento de energia. Embora o consumo gerado pelo sistema não seja de alta precisão, é preciso levar em consideração que é um projeto de fácil instalação e de baixo custo, que proporciona comodidade para monitoramento e noção estimada dos gastos, além de possuir diversas formas para futuras alterações citadas anteriormente. Nesse sentido o projeto torna-se viável, com funcionalidade moldadas de acordo com o uso e as preferencias do usuário, e com algumas mudanças é possível adaptá-lo ao local desejado.

# REFERÊNCIAS BIBLIOGRÁFICAS

CCEE. **Setor Elétrico***.* Disponível em <https://www.ccee.org.br/portal/faces/pages\_publico/onde-atuamos/setor\_eletrico?\_adf.ctrl-state=171oyuxqdo\_4&\_afrLoop=1811225644566915#%40%3F\_afrLoop%3D1811225644566915%26\_adf.ctrl-state%3Dows2597l0\_4>. Acesso em 10 de junho de 2016.

FOLHA DE S. PAULO. **Desperdício consome 10% da energia elétrica no país.** Disponível em:

<http://www1.folha.uol.com.br/mercado/2015/02/1586778-desperdicio-consome-10-da-energia-eletrica-no-pais-diz-associacao.shtml>. Acesso em: 23 set. 2016.

ABESCO. **ABESCO faz levantamento e mostra o tamanho do desperdício energético do país – Revista EXAME**. Disponível em:

<http://www.abesco.com.br/pt/novidade/quisque-placerat-auctor-consectetur/>. Acesso em: 23 de set. 2016.

ABESCO. **4 Dicas para economizar energia elétrica nas empresas**. Disponível em:

<http://www.abesco.com.br/pt/novidade/4-dicas-para-economizar-energia-nas-empresas/>. Acesso em: 23 de setembro 2016.

CAVALCANTE, K. G. **Energia elétrica**. Disponível em: <http://brasilescola.uol.com.br/fisica/energia-eletrica.htm>. Acesso em 29 de setembro de 2016.

ARDUINO. **Arduino Mega**. Disponível em: <https://www.arduino.cc/en/Main/arduinoBoardMega>. Acesso em 25 de setembro de 2016.

OPENENERGYMONITOR. **Emon**. Disponível em: <<https://openenergymonitor.org/emon/>>. Acesso em 20 de setembro de 2016.

OPENENERGYMONITOR. **Main page**. Disponível em: <<https://wiki.openenergymonitor.org/index.php/Main_Page>>. Acesso em 19 de setembro de 2017.

OPENENEGYMONITOR. **Understanding the Zero Carbon Energy System**. Disponível em: <<https://learn.openenergymonitor.org/sustainable-energy/energy/introduction>>. Acesso em 20 de maio de 2017.

OPENENERGYMONITOR. **How to build an Arduino energy monitor**. Disponível em: https://learn.openenergymonitor.org/electricity-monitoring/ctac/how-to-build-an-arduino-energy-monitor. Acesso em 20 de maio de 2017.

OPENENERGYMONITOR. **Calibration**. Disponível em: <https://learn.openenergymonitor.org/electricity-monitoring/ctac/calibration>. Acesso em 20 de setembro de 2016.

OPENENERGYMONITOR. **Measuring voltage with an AC AC power adapter**. Disponível em: <https://learn.openenergymonitor.org/electricity-monitoring/voltage-sensing/measuring-voltage-with-an-acac-power-adapter>. Acesso em 20 de setembro de 2016.

ARDUINO. **Software**. Disponível em: <<https://www.arduino.cc/en/Main/Software>>. Acesso em 25 de setembro de 2016.

ARDUINO. **Arduino ethernet Shield**. Disponivel em: <https://www.arduino.cc/en/Main/ArduinoEthernetShieldV1 >. Acesso em 15 de setembro de 2016.

ARDUINO. **Ethernet**. Disponível em: https://www.arduino.cc/en/Reference/Ethernet. Acesso em 15 de setembro de 2016.

ARDUINO. **Enviroinment**. Disponível em: <<https://www.arduino.cc/en/Guide/Environment>>. Acesso em 15 de setembro de 2016.

ARDUINO. Arduino **Ethernet Shield**. Dísponivel em: <http://www.mouser.com/catalog/specsheets/A000056\_DATASHEET.pdf>. Acesso em 15 de maio de 2017.

PostgreSQL. **PostgreSQL 9.5.7 Documentation**. Disponível em: <https://www.postgresql.org/files/documentation/pdf/9.5/postgresql-9.5-A4.pdf>. Acesso em 20 de maio de 2017

ARGOUML. **ArgoUML User Manual.** Disponível em: <https://people.cs.pitt.edu/~chang/1635/ArgoUMLman.pdf>. Acesso em 19 de maio de 2017.

HACKAGE. **The shakespeare package**. Disponível em: <https://hackage.haskell.org/package/shakespeare>. Acesso em 15 de maio de 2017.

COMISSÃO TRIPARTITE PERMANENTE DE NEGOCIAÇÃO DO SETOR ELÉTRICO NO ESTADO DE SP. **Manual de treinamento curso básico: - segurança em instalações e serviços com eletricidade - NR 10**. 1. Ed. Rio de Janeiro: FUNDAÇÃO COGE, 2005.

CIMO, F. **Bootstrap Programming Cookbook.** 1. Ed. Grécia: Exelixis Media, 2015. [Consulta em 15 de maio de 2017] Disponível em:

<https://www.webcodegeeks.com/wp-content/uploads/2015/12/Bootstrap-Programming-Cookbook.pdf>. 2015

CIMO, F. **Javascript interview questions and answers**. 1. Ed. Grécia: Exelixis Media, 2016. [Consulta em 15 de maio de 2017] Disponível em:

<https://www.webcodegeeks.com/wp-content/uploads/2016/01/JavaScript-Interview-Questions.pdf>. 2016

MONK, S. **Fritzing for inventors**. 1. Ed. Estados Unidos: McGraw-Hill Education, 2015.

ESPAKE, P. **Learning Heroku Postgres**. 1. Ed. Reino Unido: Packt Publishing Ltd, 2015.

MELONI, J. C. **Sams Teach Yourself HTML, CSS and JavaScript All in One**. 2. Ed. United States of America: Pearson Education, 2015.

SILVA, A; VIDEIRA, C. **UML Metodologias e ferramentas CASE**. 1. Ed. Portugal: Centro Atlântico, Ltda, 2001.

SNOYMAN, M. **Haskell and Yesod**. 1. Ed. United States of America: O’Reilly Media, 2012.

Evans, M; Noble, J; Hochenbaum, J. **Arduino in action**. 1. Ed. United States of America: Manning Publication, 2013.

O'SULLIVAN, B; STEWART, D; GOERZEN, J. **Real World Haskell**. 1. Ed. United States of America: O’Reilly Media, 2008.

RICHARDS, R. **Pro PHP XML and Web Services.** 1. Ed. United States of America: Apress, 2006.

LIPOVAČA, M. **Learn You a Haskell for Great Good!.** 1 Ed. San Francisco, California: No Starch Press, 2011.

PRODANOV, C. C; FREITAS, E. C. **Metodologia do trabalho científico: Métodos e Técnicas da Pesquisa e do Trabalho Acadêmico**. 2. ed. Rio Grande do Sul: Feevale, 2013.

TANEMBAUM, A. S. **Organização Estruturada de Computadores**. 5. Ed. São Paulo: Prentice Hall, 2006.

PRESSMAN, R. S. **Engenharia de software**. 1. Ed. São Paulo: Makron Books, 1995.

HEUSER. C. A. **Projeto de banco de dados**. 4. Ed. Editora Sagra, 1998.