UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE MELHORIAS EM SISTEMAS DE ANÁLISE DE PH E TEMPERATURA DE ALIMENTOS

IAN CARLOS CRUZ

DIAMANTINA
2016

UNIVERSIDADE FEDERAL DOS VALES DO JEQUITINHONHA E MUCURI
DEPARTAMENTO DE COMPUTAÇÃO
BACHARELADO EM SISTEMAS DE INFORMAÇÃO

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE MELHORIAS EM SISTEMAS DE ANÁLISE DE PH E TEMPERATURA DE ALIMENTOS

Ian Carlos Cruz

Monografia submetida à Banca Examinadora designada pelo curso de Sistemas de Informação da Universidade Federal dos Vales do Jequitinhonha e Mucuri como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Orientador: Prof. Dr. Alexandre Ramos Fonseca
Co-orientador: Prof. Dr. Euler Guimarães Horta

DIAMANTINA
2016

Ian Carlos Cruz

DESENVOLVIMENTO E IMPLEMENTAÇÃO DE MELHORIAS EM SISTEMAS DE ANÁLISE DE PH E TEMPERATURA DE ALIMENTOS

Monografia submetida à Banca Examinadora designada pelo curso de Sistemas de Informação da Universidade Federal dos Vales do Jequitinhonha e Mucuri como requisito parcial para obtenção do título de Bacharel em Sistemas de Informação.

Prof. Dr. Alexandre Ramos Fonseca (Orientador)
Universidade Federal dos Vales do Jequitinhonha e Mucuri

Prof. Dr. Euler Guimarães Horta (Co-orientador)
Universidade Federal dos Vales do Jequitinhonha e Mucuri

Prof. Dr. Marcus Vinícius Carvalho Guelpeli
Universidade Federal dos Vales do Jequitinhonha e Mucuri

Diamantina, 04 de Março de 2016

Agradecimentos

Primeiramente agradeço aos meus pais, Antônio e Iara pela dedicação e confiança. Por terem formado a base do meu caráter, ensinando tudo que considero de mais valia.

À minha irmã Patrícia e à minha tia Nadir pelo apoio durante essa trajetória.

Agradeço especialmente à minha namorada Camile. Sua presença e amor constantes me deram forças e me incentivaram a dar o melhor de mim tanto em momentos difíceis quanto em períodos de conforto. Te amo muito!

Aos meus amigos pelos momentos de festa, descontração e experiências vividas e compartilhadas. São grandes parceiros que levarei para o resto da vida. Agradeço ainda ao Otávio pelo aconselhamento técnico.

Aos professores do Curso de Sistemas de Informação, principalmente ao Prof. Alexandre e ao Prof. Euler pelas experiências e ensinamentos compartilhados. Muito obrigado!

Enfim, agradeço a todos que de alguma forma contribuíram para que essa jornada fosse finalizada com sucesso.

Resumo

Tanto na produção quanto em pesquisas na área alimentar, há a necessidade de se monitorar a qualidade dos alimentos para que estes não sejam maléficos à saúde de seus consumidores. Nesse âmbito, o potencial hidrogeniônico (pH) e a temperatura são dois dos principais fatores que auxiliam na manutenção da qualidade alimentícia, já que o descontrole de seus níveis pode acarretar o crescimento indesejável de bactérias no produto, gerando riscos à saúde. Dessa forma, para auxiliar no controle desses níveis, foi desenvolvido por um ex-aluno da Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM) o SiCarne, software que monitora os níveis de pH e temperatura de um produto coletando dados provenientes de um pHmetro conectado à uma porta serial RS232C. No entanto, o SiCarne é limitado pelo fato de as análises feitas só poderem ser obtidas no local onde o experimento está sendo feito. Como solução a esse problema, foi criado o pHremoto, sistema que coleta os dados gerados pelo SiCarne e os disponibiliza através de uma interface *Web* responsiva fazendo com que sua visualização seja possível através de qualquer dispositivo com acesso à *Internet*. O presente trabalho estende as funcionalidades do SiCarne e do pHremoto tornando-os mais completos, ampliando assim as possíveis situações de utilização de ambos os sistemas e, como consequência, melhorando a experiência de seus usuários. Uma nova tela para Cadastro de Novos Alimentos foi desenvolvida no SiCarne bem como foram implementadas melhorias para torná-lo escalável para os alimentos recém-inseridos. Com relação ao pHremoto, uma nova funcionalidade de geração de relatório foi implementada, otimizando a interpretação das análises. Para o desenvolvimento no SiCarne foi utilizada a *Swing*, *API* do *Java* que fornece elementos para a montagem de interfaces gráficas por *drag and drop*. Essa abordagem diminuiu o tempo gasto nas implementações de interface gráfica, o que possibilitou maior dedicação à verificação do correto funcionamento do que foi desenvolvido. No pHremoto utilizou-se a arquitetura *MVC* do *Java*, baseada em *Servlets* e *JSPs*. A geração do relatório foi feita através da inclusão de seus dados num *layout* codificado em *HTML*, que é então convertido para *PDF* através do *iText*.

Palavras-chave: SiCarne. pHremoto. Desenvolvimento. Tela de cadastro. Escalabilidade de sistemas. Geração de relatório.

Abstract

Both in the production and researches conducted in the food field, there's the need to monitor the food's quality in order to make sure they are safe for human consumption. As such, the potential of hydrogen (pH) and the temperature of the food are two among the main factors that help to monitor food quality, given that their lack of control may propritiate the unwanted growth of bacteria bringing risks to the consumer's health. In order to help maintaining that control, an alumnus of the Universidade Federal dos Vales do Jequitinhonha e Mucuri (UFVJM) developed a software called SiCarne which monitors a food product's pH and temperature coming from a pHmeter connected to a RS232C serial port. However, the analyses performed by SiCarne can only be obtained by going to the experiment's physical location. As a solution to this problem the pHremoto was developed, software that collects the data generated by SiCarne and makes them available through a responsive Web interface, making possible their visualization in any device that has access to the Internet. This paper's extends both the SiCarne and pHremoto features, thus making them more complete and increasing the amount of possible situations in which they could be used improving their users's experience. A new window for registering new types of food was developed for SiCarne as well as a set of improvements to make the system scalable for the newly inserted foods. Regarding pHremoto, a brand new report generation feature was implemented optimizing analyses's interpretation. The Java API Swing was used in SiCarne's development. It allows the design of graphic interfaces through drag an drop. This approach lowered the time spent on interfaces implementations allowing greater dedication to verifying the correct operation of what was developed. For the pHremoto the MVC architecture from Java, based on Servlets and JSPs, was used. The report generation was made including its data in a layout coded in HTML, which is then converted to PDF by iText.

Keywords: SiCarne. pHremoto. Development. Register window. Systems scalability. Report generation.

Sumário

Lista de Figuras

Lista de Siglas

API	*Application Programming Interface*
CRUD	*Create, Read, Update, Delete*
CSS	*Cascading Style Sheets*
CSV	*Comma-separated values*
DDR3	*Double data rate type three*
GHz	*Gigahertz*
HTML	*HyperText Markup Language*
IP	*Internet Protocol*
JAR	*Java ARchive*
JDK	*Java Development Kit*
JNI	*Java Native Interface*
JPEG	*Joint Photographic Experts Group*
JSP	*JavaServer Pages*
JVM	*Java Virtual Machine*
MVC	*Model View Controller*
PDF	*Portable Document Format*
pH	*Potencial Hidrogeniônico*
PNG	*Portable Network Graphics*
RAM	*Random Access Memory*

Capítulo 1

Introdução

Na área de estudos alimentícios, a segurança do consumo de determinado alimento pode ser avaliada, dentre outros fatores, pela medição de determinadas características intrínsecas e extrínsecas ao alimento que certificam sua estabilidade microbiológica, ou seja, que a multiplicação de microrganismos não atinja níveis considerados infecciosos. Dentre as características utilizadas como mensuração de qualidade do alimento, destacam-se o potencial hidrogeniônico (pH) como característica intrínseca e a temperatura como característica extrínseca (1).

O pH de um alimento é determinado a partir da mensuração de sua acidez e alcalinidade . A escala do pH é dada por um intervalo que vai de 0 a 14, sendo 0 considerado extremamente ácido, 7 neutro e 14 extremamente alcalino. Com isso, os alimentos podem ser classificados em alimentos que possuem baixa acidez ou alcalinos (pH maior do que 7), alimentos neutros (pH igual a 7) e alimentos ácidos (pH menor do que 7) (2).

Seres humanos, através do paladar, conseguem detectar diferenças significativas de pH em um alimento. Alimentos ácidos como laranja, limão e vinagre possuem sabor azedo, já alimentos alcalinos apresentam sabor amargo, como clara de ovo e bicarbonato de sódio. Por sua vez, a temperatura pode ser detectada pela sensação térmica de uma matéria em relação ao tato.

Dessa forma, para garantir a estabilidade microbiológica do alimento e, como conseguinte, sua qualidade para consumo humano, são utilizadas técnicas e métodos para monitorar e controlar o pH e a temperatura. Essas técnicas levam em consideração o princípio de que o pH e a temperatura do alimento inibirão o crescimento microbiano (3).

Com relação ao pH, os métodos mais comuns para seu monitoramento são a comparação de cor do alimento e a medição elétrica. Já a temperatura pode ser medida por um termopar conectado diretamente ao pHmetro ou por um termômetro independente em conjunto com o eletrodo (4).

Através da comparação de cor é possível, por exemplo, determinar o estado de uma peça de carne. Uma carne de coloração escura apresenta pH inadequado com efeito na qualidade e na vida útil do produto (5). O nível inadequado de pH na carne pode ser causado, por exemplo, no momento do abate do animal. Caso ele se encontre em estado de estresse em tal momento, sua temperatura corporal aumenta e a glicólise acontece mais rapidamente, causando, assim, a queda do pH, a desnaturação proteica e uma mudança na química dos músculos. Como consequência, ocorre o endurecimento rápido dos membros do cadáver, ou seja, estabelecimento do *rigor mortis*. A combinação desses acontecimentos altera a conversão normal do músculo em carne, deixando-a mais dura e escura (5).

Para a realização da medição elétrica, faz-se necessário o uso de um pHmetro, "aparelho que consiste de um eletrodo de referência, um eletrodo indicador e um dispositivo eletrônico de baixa impedância para medir o potencial entre os eletrodos" (4). Esse aparelho permite converter o valor de potencial do eletrodo em unidades de pH (6).

O pHmetro permite que a temperatura e os valores instantâneos de pH do objeto de teste sejam visualizados através de um visor, necessitando assim que o experimento seja acompanhado de perto por seu condutor. Contudo, alguns modelos de pHmetro possuem uma porta serial RS232C permitindo sua comunicação com computadores que forneçam uma porta RS232C para ligação de dispositivos externos (4).

Com isso, foi desenvolvido na Universidade Federal dos Vales do Jequitinhonha e Mucuri como trabalho de conclusão do curso de Sistemas de Informação do aluno Gustavo Marques Nascimento um sistema denominado SiCarne (7). O SiCarne é um sistema desenvolvido em *Java* que lê os dados enviados por uma porta RS232C conectada a um pHmetro e os interpreta, automatizando as análises de pH e temperatura.

Na sua concepção, o objetivo do SiCarne era tornar

> (...) possível a coleta e análise dos dados provenientes do pHmetro totalmente automatizada, ou seja, totalmente realizada pela aplicação, permitindo assim que os usuários concentrem seus esforços nas análises realizadas sobre as informações geradas pela aplicação. Ela se encarrega de coletar os dados (pH e temperatura) durante toda a análise, bastando ao usuário somente definir a duração total, os intervalos de tempos desejados e o tipo de carne a ser analisada. (7).

Considerando o acima exposto, a implementação original do SiCarne foi feita tomando como premissa que o software seria utilizado apenas na análise de pH e temperatura de carnes de variados tipos. Como consequência, o SiCarne possuía algumas limitações. Não era possível, por exemplo, realizar mais de uma análise em paralelo, persistir os dados da análise em banco de dados para consulta e visualização posterior ou até mesmo monitorar a análise remotamente. Dessa forma, o único modo de se utilizar o SiCarne era estando no local onde o experimento estivesse sendo realizado, monitorando-o ativamente. Ao final do experimento, o SiCarne gerava um arquivo em formato CSV contendo os dados de pH e temperatura medidos. Porém, caso o experimento fosse interrompido, esse arquivo não era gerado e todos os dados obtidos até então eram perdidos.

Em face desses problemas, o aluno Douglas Souza Lima implementou, como trabalho de conclusão do curso de Sistemas de Informação da Universidade Federal dos Vales do Jequitinhonha e Mucuri, um sistema denominado pHremoto e algumas alterações no SiCarne com a finalidade de adaptá-lo ao pHremoto (4).

O pHremoto é um sistema que provê uma interface *Web* responsiva para acompanhar e visualizar remotamente os dados provenientes de experimentos automatizados pelo SiCarne. Atualmente, o pHremoto disponibiliza um histórico das análises feitas pelo SiCarne além de apresentar gráficos que representam os resultados dos experimentos. Os gráficos gerados pelo pHremoto apresentam os dados do pH e da temperatura do produto, ambos representados através de suas variações através do tempo do experimento. Por possuir uma interface desenvolvida com base nos princípios de responsividade *Web*, o pHremoto pode ser utilizado em

qualquer dispositivo com acesso à *Internet* independentemente do tamanho e da resolução de sua tela.

Para que o pHremoto pudesse obter as análises feitas pelo SiCarne, foi necessário a modificação deste para que os dados das análises pudessem ser persistidos para posterior utilização. Dessa forma, foi criado um banco de dados para possibilitar a gravação dos dados oriundos do SiCarne e a leitura pelo pHremoto, além de feitas as devidas alterações no SiCarne para adaptá-lo ao recém-criado banco de dados.

Contudo, as modificações feitas por Lima (4) no SiCarne não introduzem novas funcionalidades para os usuários. O SiCarne continuou limitado a analisar carnes e gerar gráficos com base nos dados da análise. O pHremoto representou um grande avanço em relação ao SiCarne possibilitando que o acompanhamento dos experimentos e a visualização dos gráficos gerados fossem feitos remotamente e por meio de qualquer dispositivo que dispõe de acesso à internet. Com relação a requisitos funcionais, o pHremoto introduziu a visualização do histórico de análises, mantendo as mesmas funcionalidades trazidas pelo SiCarne.

Dessa forma, foram identificadas deficiências em ambos os sistemas que, quando supridas, ampliarão drasticamente suas possibilidades de utilização. Isso fará com que a experiência de seus usuários seja mais completa e que o universo de emprego dos sistemas aumente.

Com relação ao SiCarne, foi decidido que será implementada uma nova funcionalidade que possibilitará o cadastro de novos alimentos. Essa funcionalidade aumentará a quantidade de possíveis cenários de uso do SiCarne incluindo a realização de análises de pH e temperatura com outros alimentos. Para que o cadastro de novos alimentos seja possível, serão feitas algumas adaptações ao sistema para que se torne escalável.

Tendo em vista o pHremoto, identificou-se que ele poderia ser ampliado para fornecer todas as informações possíveis a respeito de um determinado experimento de forma compactada e sucinta. Essa abordagem deixaria a análise do experimento mais eficiente economizando o tempo que o pesquisador dispenderia somente para reunir os dados. Para que isso seja possível, será implementada a geração de um relatório completo contendo os dados referentes ao usuário responsável pelo experimento, os dados do alimento objeto do experimento, as informações da análise e os gráficos gerados.

1.1 Objetivos

O objetivo desse trabalho é implementar funcionalidades para dois sistemas criados por alunos da Universidade Federal dos Vales do Jequitinhonha e Mucuri: o SiCarne e o pHremoto.

No SiCarne será implementada uma tela de cadastro de alimentos visando ampliar o universo de objetos de estudo que podem ter suas análises de pH e temperatura automatizados pelo sistema. Com isso, alguns elementos do sistema serão adaptados para receber a alteração, como a caixa de seleção do tipo de carne que poderá selecionar qualquer alimento que for cadastrado no banco de dados, o painel de configurações avançadas que comportará o botão com a opção de cadastro de novo alimento e o método que monta a conclusão do experimento que deixará de exibir textos fixos para experimentos com carnes e será dinâmico adaptando-se a qualquer alimento utilizado .

No pHremoto será implementado um relatório com todas as informações possíveis a respeito da análise realizada.

1.2 Objetivos específicos

- Construir uma tela de cadastro de alimentos no SiCarne;
- Adaptar o SiCarne para comportar a funcionalidade de Cadastro de Alimentos:
  - Alterar a caixa de seleção do tipo de carne;
  - Alterar o painel de Configurações Avançadas adicionando o botão que mostrará a nova tela de cadastro de alimentos;
  - Alterar a conclusão do experimento.
- Implementar a geração de um relatório com os dados da análise no pHremoto;

1.3 Metodologia

Para o desenvolvimento desse trabalho, mostrou-se fundamental o estudo de ambos os sistemas, SiCarne e pHremoto. Cada sistema foi analisado de forma a entender a estrutura e estratégia de implementação usadas para fazer com que as funcionalidades propostas fossem desenvolvidas. Conhecer os sistemas foi indispensável para manter os padrões utilizados pelos

desenvolvedores que já passaram pelos projetos, assim como para respeitar as arquiteturas utilizadas.

Também mostrou-se essencial a realização de pesquisas e leituras de artigos a respeito da área de conhecimento abordada pelos sistemas. O aprofundamento nos temas abordados forneceu base suficiente para que as implementações fossem feitas visando melhorar a experiência de seus usuários e respeitar o escopo do trabalho.

A implementação das alterações no SiCarne requereu o aprendizado da linguagem *Java* juntamente com a tecnologia *Swing*. Primeiramente, foram desenvolvidas todas as demandas de interface gráfica utilizando o *Swing* para criar telas e modificar elementos de interface. Após montadas as interfaces gráficas, o *Java* foi utilizado para codificar os eventos disparados pela interface. As alterações no SiCarne envolveram também a interação entre o sistema e o banco de dados.

Com relação ao pHremoto, foi necessário entender como o padrão *MVC* é implementado na linguagem *Java*. Esse aprendizado trouxe o aprofundamento a respeito dos *Servlets* e das *JSP*, elementos do *Java* que representam os *Controllers* e as *Views*, respectivamente, e que foram utilizados para codificar o sistema *Web*. Para a implementação do relatório foi utilizado o *iText*, *API* capaz de gerar arquivos *PDF* a partir de uma cadeia de caracteres contendo uma página *HTML*. Dessa forma, o *layout* do relatório foi codificado em *HTML* dentro de uma *string* que foi então passada para o *iText* realizar a conversão.

1.4 Estrutura do texto

O presente trabalho estrutura-se em cinco capítulos que descrevem todas as etapas necessárias para que o escopo do estudo fosse atingido de forma a manter uma estrutura lógica e coerente.

O Capítulo 1 - apresenta a base teórica do campo de estudo relativo aos sistemas utilizados, bem como descrições sucintas de cada um. Logo após, são descritos os objetivos gerais e específicos do trabalho e a metodologia utilizada em seu desenvolvimento.

O Capítulo 2 - descreverá os conceitos, métodos e tecnologias utilizados. Serão apresentadas breves descrições sobre o SiCarne e o pHremoto. Logo após, serão descritos alguns conceitos e tecnologias utilizados no desenvolvimento dos sistemas, tais com *Swing*, *MVC* e *CRUD*.

O Capítulo 3 - mostrará como as novas funcionalidades propostas pelo trabalho foram implementadas.

O Capítulo 4 - apresentará os testes realizados para verificar e validar as funcionalidades implementadas.

O Capítulo 5 - mostrará as conclusões obtidas com o desenvolvimento do trabalho e apresentará algumas propostas para trabalhos futuros.

Capítulo 2

Referencial Teórico

Nesse capítulo serão apresentados os conceitos, métodos e tecnologias utilizados na implementação das funcionalidades propostas no presente trabalho.

2.1 SiCarne

O SiCarne é um sistema desenvolvido na linguagem *Java* por Nascimento (7) como trabalho de conclusão de curso. Segundo os referidos autores:

> O desenvolvimento desta aplicação contribuirá com a qualidade das análises, uma vez que os resultados serão emitidos com mais precisão, maior rapidez e praticidade, gerando, assim, resultados mais eficientes. Além de permitir aos usuários que concentrem seus esforços nas análises dos resultados gerados e não na coleta dos dados que será realizada totalmente pela aplicação. (7).

Ou seja, o objetivo do SiCarne é fazer com que o trabalho dos usuários do sistema seja mais eficiente eliminando a preocupação com a coleta e manejo dos dados, focando, assim, na geração de conhecimento através das análises dos resultados.

Para seu correto funcionamento, foi necessário que o SiCarne estabelecesse uma conexão com o pHmetro utilizado no experimento. Para isso, foram utilizadas as conexões seriais RS232C, presentes tanto no pHmetro quanto na máquina hospedeira do SiCarne, e um cabo de conexão

modelo RS232C. Para que o *software* pudesse interpretar os dados enviados pelo pHmetro, utilizou-se a *RXTX*, uma *API* escrita em *Java* que usa implementação nativa (*JNI*) e provê interfaces de comunicação com portas seriais e paralelas para o *JDK* (8).

Para a geração de gráficos, o SiCarne utilizou a biblioteca *JFreeChart* que, segundo seus desenvolvedores, *"is a free* 100% *Java chart library that makes it easy for developers to display professional quality charts in their applications"* (9). O *JFreeChart* fornece métodos para criação de diversos tipos de gráficos, como barra, linha e pizza, juntamente com variadas opções de saída, como gravação em *JPEG*, *PNG*, impressora, entre outras.

2.2 pHremoto

O pHremoto foi desenvolvido pelo aluno Douglas Souza Lima em seu trabalho de conclusão de curso (4). De acordo com os referidos autores, tal trabalho propõe

> (...) o desenvolvimento de um sistema que possibilite a integração de uma aplicação web com o SiCarne para que o monitoramento da queda do pH da carne e de outros alimentos possam ser feitas de maneira remota. Com esta aplicação web utilizando a internet será possível acessar esse sistema de qualquer rede, a partir de qualquer desktop, smartphone ou tablet. (4)

Com isso, o pHremoto é um sistema *Web* que utiliza os dados provenientes do SiCarne, que por sua vez alimenta o banco de dados em comum. Ele utiliza a arquitetura *MVC*, descrita na Seção 2.4, e possui uma interface responsiva, tecnologia que adapta automaticamente um *web site* à resolução e ao tamanho da tela em que está sendo visualizado.

Seu desenvolvimento foi feito em variadas linguagens tendo como base o *Java*. O *back-end* do sistema foi feito em *Java* utilizando a tecnologia de *Servlets*. Já no *front-end* foi utilizada a tecnologia *JSP*, capaz de gerar páginas *Web* dinâmicas através da interação de páginas escritas em *HTML*, *CSS* e *JavaScript* com o motor do *Java*.

Dessa forma, a Figura 2.1 mostra como o pHremoto se relaciona com o SiCarne.

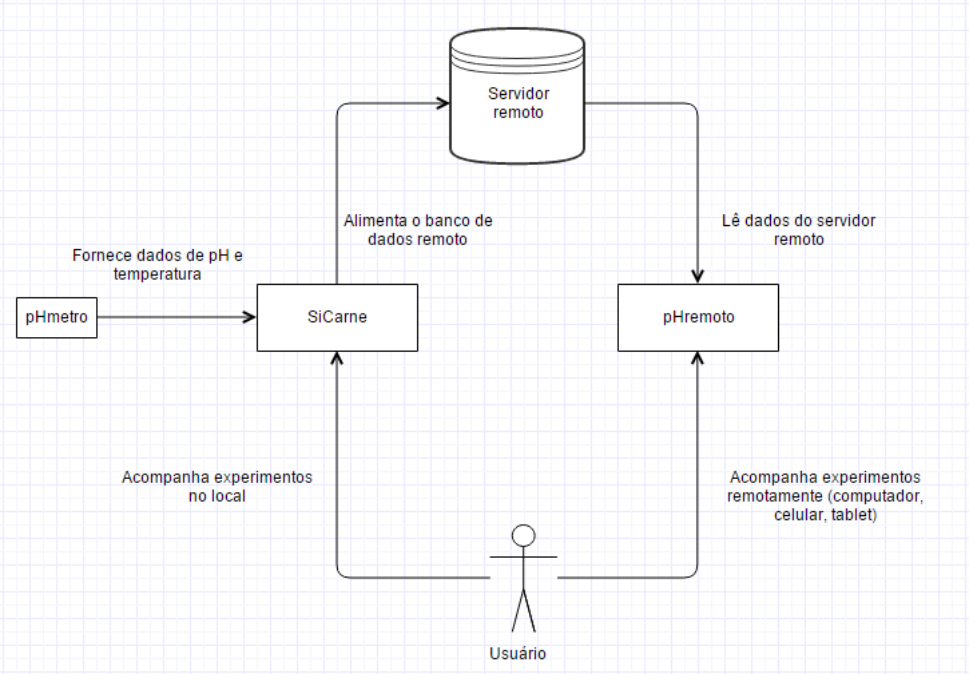

Figura 2.1: Diagrama do relacionamento entre o SiCarne e o pHremoto.

2.3 *Swing*

Um bom sistema de informação deve realizar aquilo que se propõe a fazer. Deve implementar as devidas funcionalidades de forma eficiente, utilizando da melhor forma possível os recursos que as tecnologias escolhidas para seu desenvolvimento disponibilizam. Além disso, deve ser eficaz, realmente se comportando da maneira prevista em seu planejamento.

Porém, um sistema não deve apenas implementar eficientemente e eficazmente as funcionalidades propostas. Para que ele realmente agregue valor ao seu campo de atuação, seus usuários devem ser capazes de utilizar a maior quantidade possível de funcionalidades dispendendo o menor esforço possível. Isso se torna concebível através de interfaces gráficas bem desenhadas.

O *Java*, linguagem utilizada no desenvolvimento do SiCarne, provê um conjunto de componentes que faz com que o desenvolvimento de interfaces gráficas se torne rápido e possibilita que estas sejam desenhadas da melhor forma possível para atender os requisitos de seus usuários. Esse conjunto de componentes chama-se *Swing*.

A *API* do *Swing* provê elementos básicos de uma interface gráfica como botões, caixas de textos, *labels*, caixas de seleção e os torna disponíveis ao programador através de uma aba de propriedades. Dessa forma, as interfaces são montadas por meio de *drag and drop*, técnica

que consiste em selecionar um componente e arrastá-lo até sua posição final. Com base nisso, é possível montar toda a tela utilizando recursos simples de fácil manuseio.

Após montada, uma interface feita com a *Swing* dá ao programador a opção de determinar quais eventos poderão ser disparados por dado componente. Após selecionar o evento, a *Swing* leva o desenvolvedor até o código responsável por gerenciá-lo. Uma vez dentro do método do evento, o programador consegue executar lógicas e regras de negócio, dando ao usuário o poder de modificar o sistema.

A *Swing* foi a tecnologia escolhida para desenvolver a interface gráfica do SiCarne e, por esse motivo, foi utilizada na implementação das telas e alterações de interface gráfica propostas por esse trabalho.

2.4 *MVC*

Sistemas de informação automatizados são estruturas complexas internamente formadas por um conjunto de partes menores que interagem entre si. Em sistemas maiores e mais complexos, existe a necessidade de fazer com que as pequenas partes estejam logicamente organizadas. A organização do sistema facilita sua manutenção, diminui a complexidade da adição de novas funcionalidades e aumenta seu fator de escalabilidade, visto que é mais facilmente entendido por novos desenvolvedores que porventura se juntem ao projeto.

Com isso, surgiu a necessidade da criação de arquiteturas de sistemas bem definidas e documentadas. Segundo Bass (10):

> (...) uma arquitetura de software para um sistema é a estrutura ou estruturas do sistema que envolvem seus elementos, os comportamentos externamente visíveis desses elementos e os relacionamentos entre eles. (10).

Previamente, em 1992, Steve Burbeck publicou um artigo denominado *Applications Programming in Smalltalk-80™: How to use Model-View-Controller (MVC)* (11). Nesse artigo, Burbeck (11) originalmente implementou o conceito de um novo padrão de design, descrito pela primeira vez por Trygve Reenskaug, que trazia uma nova arquitetura de sistemas composta por macroelementos, descrevendo-os e as relações entre eles (12).

Esse novo padrão de design foi chamado de *MVC*. De acordo com Burbeck (11):

> No paradigma MVC a entrada de usuário, a modelagem do mundo externo e o feedback visual para o usuário são explicitamente separados e controlados por três tipos de objetos, cada um especializado em realizar sua tarefa. (11).

Dessa forma, o *MVC* implementa três entidades responsáveis unicamente por realizar uma tarefa específica, sendo elas o *Model*, a *View* e o *Controller*.

O *Model* é o elemento responsável por modelar o mundo externo. Ele representa entidades externas por meio de dados e relacionamentos dentro do domínio da aplicação. Dentro do *MVC*, o *Model* se relaciona tanto com a *View* quanto com o *Controller*, respondendo a requisições daquela sobre seu estado atual e às instruções de mudança de estado desta.

A *View* é a apresentação do estado do *Model* para o mundo exterior, sendo responsável por interagir diretamente com o usuário. É através dela que instruções chegam ao sistema e que o resultado de seus comandos é exibido. A forma de *View* mais conhecida e utilizada atualmente é a página da *Web*. Através dela usuários conseguem enviar comandos para que o sistema faça algo e visualizar o que o sistema manda em retorno. Fechando o ciclo, o *Controller* é o objeto responsável por controlar a interação da *View* com o *Model*. As instruções enviadas pela *View* são recebidas pelo *Controller* que se encarrega de traduzi-las para a linguagem interna do sistema e determinar o que deve ser feito para atender a requisição externa. Dessa forma, o *Controller* é capaz de dizer ao *Model* para alterar seu estado em razão de atender a instrução externa. Com isso, a *View* é capaz de retornar ao ambiente externo o novo estado do sistema.

Analogamente ao modelo cliente-servidor, quando é utilizada a arquitetura *MVC*, as *Views* são consideradas elementos pertencentes ao lado do cliente enquanto os *Models* e *Controllers* são objetos do servidor.

As interações entre os elementos da arquitetura *MVC* é ilustrada na Figura 2.2.

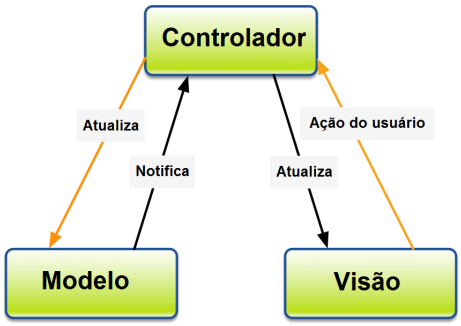

Figura 2.2: Relacionamento entre os elementos em uma arquitetura *MVC*. Adaptado de http://stackoverflow.com/questions/29594105/mvc-is-it-model-to-view-or-controller-to-view

2.4.1 Servlets

Na arquitetura *MVC*, o *Controller* é o elemento responsável por receber as requisições da *View* e alterar o estado do sistema de acordo com tais. Cada linguagem de programação e/ou plataforma de desenvolvimento implementa sua própria versão do objeto *Controller* de modo a torná-lo familiar para os outros elementos que as compõem.

Considerando que o pHremoto foi implementado utilizando a linguagem *Java*, foi necessário usar o objeto *Controller* dessa linguagem para possibilitar a construção em *MVC*: o *Servlet*.

O *Servlet* é uma classe *Java* responsável por responder requisições. Dessa forma, quando utilizada em uma aplicação *Web*, o *Servlet* estende as funcionalidades do servidor proporcionando-o a capacidade de receber requisições. Essas requisições são, em sua maioria, requisições *HTTP* provenientes de aplicações *Web*. Com isso, o *Servlet* é capaz de interpretá-las, respondendo tais requisições de volta à camada de apresentação da aplicação.

Um *Servlet* é capaz de implementar um sistema *Web* criando conteúdo dinamicamente para uma página da *Web*. Dessa forma, sistemas *Web* se diferenciam de páginas *Web* por conseguirem um nível maior de interação com usuário ao invés de simplesmente permitir a navegação por um conteúdo estático. (13)

A funcionalidade de permitir que o conteúdo criado dinamicamente seja enviado e posteriormente exibido por uma página *Web* é fundamental no *MVC*. Isso faz com a *View* esteja sempre atualizada em conformidade com o que foi alterado no sistema, ou seja, que o usuário esteja sempre interagindo com os dados do sistema, ou *Models*, em seu estado mais atual.

2.4.2 JSP

Analisando o sistema em função do usuário, nenhum outro elemento daquele possui mais importância do que a interface. Ela é a responsável pela interação entre o sistema e o usuário permitindo-o realizar as alterações desejadas.

No *MVC*, a interface do sistema é representada pela *View*. Com isso, as Views se encarregam de fornecer ao usuário as funcionalidades que o sistema oferece e de exibir o resultado que as instruções recebidas provocaram internamente. Na plataforma *Java*, as *Views* são representadas pelas *JSP*.

JSP é a sigla para *JavaServer Pages*. Uma *JSP* é uma página *Web* comum desenvolvida em *HTML*, *CSS* e *JavaScript* que consegue interpretar código *Java*. Isso faz com que seja possível capturar o estado atual do sistema em tempo de execução da página. Dessa forma, a camada de apresentação, composta pelas *Views*, consegue acompanhar as mudanças realizadas nos dados internos e exibi-las para o usuário.

2.4.3 Model

Como dito anteriormente, *Controllers* são os componentes responsáveis por receber e responder as requisições enviadas pela camada de apresentação e as *Views* são os elementos de interface que compõem tal camada. Dessa forma, há a necessidade de conectá-las. O componente responsável por realizar tal tarefa é o *Model*.

Um *Model* representa o encapsulamento dos dados que a *View* apresenta. Quando uma *View* manda uma requisição para o *Controller* realizar alguma alteração no sistema, após realizadas todas as operações internas, o *Controller* instancia o *Model* referente àquela *View* e o preenche com o estado atual do sistema. Após ser atualizado, o *Model* fica disponível para que a *View* o leia e apresente as alterações para o usuário.

No *Java*, quando um *Servlet* recebe uma requisição de uma *JSP*, ele realiza todas as alterações pedidas executando lógicas de negócio, chamando *Web Services* e realizando os acessos a banco. Depois de concluídas as operações, o *Servlet* instancia um *Model*, classe que representa os dados que a *JSP* deve exibir ao usuário. Após o *Model* ser instanciado, o *Servlet* o preenche com os dados que devem ser apresentados pela *JSP*, que então o acessa por meio do motor *Java* disponível dentro da página *HTML*. Com isso, o usuário é capaz de visualizar as alterações que foram previamente requisitadas.

2.5 *CRUD*

Todo sistema de informação necessita realizar acesso a dados para que estes sejam lidos, alterados por lógicas de negócio e então salvos com seu novo estado. Tal comportamento dinâmico com manipulação de dados é o que o diferencia de simples páginas *Web*. Existem variadas estratégias e métodos para obter os dados necessários, como utilização de *Web Services*, *upload* do sistema de arquivos, acesso a dados disponíveis na *Internet* e conexão com bancos de dados. Esse último representa o método mais comum utilizado atualmente devido a sua vantagem em manipular grandes quantidades de dados com boa performance.

Dessa maneira, foi criado o conceito de ciclo *CRUD*, que descreve as funcionalidades básicas de um banco de dados persistente (14). A referida sigla representa as quatro funções fundamentais de um banco de dados: *Create, Read, Update, Delete* (Criar, Ler, Atualizar e Apagar).

Levando o supracitado conceito para os sistemas de informação que utilizam conexões com bancos de dados, é premissa básica que esses sistemas possam manipular os dados provenientes de tais bancos, além de serem capazes de incluir novos registros quando necessário. Portanto, quando analisados a partir de sua camada mais próxima aos dados crus, todo sistema de informação que utiliza banco de dados representa um sistema *CRUD*.

O SiCarne, em suas versões anteriores, não apresentava o conceito de *CRUD* por não realizar nenhum tipo de acesso a banco de dados. Com as alterações realizadas por Lima (4), o SiCarne passou a contar com uma conexão a um banco de dados relacional para que os dados das análises realizadas pudessem ser persistidos para posterior leitura pelo pHremoto. Com isso, o conceito de *CRUD* foi incluído já que o SiCarne passou a fazer inserções, leituras e atualizações no banco. Após as implementações desse trabalho, o ciclo *CRUD* foi reforçado e ficou mais perceptível ao usuário através da possibilidade do cadastro direto de novos alimentos.

Capítulo 3

Desenvolvimento e Implementações

Para que o objetivo desse trabalho fosse alcançado, foram feitas implementações em ambos os sistemas, o SiCarne (7) e o pHremoto (4).

3.1 Implementações no SiCarne

Anteriormente à funcionalidade adicionada por esse trabalho, o SiCarne funcionava de modo limitado em relação aos possíveis alimentos alvos do experimento. O sistema se restringia a analisar somente determinados tipos de carne, sendo impossível para o usuário alterar tal comportamento.

Por ter sido concebido dessa forma, o SiCarne, em termos de implementação, não apresentava uma estrutura passível de escalabilidade. Analisando seu código, foi concluído que, para receber a funcionalidade de cadastro de novos alimentos, tanto elementos de interface do SiCarne quanto estratégias de codificação deveriam ser alterados.

Assim, as implementações realizadas no SiCarne foram divididas em implementações de interface gráfica e implementações internas.

3.1.1 Implementações de Interface Gráfica

A principal tarefa referente à funcionalidade de cadastro de novos alimentos foi a implementação da tela Cadastro de Novos Alimentos.

A tela de Cadastro de Novos Alimentos permite que o usuário do SiCarne cadastre novos alimentos que poderão ser utilizados como alvo do experimento. Com isso, de forma independente e rápida, o universo de utilização do SiCarne pode ser ampliado pelo próprio pesquisador. A tela possui três campos a serem preenchidos, sendo eles:

- Tipo de alimento: Campo de texto que deve receber o nome do tipo de alimento que será cadastrado no SiCarne. Esse é o nome que será exibido na caixa de seleção do tipo de alimento na janela principal do sistema;
- Mínimo pH bom: Campo de texto que deverá ser preenchido com o menor pH considerado bom para o alimento em questão. Esse dado será o limite inferior do intervalo que dirá se o alimento possui boa qualidade;
- Máximo pH bom: Campo de texto correspondente ao maior pH considerado bom para o alimento em cadastro. Esse será o limite superior do intervalo corresponde ao indicador de qualidade do alimento.

Destaca-se que a tela de Cadastro de Novos Alimentos pode ser visualizada na Figura 3.1

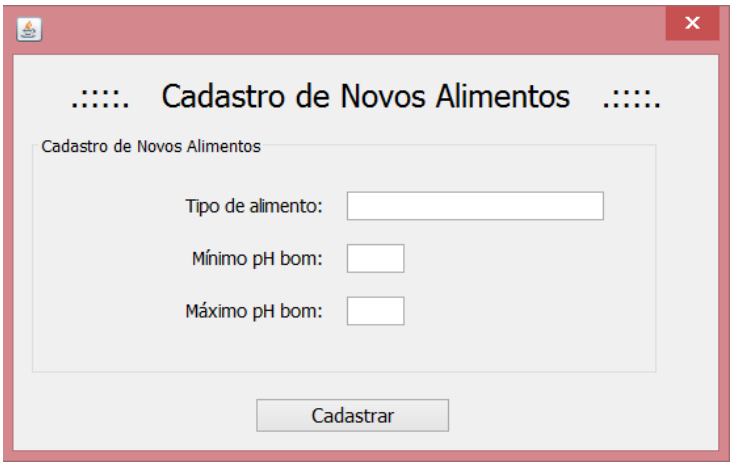

Figura 3.1: Tela de Cadastro de Novos Alimentos.

Ao cadastrar o alimento, as informações fornecidas passam por uma verificação de inconsistência. Caso o cenário de cadastro possua algum erro previsto pelo sistema, uma frase de alerta será exibida na própria tela indicando especificamente qual foi o erro detectado fazendo com que o sistema não conclua a transação de inserção. Os possíveis casos de erro são:

- Tentar cadastrar um alimento passando algum dos campos vazio;
- Fornecer valores de pH fora do intervalo de 0 a 14;
- Fornecer um valor de máximo pH bom menor ou igual ao valor de mínimo pH bom.

Ao passar pelas verificações, as informações são enviadas para o banco de dados "sicarne" do *MySQL Server* modificado por Lima (4). Elas serão armazenadas na tabela "alimento" nas colunas "nome", "chaophbom" e "tetophbom", respectivamente.

Ao final do processo de gravação no banco, o sistema exibirá uma janela de alerta informando ao usuário se a transação ocorreu com sucesso ou se houve algum problema. Caso o processo tenha sido concluído com sucesso, o fechamento do alerta ocasionará o mútuo fechamento da janela de Cadastro de Novos Alimentos. Contudo, caso tenha ocorrido algum erro na gravação dos dados do alimento, o fechamento da janela de Cadastro de Novos Alimentos não ocorrerá. Dessa forma, o usuário poderá tentar cadastrar o alimento novamente sem precisar voltar à tela principal do sistema.

As janelas de alerta podem ser vistas nas Figura 3.2 e Figura 3.3.

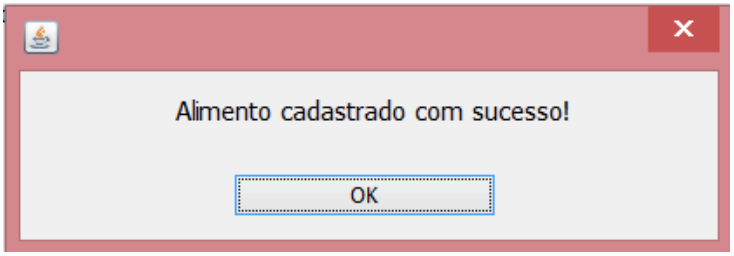

Figura 3.2: Alerta de sucesso na transação de cadastro de alimento.

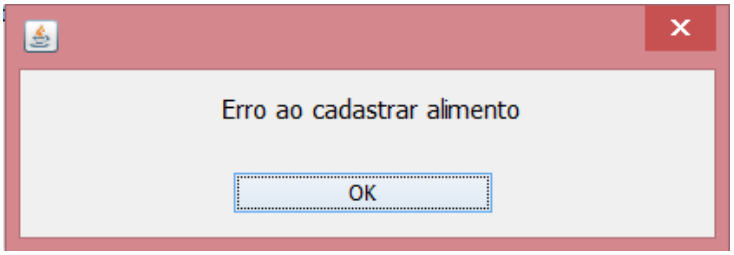

Figura 3.3: Alerta de erro no cadastro de alimento.

O desenvolvimento da tela de Cadastro de Novos Alimentos e da janela de alerta foi feito utilizando a tecnologia *Swing*, parte da suíte de desenvolvimento do *Java*. Isso permitiu que a compatibilidade com o restante do sistema fosse mantida, considerando que essa foi a tecnologia escolhida no momento de sua concepção. Além disso, por ser um recurso que permite a montagem da interface por *drag and drop*, o *Swing* diminuiu consideravelmente o tempo gasto no desenvolvimento da tela.

Com relação ao *design* das janelas, procurou-se manter o padrão utilizado em outras telas do sistema para que sua identidade visual não fosse prejudicada.

Além da criação da tela de Cadastro de Novos Alimentos e dos alertas de exibição do resultado da operação, houve também a necessidade de realizar pequenas alterações na janela principal do SiCarne para suportar a nova funcionalidade.

Na área "Comandos Avançados" foi criado o botão "Cadastrar Novo Alimento". Esse botão dá acesso à já mencionada tela de Cadastro de Novos Alimentos. Na mesma área, o *label* "Tipo de carne", utilizado na versão antiga para identificar a caixa de seleção dos tipos de carne suportados pelo SiCarne, foi alterado para "Tipo de alimento".

A Figura 3.4 mostra a nova área "Comandos Avançados" da janela principal do SiCarne.

3.1.2 Implementações Internas

Em sua primeira versão, o SiCarne somente realizava a análise de carnes. Não era possível automatizar o experimento de outros tipos de alimento pois o sistema não provia mais opções. Com as alterações realizadas nesse trabalho, o SiCarne passou a oferecer a opção de cadastro de novos alimentos. Contudo, para que o sistema realmente utilizasse tal funcionalidade, foi necessário não somente criar a interface de comunicação com o usuário, mas também modificar o sistema internamente para comportá-la.

A primeira alteração foi em relação à caixa de seleção dos tipos de alimento disponíveis para experimento. Na versão anterior do sistema, devido ao fato de somente serem permitidas análises de carnes, essa caixa de seleção era montada estaticamente. O modelo de dados utilizado para exibir as opções disponíveis eram strings escritas diretamente no código. A referida prática não permite a escabilidade do sistema, pois, para adicionar qualquer opção a ele, seria necessário abrir seu código e adicionar tal opção manualmente.

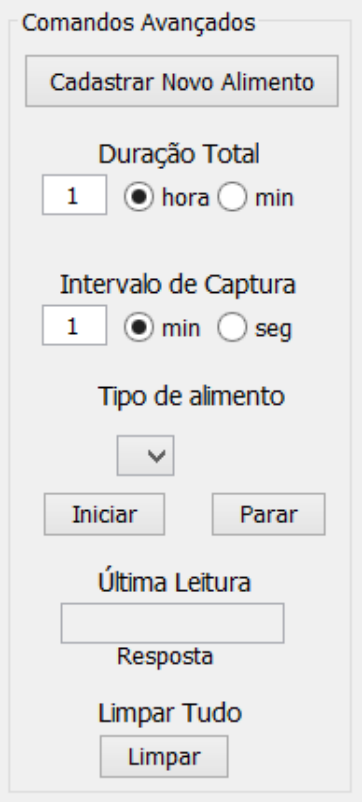

Figura 3.4: Área "Comandos Avançados" da tela principal do SiCarne.

Com base no exposto, a implementação do cadastro de novos alimentos não seria possível se essa estratégia fosse mantida. Assim, foi necessário alterar o modo como o modelo de dados da caixa de seleção é obtido. Com essa alteração, o sistema agora lista todos os alimentos pertencentes à tabela "alimento" do banco de dados "sicarne" e os coloca em um membro privado da janela. Posteriormente, os alimentos listados são adicionados à caixa de seleção na ordem em que vieram do banco. Tal prática permite que as opções de tipos de alimento oferecidas pelo sistema estejam sempre sincronizadas com o banco, tornando possível o cadastro de novos alimentos.

Por sua vez, caso ocorra algum erro ao listar os alimentos do banco de dados, o sistema construirá a caixa de seleção baseado nas opções disponíveis nas versões anteriores, ou seja, limitando-se a experimentos com carnes.

Outra alteração necessária foi na conclusão do experimento. Em versões prévias do sistema, por tratar somente de análises de carnes, os níveis de mínimo e máximo pH bom eram fixos no código para qualquer carne analisada. O texto de conclusão também era o mesmo para

qualquer objeto de experimento, variando apenas de acordo com a relação do pH medido e dos níveis de máximo e mínimo pH bom.

Atualmente, com o cadastro de novos alimentos implementado, cada alimento possui seus próprios valores de mínimo e máximo pH bom bem como seus textos de conclusão personalizados. Com isso, o SiCarne foi adaptado para utilizar os dados específicos do registro no banco do alimento selecionado no experimento para personalizar a conclusão.

3.2 Implementações no pHremoto

No phRemoto foi implementada a geração de um relatório contendo as informações referentes a determinada análise. Procurou-se utilizar a maior quantidade possível de dados gravados sobre a análise. Dessa forma, foi decidido que o relatório conterá os seguintes dados:

- Informações sobre o usuário responsável por realizar o experimento:
  - Identificador do usuário;
  - *Login* utilizado para acesso ao sistema.
- Informações inerentes à análise em si:
  - Identificador da análise;
  - Data e hora do início do experimento;
  - Tempo de duração do experimento expresso em minutos;
  - Frequência com que os dados de pH e temperatura foram coletados, em segundos;
  - Informação que indica se a análise foi ou não finalizada.
- Informações do alimento objeto da análise:
  - Identificador do alimento;
  - Nome do alimento utilizado;
  - Valor de mínimo pH bom;
  - Valor de máximo pH bom.
- Gráficos dos dados recolhidos durante a análise:
  - Gráfico de pH x Tempo;

– Gráfico de Temperatura x Tempo.

O texto de conclusão do experimento exibido pelo SiCarne não foi incluído no relatório. Essa decisão baseia-se no fato de tal texto ser fundamentado somente no último valor de pH medido. Diante disso, dependendo da duração da análise a conclusão não representaria fielmente o estado real do alimento. Para fins de interpretação do experimento, os gráficos inseridos no relatório são mais fidedignos à situação do alimento por trazerem variadas medições de pH e temperatura tomadas em períodos de tempo configurados pelo usuário.

Para a geração do relatório, foi adicionado o botão "Gerar relatório" na página de análises do pHremoto responsável por disparar o evento de criação do documento. Esse botão foi implementado utilizando *HTML* com estilização em *Bootstrap*, *framework* de *CSS* responsivo. Internamente, foi criada uma nova entidade no sistema responsável por encapsular os dados utilizados no documento. A mencionada entidade é preenchida através de uma consulta ao banco de dados "sicarne" que traz todas as informações necessárias por meio de *joins*. Dessa forma, é necessário somente uma ida ao banco, o que torna a leitura performática.

Após serem buscados do banco, os dados são utilizados para preencher a entidade Relatório. Assim, com todas as informações necessárias em memória, o sistema as usa para montar uma página *HTML* que implementa o *layout* do relatório. Essa página é guardada em uma *string* que será passada para o componente do *iText* responsável por convertê-la internamente para *PDF* e utilizará uma saída de arquivo para salvar a página convertida. Dessa forma, o arquivo gerado é salvo no sistema de arquivos da máquina do usuário podendo ser visualizado a qualquer momento. Por padrão, o caminho utilizado é a pasta *home* do usuário.

O relatório utiliza o identificador da análise na formação do nome do arquivo atribuindo cada relatório a uma análise. Portanto, o usuário é capaz de identificar o relatório correto que diz respeito à analise desejada.

Ao gerar o relatório com sucesso, o sistema deve exibir um *popup* dizendo ao usuário que o relatório foi criado com sucesso e mostrando seu caminho no sistema de arquivos. Caso ocorra algum erro no processo, um *popup* também deve ser exibido devendo, dessa vez, informar a ocorrência do erro.

Capítulo 4

Testes e Resultados

Para testar as funcionalidades implementadas, foi necessário preparar o ambiente de testes. Essa tarefa envolveu preparar o banco de dados inicial e gerar o *release* do SiCarne. Após preparado o ambiente, cada funcionalidade foi testada independentemente para verificar seu correto funcionamento.

4.1 Preparação do ambiente

Após as implementações realizadas, o SiCarne tornou-se uma aplicação completamente baseada em banco de dados. Isso significa que, para atingir seu pleno funcionamento, uma conexão ao seu banco de dados é fundamental para obter os recursos necessários. Dessa forma, foi utilizado o banco de dados "sicarne" implementado por Lima (4). Esse banco possui quatro tabelas, são elas:

- Alimento: Tabela responsável por guardar os dados referentes ao alimento. É a que possui maior importância para esse trabalho, pois é nela em que devem ser gravados os dados fornecidos pela tela de Cadastro de Novos Alimentos;
- Usuario: É a tabela responsável por guardar os registros de *login*. Para que um usuário tenha acesso ao SiCarne e ao pHremoto, seu login e sua senha devem estar nessa tabela;
- Analise: Guarda as informações das análises realizadas. Por ela é possível saber quem foi o usuário que realizou o experimento, qual o alimento objeto do estudo, horário de início, duração, entre outras informações;

- Dados: Tabela que salva os dados granulados do experimento. Para cada análise existirão N registros nessa tabela. Seus registros são utilizados para gerar os gráficos das análises de pH e temperatura.

A Figura 4.1 mostra o modelo do banco "sicarne" com suas tabelas e relacionamentos.

Figura 4.1: Modelo do banco de dados "sicarne".

Por conseguinte, foram criados dois usuários de banco: sicarne e phremoto. O usuário sicarne possui permissões de *insert*, *delete*, *update* e *select* no banco "sicarne", enquanto o usuário

phremoto possui apenas permissão de *select*. Cada sistema utilizará o respectivo usuário para acessar o banco de dados.

Para o funcionamento inicial do SiCarne, esse banco de dados foi populado com os alimentos suportados inicialmente pelo Sicarne: Carne de aves, carne bovina, carne equina, carne ovina e carne suína. Todas as carnes possuem mínimo pH bom de 5,3 e máximo pH bom de 5,7 conforme configuração inicial do SiCarne.

Para acesso aos sistemas, foram criados dois usuários: "sicarne", e "phremoto". Importante lembrar que por ser um banco de dados compartilhado entre o SiCarne o pHremoto, qualquer um dos usuários pode ser utilizado para autenticação independentemente do sistema utilizado. Para acesso aos sistemas, foram criados dois usuários: "sicarne", e "phremoto". Importante lembrar que por ser um banco de dados compartilhado entre o SiCarne o pHremoto, qualquer um dos usuários pode ser utilizado para autenticação independentemente do sistema utilizado.

Com relação às análises, de forma a acelerar os testes, foram incluídos no banco dados de simulação. Dessa forma, o pHremoto seria capaz de ler tais dados e gerar o relatório apropriadamente. Os dados inseridos na tabela "analise" podem ser vistos na Figura 4.2:

idanalise	idusuario	inicio	duracao	frequenciacoleta	idalimento	finalizada
1	1	2016-02-22 13:04:47	2	20	6	1

Figura 4.2: Conteúdo inserido para testes na tabela "analise"

Para essa análise, foram inseridos registros na tabela "dados" referentes à simulação de dados coletados. Os dados inseridos foram desenhados para estarem de acordo com o que foi inserido na tabela "analise". Dessa forma, os registros inseridos possuem intervalos de coleta de 20 segundos de modo a completar a duração determinada de 2 minutos. A Figura 4.3 ilustra o conteúdo da tabela "dados":

A máquina utilizada como servidor para o banco de dados foi um *ASUS* X550L com um processador *Intel Core* i5 - 4210U com quatro núcleos de 1,7 *GHz*, memória *RAM* de 6 *GB DDR3* utilizando o *Windows* 8.1 como sistema operacional. O servidor de banco de dados usado foi o *MySQL Server* para *Windows*.

iddado	idanalise	ph	temperatura	datahora
1	1	3.7	35.7	2016-02-21 15:18:11
2	1	3.7	25.7	2016-02-21 15:18:31
3	1	3.6	25.9	2016-02-21 15:18:51
4	1	3.4	26	2016-02-21 15:19:11
5	1	3.9	25.8	2016-02-21 15:19:31
6	1	4	25.6	2016-02-21 15:19:51

Figura 4.3: Conteúdo de simulação na tabela "dados".

Para gerar a versão *release* da aplicação basta construí-la. Isso gerará um arquivo *JAR* que poderá ser executado em qualquer sistema operacional que possuir a Máquina Virtual Java (*JVM*) instalada. O arquivo *JAR* executará a classe definida como principal no projeto. No caso do SiCarne, essa classe é a tela de *login*.

Por fim, os testes foram realizados localmente, ou seja, com as aplicações e o servidor de banco de dados sendo executados na mesma máquina.

4.2 Testes no SiCarne

Para realizar os testes com o SiCarne, primeiramente é necessário configurar seu acesso ao banco de dados. Isso é feito através do arquivo "bd.conf". Esse arquivo deve estar na mesma pasta em que o *release* da aplicação estiver. A primeira linha desse arquivo determina o endereço *IP* do servidor hospedeiro do banco de dados e o nome da base. O formato correto dessa linha é "endereço:porta/banco", sem aspas. A segunda linha do arquivo "bd.conf" deve trazer o usuário de autenticação do banco de dados. Finalmente, a terceira linha deve conter a senha do usuário especificado na linha acima. Linhas começadas por "#" são entendidas como comentários e não são lidas pelo sistema.

No caso do SiCarne, o arquivo "bd.conf" foi configurado conforme o exibido na Figura 4.4.

Dessa maneira, serão executados vários cenários de testes para verificar as funcionalidades implementadas.

```
#Abaixo o endereco do banco de dados
localhost:3036/sicarne
#Abaixo o login do banco de dados
sicarne
#Abaixo a senha do banco de dados
"senha"
```

Figura 4.4: Arquivo de configuração para acesso a banco do SiCarne.

4.2.1 Testes do Cadastro de Novos Alimentos

A finalidade dos primeiros cenários de teste executados foi verificar a nova tela de Cadastro de Novos Alimentos.

Para isso, é necessário executar o arquivo *JAR* gerado na pasta "dist" do projeto. Tal arquivo iniciará a execução da aplicação exibindo a tela de *login*. O usuário fornecido deve estar registrado na tabela "usuário" do banco "sicarne" para que ocorra a autenticação. Realizado o *login*, o sistema exibirá sua janela principal onde deve ser escolhida a opção "Cadastrar Novo Alimento".

O primeiro teste executado foi tentar cadastrar um alimento deixando algum dos campos em branco. O comportamento esperado para esse teste é que a tela de Cadastro de Novos Alimentos exiba uma mensagem de erro dizendo que é necessário preencher todos os campos. Foram testadas realizações de cadastro com todos os casos de campos em branco. A Figura 4.5 mostra o resultado obtido em todos os casos.

Posteriormente, foi testado o cenário em que o máximo pH bom é menor ou igual ao mínimo pH bom. Nesse caso, deve ser exibida uma mensagem de erro dizendo que o mínimo pH bom deve ser menor que o máximo pH bom. A Figura 4.6 mostra a tela Cadastro de Novos Alimentos após a execução desse cenário de teste.

O objetivo do próximo cenário de teste executado foi mostrar qual o comportamento do sistema ao tentar cadastrar um alimento quando há uma falha na conexão com o banco de dados. Uma vez realizado o *login*, o servidor de banco de dados foi desligado matando a conexão à base. Em seguida, foi feita a tentativa de cadastro de alimento. Nesse caso, o sistema deve exibir um *popup* dizendo que houve erro no cadastro. A Figura 4.7 ilustra a tela de Cadastro de Novos Alimentos após a realização do teste.

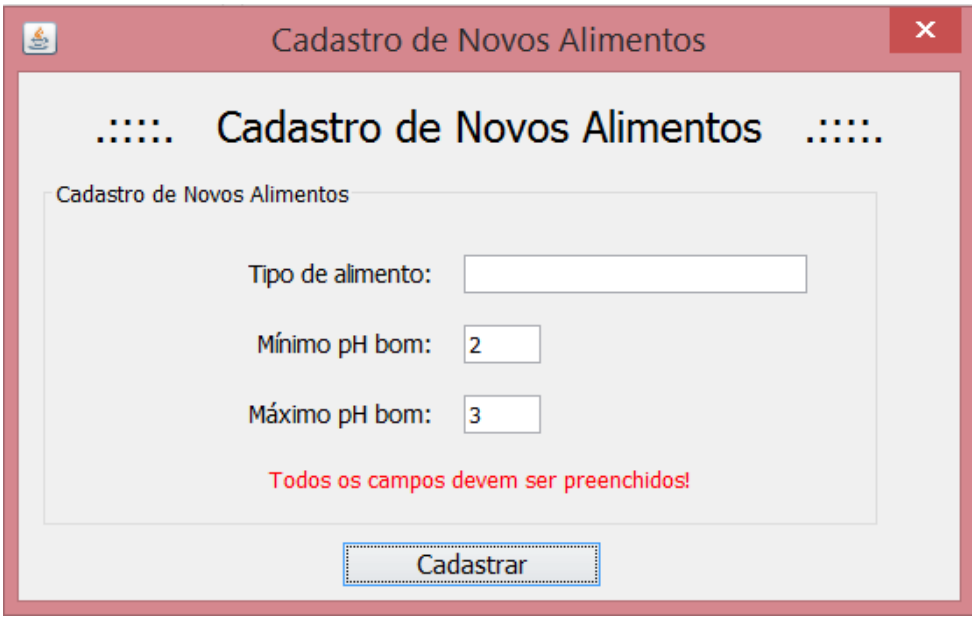

Figura 4.5: Tentativa de cadastro com campos não preenchidos.

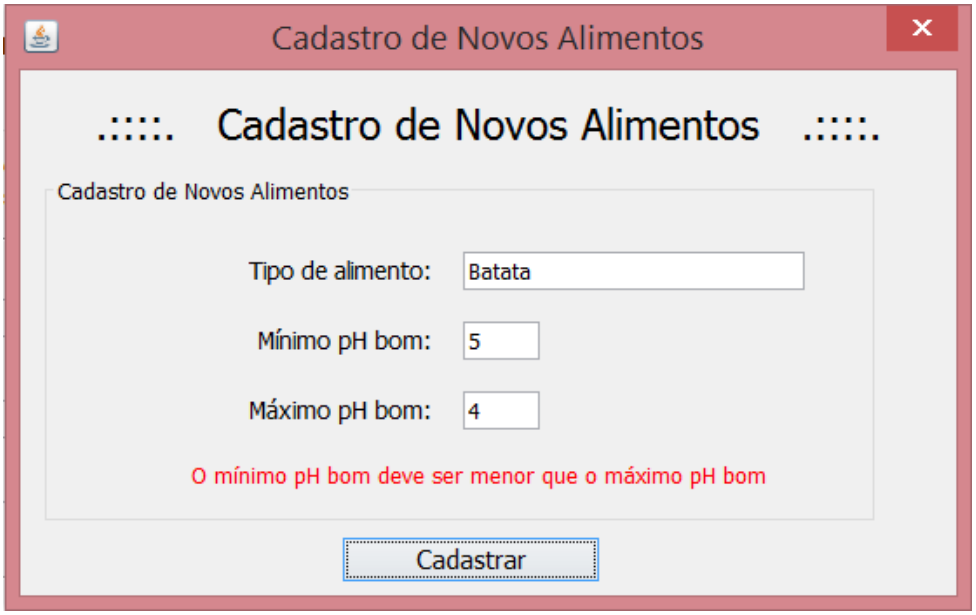

Figura 4.6: Tela de cadastro de alimentos após fornecer mínimo pH bom maior ou igual ao máximo pH bom.

O último cenário de teste do cadastro de novos alimentos foi verificar qual a ação do sistema ao conseguir salvar um cadastro com sucesso. Isso requereu que todos os dados fornecidos pela tela tivessem passado pela verificação de consistência e que a conexão com o banco tivesse sido estabelecida estavelmente. Nesse caso, os registros devem ser gerados na tabela

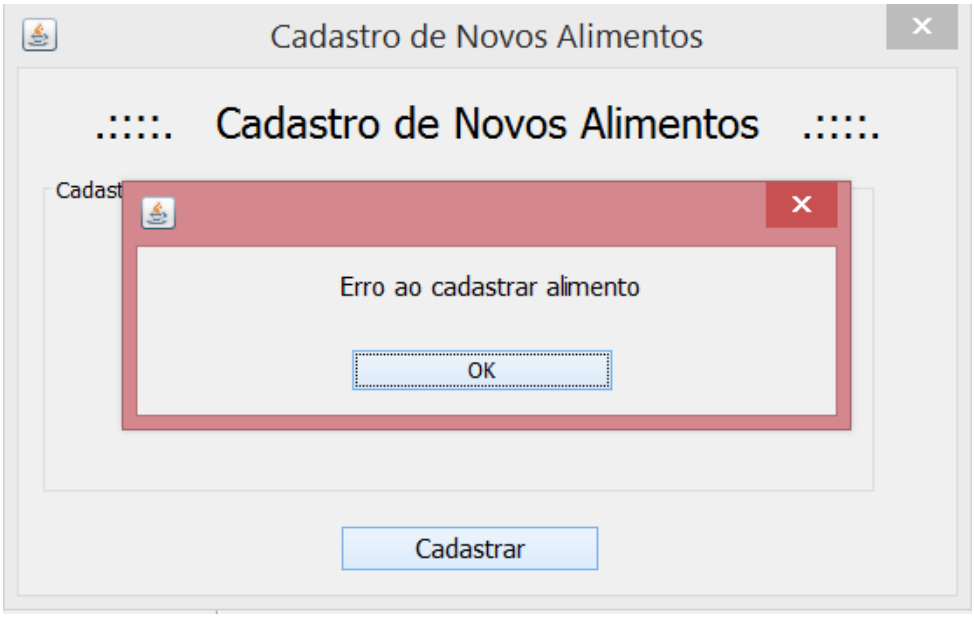

Figura 4.7: Erro ao tentar salvar o alimento no banco.

"alimento" do banco "sicarne" e o sistema deve exibir um *popup* mostrando ao usuário que a operação foi concluída com sucesso.

A Figura 4.8 mostra o conteúdo da tabela "alimento" previamente ao cadastro.

```
mysql> select * from alimento;
+------------+---------------+-----------+-----------+------------------+----------------+-----------------+
| idalimento | nome          | chaophbom | tetophbom | conclusaophbaixo | conclusaophbom | conclusaophalto |
+------------+---------------+-----------+-----------+------------------+----------------+-----------------+
|          1 | Carne de aves |       5.3 |       5.7 | NULL             | NULL           | NULL            |
|          2 | Carne bovina  |       5.3 |       5.7 | NULL             | NULL           | NULL            |
|          3 | Carne equina  |       5.3 |       5.7 | NULL             | NULL           | NULL            |
|          4 | Carne ovina   |       5.3 |       5.7 | NULL             | NULL           | NULL            |
|          5 | Carne suína   |       5.3 |       5.7 | NULL             | NULL           | NULL            |
+------------+---------------+-----------+-----------+------------------+----------------+-----------------+
5 rows in set (0.00 sec)
```

Figura 4.8: Conteúdo da tabela "alimento" antes de cadastrar novos alimentos.

Como evidenciado pela imagem, a tabela "alimento" contém somente os alimentos inseridos na preparação do ambiente de testes. Com isso, serão inseridos pela tela de Cadastro de Novos Alimentos os alimentos Pepino, Cebola vermelha e Leite. Seus pHs são, respectivamente, 3,8, 5,5 e 6,3 (15). Dessa forma, foi decidido que os níveis de mínimo e máximo pH bom serão determinados por uma margem de 0,2. Portanto, seus valores de mínimo pH bom serão 3,6, 5,3 e 6,1, ficando seus valores de máximo pH bom em 4, 5,7 e 6,5, respectivamente.

Ao executar cada inserção com sucesso, a tela de cadastro exibe o *popup* indicando que as alterações no banco foram concluídas corretamente. Como comportamento pré-definido, a tela de Cadastro de Novos Alimentos é terminada no momento em que o *popup* é fechado, devendo o usuário abri-la novamente a partir da janela principal do sistema. A Figura 4.9 mostra a tela de Cadastro de Novos Alimentos após a conclusão com sucesso do processo.

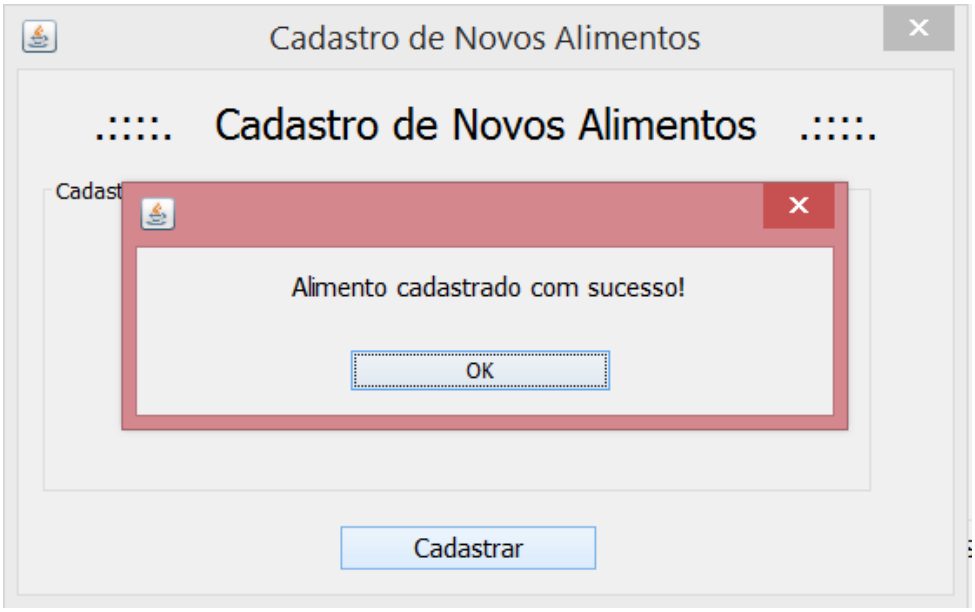

Figura 4.9: Tela de Cadastro de Novos Alimentos ao cadastrar um alimento com sucesso.

Após cadastrar os alimentos previamente citados, eles agora devem estar disponíveis para utilização pelo SiCarne. Para tanto, eles devem estar registrados na tabela "alimento". A Figura 4.10 ilustra tal tabela após a inserção dos alimentos Pepino, Cebola vermelha e Leite.

```
mysql> select * from alimento;
+------------+----------------+-----------+-----------+-----------------+----------------+----------------+
| idalimento | nome           | chaophbom | tetophbom | conclusaophbaixo | conclusaophbom | conclusaophalto |
+------------+----------------+-----------+-----------+-----------------+----------------+----------------+
|          1 | Carne de aves  |       5.3 |       5.7 | NULL            | NULL           | NULL           |
|          2 | Carne bovina   |       5.3 |       5.7 | NULL            | NULL           | NULL           |
|          3 | Carne equina   |       5.3 |       5.7 | NULL            | NULL           | NULL           |
|          4 | Carne ovina    |       5.3 |       5.7 | NULL            | NULL           | NULL           |
|          5 | Carne suína    |       5.3 |       5.7 | NULL            | NULL           | NULL           |
|          6 | Pepino         |       3.6 |         4 | NULL            | NULL           | NULL           |
|          7 | Cebola vermelha |       5.3 |       5.7 | NULL            | NULL           | NULL           |
|          8 | Leite          |       6.1 |       6.5 | NULL            | NULL           | NULL           |
+------------+----------------+-----------+-----------+-----------------+----------------+----------------+
8 rows in set (0.00 sec)
```

Figura 4.10: Tabela "alimento" após os cadastros de alimentos.

Salienta-se a necessidade de reiniciar o SiCarne para que os novos alimentos cadastrados sejam listados.

4.2.2 Testes da Caixa de Seleção de Tipo de Alimento

Em sua versão atual, o SiCarne lista todos os alimentos do banco de dados e os utiliza para montar a caixa de seleção dos tipos de alimentos que o usuário pode escolher para realizar o experimento. Dessa forma, existem basicamente dois cenários de teste para tal funcionalidade. O primeiro cenário diz respeito a quando o sistema não é capaz de ler os alimentos da base de dados. Quando isso ocorre, o SiCarne deve montar a caixa de seleção de tipos de alimento com base nas versões anteriores do sistema. Portanto, o sistema oferecerá somente as carnes previstas por Nascimento (7).

Como está implementado atualmente, o SiCarne acessa o banco buscando os alimentos nele contidos no momento da construção da janela principal do sistema. Assim, a leitura é feita imediatamente após a autenticação do usuário, fazendo com que a realização do teste desse cenário não seja possível localmente na visão de usuário. Portanto, para efeitos de verificação da funcionalidade, o projeto foi executado em modo de depuração e, realizada a autenticação de usuário, a conexão com o banco foi cortada para que não fosse possível realizar a leitura dos alimentos.

A Figura 4.11 mostra como a caixa de seleção dos tipos de alimentos se comportou ao não ser possível ler os alimentos do banco.

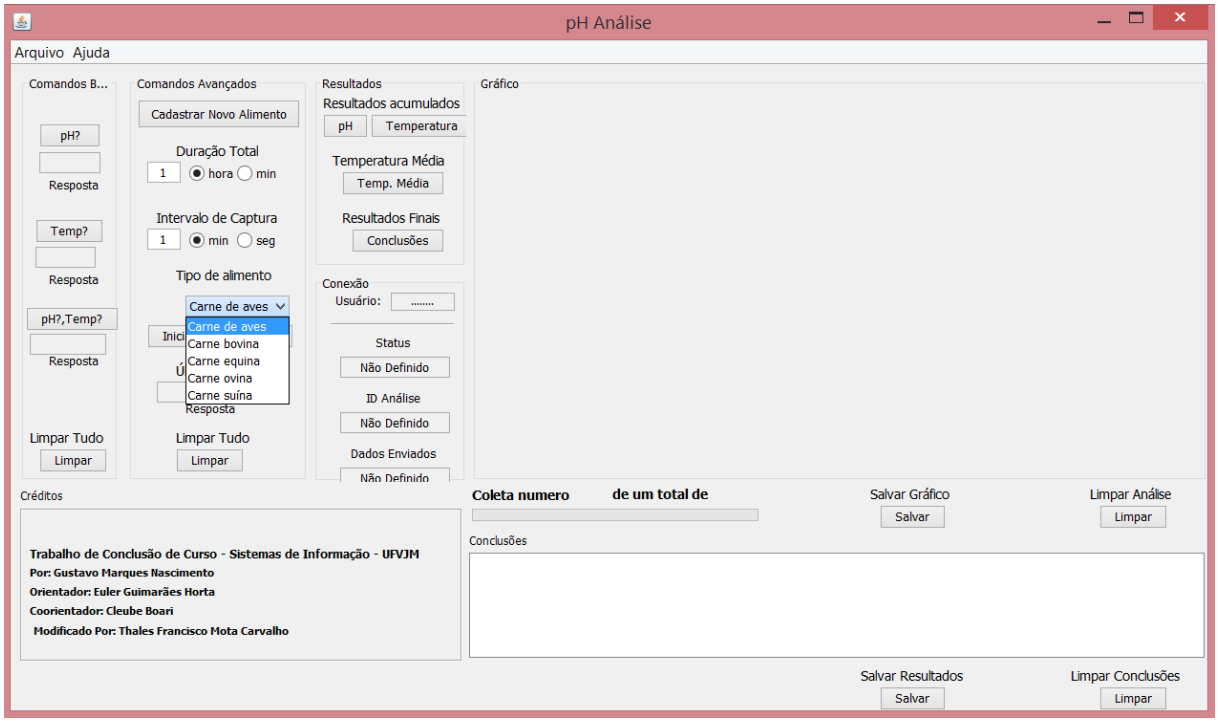

Figura 4.11: Caixa de seleção de tipos de alimento após erro na leitura dos alimentos do banco de dados.

O segundo cenário possível nos testes da caixa de seleção dos tipos de alimento ocorre quando o sistema consegue ler os alimentos da base de dados. Nesse caso, todos alimentos registrados na tabela "alimento" devem ser exibidos como opções para o usuário. O referido teste foi realizado após o cadastro dos alimentos citados na Subseção 4.2.1. Dessa forma, o usuário do SiCarne deve ser capaz de visualizá-los na caixa de seleção de tipos de alimento. A Figura 4.12 ilustra a janela principal do sistema quando a listagem dos alimentos foi realizada com sucesso.

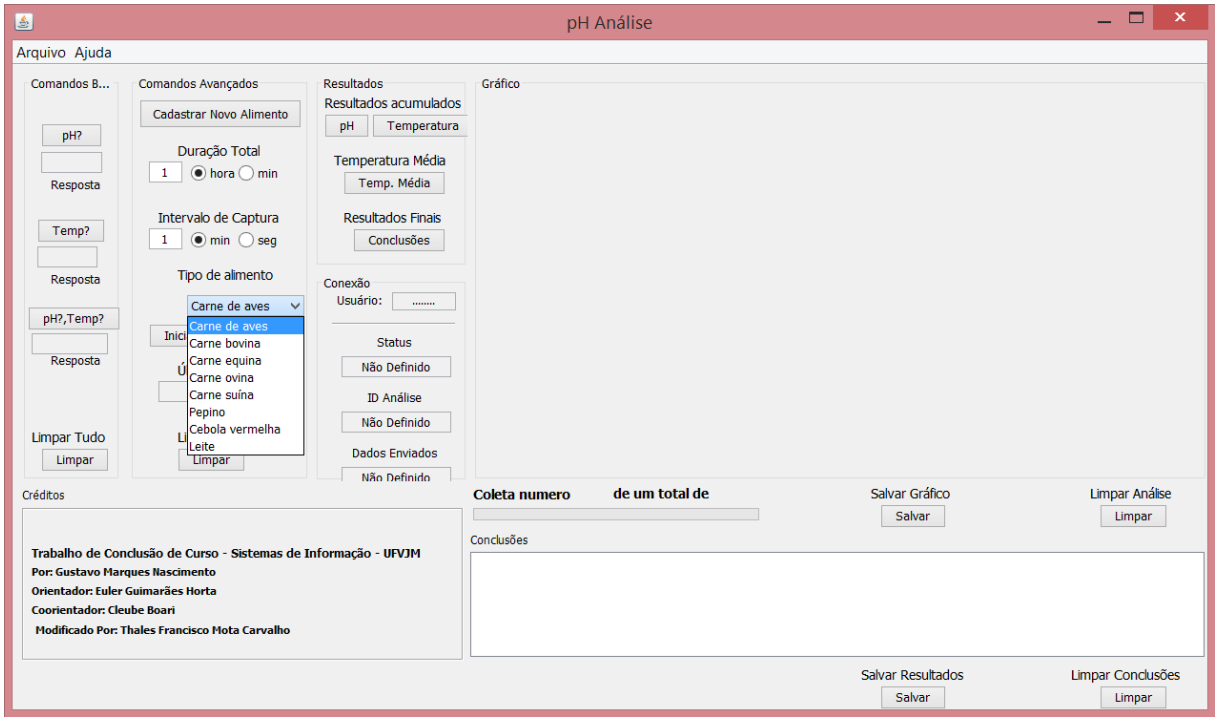

Figura 4.12: Caixa de seleção de tipos de alimento após leitura dos alimentos do banco de dados.

4.2.3 Testes de Conclusão do Experimento

Previamente às implementações feitas pelo presente trabalho, os textos de conclusão dos experimentos eram fixos e não levavam em consideração as particularidades de cada alimento. Todos os mínimos e máximos pHs bons eram fixos em 5,3 e 5,7, respectivamente, e o método que gerava as conclusões não possuía estrutura escalável.

Com a variabilidade dos alimentos após as funcionalidades adicionadas, esse método foi alterado para considerar os pHs específicos de cada alimento. Com relação ao texto da conclusão, foi adicionado um modelo geral aplicável para qualquer alimento. Para manter o padrão da versão anterior, o antigo texto utilizado para conclusões de experimentos feitos com as carnes foi mantido. Ele é utilizado quando o sistema determina que alguma dessas carnes é o objeto do experimento. Assim, o usuário que já utilizava o SiCarne não sentirá o impacto das alterações quando realizar experimentos legados.

Com base no exposto, há dois cenários de teste relativos à conclusão do experimento. O primeiro considera um experimento feito com uma carne que já era oferecida nas versões

anteriores do SiCarne. A Figura 4.13 a seguir mostra como era a conclusão de um experimento feito em uma versão antiga do sistema.

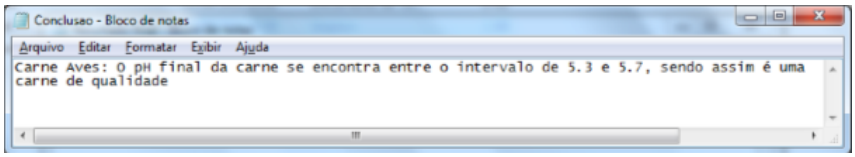

Figura 4.13: Conclusão do experimento em uma versão antiga do SiCarne.

Após as alterações efetuadas nas conclusões, foi realizado um experimento com o mesmo tipo de alimento utilizado para gerar a conclusão mostrada na Figura 4.13. A Figura 4.14 ilustra como ficou a conclusão de experimentos legados no SiCarne posteriormente à realização das melhorias.

O segundo cenário de teste relativo à conclusão do experimento tem como base a escolha de um alimento que não estava presente nas versões do sistema, ou seja, cadastrado pela nova funcionalidade de Cadastro de Novos Alimentos. Nesse caso, o SiCarne deve montar a conclusão do experimento utilizando o texto geral. A Figura 4.15 mostra a conclusão de uma análise utilizando o Pepino como objeto do estudo.

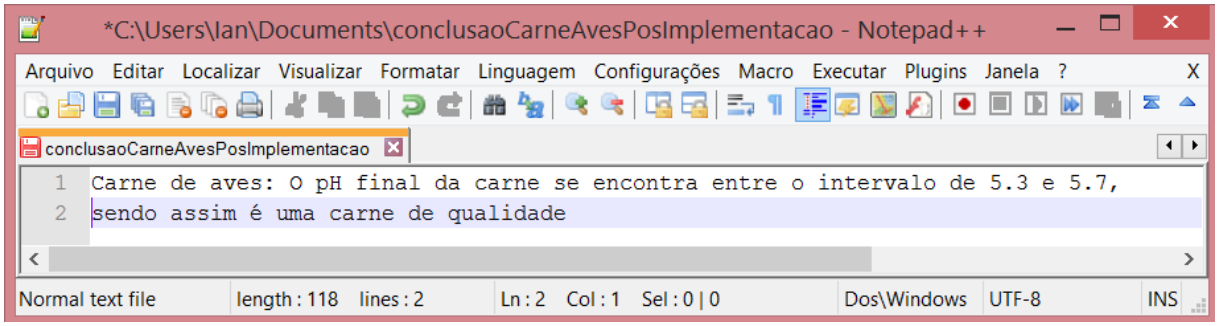

Figura 4.14: Conclusão de experimento legado após implementações.

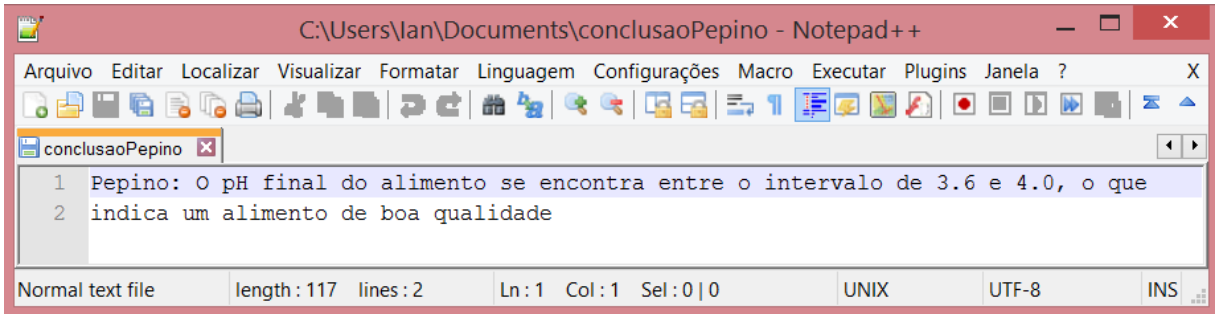

Figura 4.15: Conclusão de experimento utilizando alimento cadastrado.

4.3 Testes no pHremoto

Para testar as funcionalidades implementadas no pHremoto, o sistema foi executado do mesmo modo em que seria acessado em condições reais de uso. O *login* "sicarne/projetosicarneufvjm" foi utilizado para que o sistema consiga buscar a análise de simulação no banco de dados.

O primeiro teste executado foi verificar o *layout* do novo botão "Gerar relatório". Ele deve apenas ser exibido quando há uma análise válida selecionada, não devendo ser exibido na página inicial do pHremoto. A Figura 4.16 mostra a página inicial do pHremoto uma vez realizado o *login*.

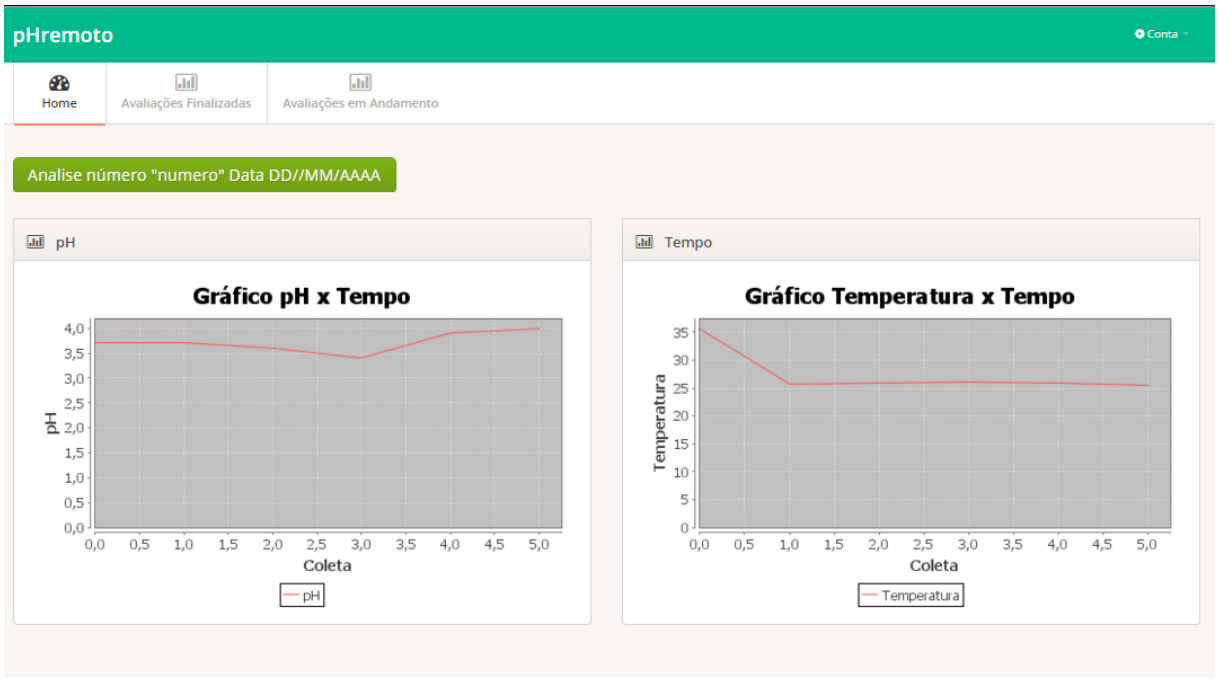

Figura 4.16: Tela inicial do pHremoto. Não há análise selecionada.

Após selecionada uma análise na área "Avaliações Finalizadas", o pHremoto deve exibir o botão "Gerar relatório" permitindo que um documento contendo os dados da análise selecionada seja gerado. A Figura 4.17 mostra a tela de análises do pHremoto após selecionada uma análise.

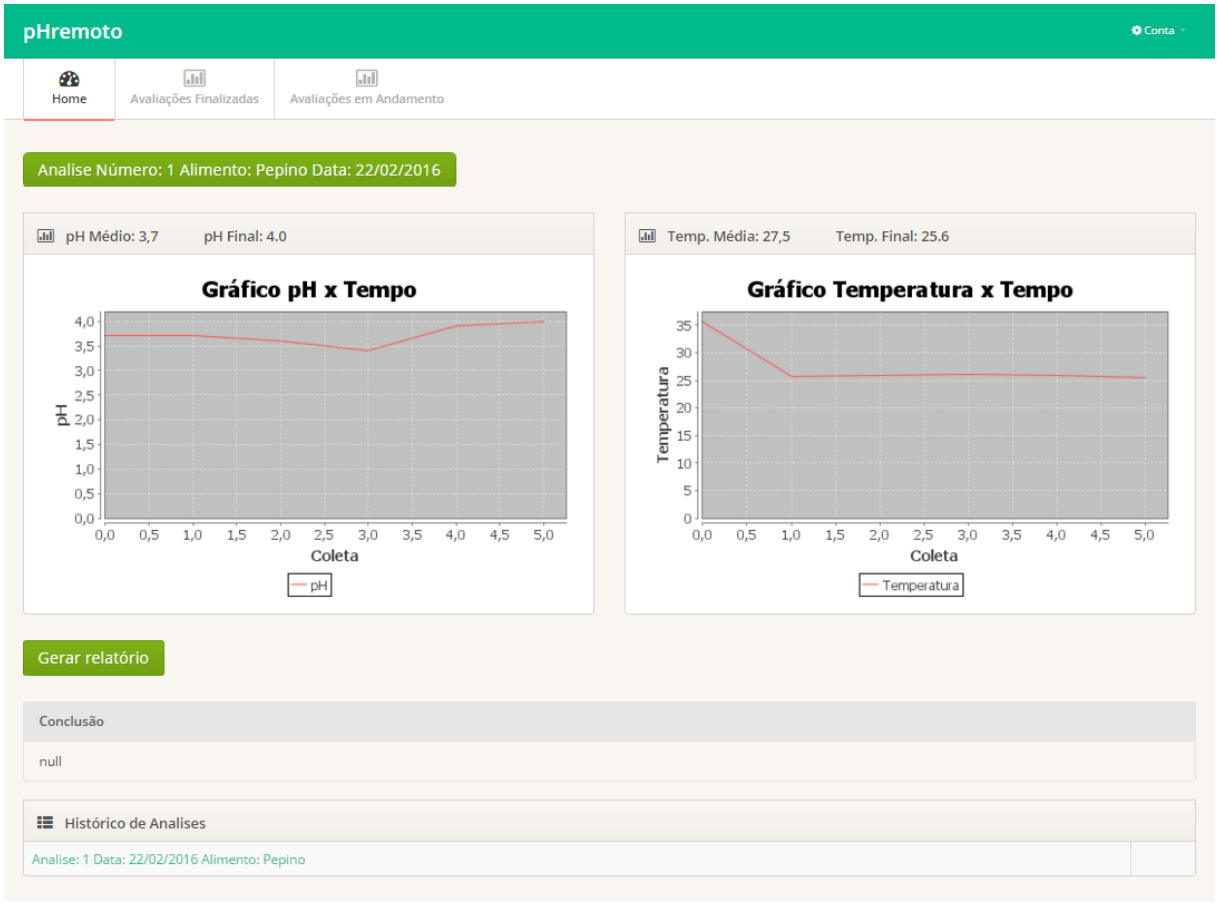

Figura 4.17: Tela principal do pHremoto após selecionar uma análise.

O próximo cenário executado foi para testar a ocorrência de erro na geração do relatório. Para isso, após logar no sistema e selecionar uma análise na área "Avaliações Finalizadas", a conexão com o banco de dados foi terminada e a geração do relatório foi requisitada através do botão "Gerar relatório". Dessa forma, o sistema lançou uma exceção ao buscar o conteúdo do relatório. O comportamento do sistema diante desse caso está ilustrado na Figura 4.18.

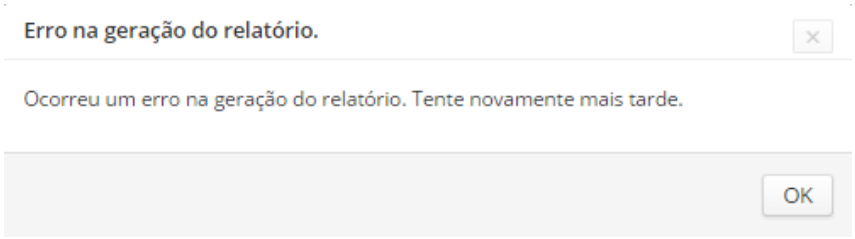

Figura 4.18: Erro na geração do relatório.

Por último, foi testada a geração com sucesso de um relatório. Após selecionada a análise, o botão "Gerar relatório" foi clicado disparando a geração do relatório. O resultado dessa operação é mostrado na Figura 4.19.

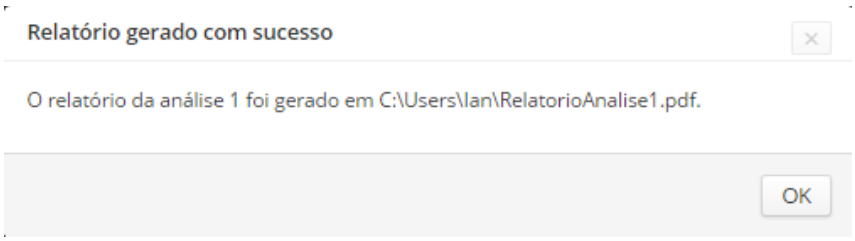

Figura 4.19: Tela de visualização de análise geração do relatório.

Após gerado, o relatório estará no caminho indicado na Figura 4.19. A Figura 4.20 ilustra seu conteúdo.

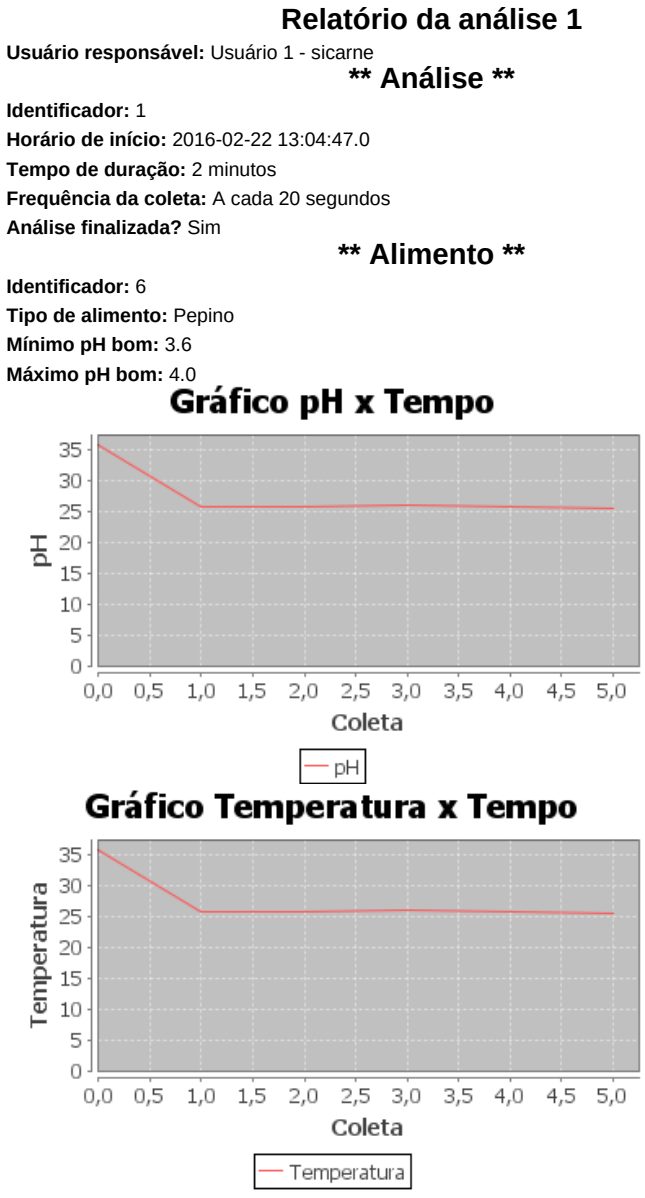

Figura 4.20: Relatório gerado pelo pHremoto.

Capítulo 5

Conclusão

A segurança de consumo dos alimentos é fator fundamental para a saúde e bem-estar do ser humano. Essa é uma preocupação intrínseca à Engenharia de Alimentos, considerando que é de sua responsabilidade garantir que os alimentos produzidos estejam nas condições ideais de consumo. Assim, a medição do pH e da temperatura dos alimentos é um método eficaz e amplamente utilizado para garantir que o alimento esteja microbiologicamente estável, garantindo que microrganismos não atinjam níveis infecciosos.

Para automatizar as análises de pH e temperatura realizadas na UFVJM, foi implementado o SiCarne. Seu desenvolvimento foi feito com a tecnologia *Swing*, presente no *kit* de desenvolvimento do *Java*. Contudo, ele se limitava a realizar análises de carnes fixadas no código do sistema. O presente trabalho construiu uma funcionalidade que possibilitou o cadastro de novos alimentos tornando o SiCarne uma aplicação escalável. Para tal, além da implementação de uma nova tela de Cadastro de Novos Alimentos, foram realizadas melhorias na estruturação do sistema responsáveis por adaptá-lo à generalização do seu universo de objetos de estudo.

O SiCarne também era limitado em relação à coleta dos resultados. Era necessário estar no local da realização dos experimentos para visualizar suas conclusões. Para eliminar tal deficiência, foi criado o pHremoto, sistema que oferece uma interface *Web* responsiva para monitoramento remoto das análises. Ele foi desenvolvido com *back-end* em *Java* juntamente com páginas codificadas em *HTML, CSS* e *JavaScript* utilizando o padrão de projetos *MVC*. Contudo, o pHremoto se limitava às funcionalidades oferecidas pelo SiCarne. Dessa forma, esse trabalho adicionou ao pHremoto a geração de relatórios contendo os dados referentes à análise. A mencionada funcionalidade reuniu as informações relativas às análises fazendo com que o trabalho do pesquisador ficasse mais eficiente, eliminando assim o desperdício de tempo

relativo à coleta dos dados dos resultados. A geração do relatório foi implementada em duas etapas. A primeira busca os dados necessários do banco e os insere em um código *HTML* que formata o *layout* do relatório. A segunda passa o código *HTML* gerado para o componente do *iText* que se encarrega de realizar a conversão para *PDF*.

5.1 Trabalhos Futuros

O tratamento de exceções do SiCarne é precário. Há casos em que o sistema é terminado devido a uma exceção lançada sem repassar ao usuário o que aconteceu. Para versões futuras do SiCarne, espera-se que o sistema seja refatorado para que o tratamento de erros e exceções reflita a orientação ao usuário e para que a execução do sistema se estabilize.

Com relação à segurança, quando o cadastro de um novo usuário é feito, a senha fornecida é gravada diretamente no banco. A autenticação é realizada simplesmente pegando a senha fornecida e verificando se existe seu registro no banco de acordo com o usuário dado. Isso faz com que possíveis ataques ao banco recuperem as reais senhas dos usuários. Para versões futuras do SiCarne, o cadastro de usuários deve fazer a criptografia da senha antes de gravá-la no banco. Assim, no momento da autenticação, a senha criptografada deve ser descriptografada e só então comparada à senha fornecida no *login*.

Para o pHremoto, sugere-se o desenvolvimento de um aplicativo nativo para dispositivos móveis. Tal abordagem melhora a segurança do acesso remoto reduzindo a necessidade de obtenção de um certificado digital. Como alternativa, pode ser implementada uma estratégia de código de autenticação. Dessa forma, cada instalação do aplicativo geraria um código aleatório que seria gravado no banco. No momento da autenticação do usuário, o aplicativo mandaria seu código para o banco que verificaria se ele é válido ou não. Caso o banco ateste a validade do código, a autenticação do usuário seguiria seu fluxo. Caso o código não seja válido, o sistema não poderia ser utilizado. Com isso, a instalação do aplicativo poderia ser controlada limitando quais usuários poderiam utilizar o sistema.

Referências Bibliográficas

[1] SIDONIO, M. Fatores intrÍnsecos e extrÍnsecos. http://nutrimico.blogspot.com.br/2012/10/fatores-1.html, 2012. Acessado em Fevereiro, 2016.

[2] CARDOSO, M. Escala de ph. http://www.infoescola.com/quimica/escala-de-ph/, 2006/2016. Acessado em Fevereiro, 2016.

[3] FIB. Segurança alimentar, 2008. Acessado em Fevereiro, 2016.

[4] LIMA, D. S.; FONSECA, A. R. Desenvolvimento de um sistema web responsivo para comunicação e análise de dados provenientes de phmetro aplicado à análise de alimentos. Journal Desconhecido, 2014.

[5] LOPES, M. R. F.; PEREIRA, A. S. C. Manejo pré-abate e qualidade da carne. Journal Desconhecido, 2006. Acessado em Fevereiro, 2016.

[6] MENDONÇA, F. C. Relatório de práticas em métodos instrumentais de análise. aula prática 01: Determinação do ph em diferentes soluções, 2013. Acessado em Fevereiro, 2016.

[7] NASCIMENTO, G. M.; AES HORTA, E. G. Aplicativo para comunicação e análise de dados provenientes de phmetro para o emprego na análise de variação de ph em alimentos e matérias-primas agroalimentares. Journal Desconhecido, 2011.

[8] RXTX. Faq, 2009. Acessado em Fevereiro, 2016.

[9] JFREECHART. Titulo desconhecido, 2005/2014. Acessado em Fevereiro, 2016.

[10] BASS, L.; CLEMENTS, P.; GARLAN, D.; IVERS, J.; LITTLE, R.; NORD, R.; STAFFORD, J. Documenting software architectures: Views and beyond. Pearson Education, 2002. Acessado em Fevereiro, 2016.

[11] BURBECK, S. Applications programming in smalltalk-80^{TM}: How to use model-view-controller (mvc). Journal Desconhecido, 1987/1992. Acessado em Fevereiro, 2016.

[12] REENSKAUG, T. Mvc. Journal Desconhecido, 1979. Acessado em Fevereiro, 2016.

[13] CRAWFORD, W.; HUNTER, J. Java servlet programming. Segunda. ed. O'Reilly, 2001. Acessado em Fevereiro, 2016.

[14] ROUSE, M. Crud cycle (create, read, update and delete cycle): What is the crud cycle? TechTarget, 2008.

[15] ESTÉTICA, C. P. D. A. D. E. O ph dos alimentos. Portal Educação, 2013. Acessado em Fevereiro, 2016.