UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
ENGENHARIA ELÉTRICA

NATHAN FERNANDES

DESENVOLVIMENTO DE UMA INTERFACE COMPUTACIONAL PARA ACESSO A UM BANCO DE DADOS SQL DE SISTEMAS SUPERVISÓRIOS INDUSTRIAIS

TRABALHO DE CONCLUSÃO DE CURSO

CORNÉLIO PROCÓPIO
2020

NATHAN FERNANDES

DESENVOLVIMENTO DE UMA INTERFACE COMPUTACIONAL PARA ACESSO A UM BANCO DE DADOS SQL DE SISTEMAS SUPERVISÓRIOS INDUSTRIAIS

Trabalho de Conclusão de Curso de graduação, apresentado à disciplina Trabalho de Conclusão de Curso 2, do curso de Engenharia Elétrica da Universidade Tecnológica Federal do Paraná – UTFPR, como requisito parcial para a obtenção do título de Bacharel.

Orientador: Prof. Dr. Wagner Endo

CORNÉLIO PROCÓPIO
2020

Universidade Tecnológica Federal do Paraná
Campus Cornélio Procópio
Departamento Acadêmico de Elétrica
Curso de Engenharia Elétrica

UTFPR
UNIVERSIDADE TECNOLÓGICA FEDERAL DO PARANÁ
CAMPUS CORNÉLIO PROCÓPIO

FOLHA DE APROVAÇÃO

Nathan Fernandes

Desenvolvimento de uma interface computacional para acesso a um banco de dados SQL de Sistemas Supervisórios Industriais

Trabalho de conclusão de curso apresentado às 16:00hs do dia 06/10/2020 como requisito parcial para a obtenção do título de Engenheiro Eletricista no programa de Graduação em Engenharia Elétrica da Universidade Tecnológica Federal do Paraná. O candidato foi arguido pela Banca Avaliadora composta pelos professores abaixo assinados. Após deliberação, a Banca Avaliadora considerou o trabalho aprovado.

_____
Prof(a). Dr(a). Wagner Endo - Presidente (Orientador)

_____
Prof(a). Dr(a). Cristiano Marcos Agulhari - (Membro)

_____
Prof(a). Dr(a). Paulo Rogério Scalassara - (Membro)

A folha de aprovação assinada encontra-se na coordenação do curso.

RESUMO

FERNANDES, Nathan. DESENVOLVIMENTO DE UMA INTERFACE COMPUTACIONAL PARA ACESSO A UM BANCO DE DADOS SQL DE SISTEMAS SUPERVISÓRIOS INDUSTRIAIS. 2020. 62 f. Trabalho de Conclusão de Curso (Graduação) – Engenharia Elétrica. Universidade Tecnológica Federal do Paraná. Cornélio Procópio, 2020.

Este trabalho propôs a construção de um sistema supervisório alternativo de uma planta industrial para a coleta e extração das informações do banco de dados do supervisório principal, com o objetivo de disponibilizar suas informações em um servidor hospedado remotamente. Para atingir o objetivo foi construído um supervisório simples, utilizando a ferramenta NetBeans, para rodar em qualquer máquina que seja necessário acessar as informações e comandos da planta industrial, e também foi desenvolvido um outro software para rodar em conjunto com a máquina local para extrair os dados do banco de dados local para o banco de dados remoto. Neste trabalho foi possível entender os conceitos de registro e leitura de dados, em um banco de dados relacional hospedado em um servidor remoto.

Palavras-chave: Sistemas Supervisórios. Automação Industrial. Indústria 4.0. Redes Industriais. Bancos de Dados Relacionais.

ABSTRACT

FERNANDES, Nathan. DEVELOPMENT OF A COMPUTIONAL INTERFACE TO ACCESS A SQL DATABASE OF AN INDUSTRIAL SUPERVISORY SYSTEM. 2020. 62 f. Course Completion Work (Undergraduate) – Electrical Engineering. Federal University Technological of Paraná. Cornélio Procópio, 2020.

This work proposed the creation of an alternative supervisory system of an industrial plant to collect an extract information from the main supervisory database, in order to make this information available on a remotely hosted server. To achieve this goal, a supervisory system were made, using NetBeans environment, to run on any machine that needs access to all available data, and other software were developed to run in the local machine to extract the data from the local to the remote database. In this work was possible to understand the concepts of data recording and reading, in a relational database hosted on a remote server.

Keywords: Supervisory Systems. Industrial Automation. Industry 4.0. Industrial Networks. Relational Databases.

LISTA DE QUADROS

LISTA DE FIGURAS

LISTA DE SIGLAS

GUI	*Graphical User Interface*
IDE	*Integrated Development Environment*
IP	*Internet Protocol*
RDBMS	*Relational Database Management System*
SQL	*Structured Query Language*
TCP	*Transmission Control Protocol*
URL	*Uniform Resource Locator*

SUMÁRIO

1 INTRODUÇÃO

Segundo Ribeiro (2011), a automação industrial começou a dar seus primeiros passos graças à possibilidade de criação de circuitos eletrônicos cada vez menores e com a capacidade de serem programáveis. Com isso, tornou-se possível alterar uma linha de produção a um custo e tempo muito baixos, apenas alterando a programação inserida nas máquinas, seja reescrevendo-a ou substituindo-a por outra, já que os códigos são armazenados em chips de memória, otimizando o processo por não necessitar de substituição ou aquisição de diversas ferramentas para processos específicos.

De acordo com Sommerville (2011), os sistemas de software são imprescindíveis para a sociedade moderna em qualquer lugar do planeta, estão diretamente ligados aos avanços em diversas áreas da indústria como a de infraestrutura e serviços, produção e produtos, entretenimento, etc, sendo muitas vezes os responsáveis por estes avanços. São sistemas tão presentes atualmente devido à engenharia responsável pelo seu desenvolvimento não possuir uma limitação física ou natural para o que possa ser feito, dando espaço para um rápido crescimento. Outro fator importante foi a expansão e desenvolvimento da Internet que teve grande impacto no desenvolvimento desses sistemas, fazendo com que seu custo de manutenção reduzisse devido à conexão com um servidor Web, onde o sistema poderia ser implantado, concedendo fácil acesso a correções pelo próprio usuário. Também é apontado que um sistema de software não é composto apenas pelo desenvolvimento de um programa em si ou por suas linhas de código que realizam as funções determinadas, mas também por toda a documentação necessária para a utilização do programa, como os requisitos de software e usuário; descrições de configuração mínima para hardwares; descrição de banco de dados e sua organização lógica e relacional.

Conforme Silberchatz (2006), um sistema de gerenciamento de banco de dados (DBMS) é uma coleção de dados que representam informações importantes e os programas necessários para o acesso destas informações. São sistemas que gerenciam as informações para facilitar a sua recuperação, por meio de técnicas de computação, de forma rápida e precisa, e podem ser aplicados em diversas áreas como bancos, linhas aéreas, universidades, transações de cartão de crédito, telecomunicação, finanças, vendas, revendedores online, recursos humanos e

indústria. Diversos softwares utilizam estes sistemas para alocação segura das informações em um banco de dados, onde o usuário é capaz de realizar várias atividades, como as citadas anteriormente, por meio de interfaces que tornam tais atividades também disponíveis online.

Com o crescimento dos dispositivos conectados à rede e maior conectividade dos processos nas indústrias, além da evolução dos sistemas de software e interfaces para facilitar este tipo de conexão pelos usuários, este trabalho propõe o desenvolvimento de uma interface para exportação e manipulação dos dados, em um banco de um Sistema Supervisório com acesso restrito à máquina local, ou seja, sem acesso à rede, para um banco de dados hospedado em um servidor online, onde será possível a visualização e controle dos processos industriais da planta fora do local de trabalho.

Desta forma, o presente trabalho propõe a possibilidade de construção de um sistema supervisório para visualização, controle e manipulação, parcial ou completa, do banco de dados da Planta Didática SMAR, presente na sala G-105 da UTFPR-CP, em qualquer computador com sistema operacional *Windows* e acesso à internet. Mas, como controlar e gerenciar um sistema supervisório remotamente via Internet (online), cujo acesso é limitado à máquina local? Esta é a pergunta que será respondida neste trabalho.

1.1 Objetivo geral

Desenvolver uma interface gráfica do utilizador (GUI) integrada a um sistema de banco de dados relacional hospedado em um servidor online, para controle e gerenciamento remoto de uma planta industrial.

1.2 Objetivos específicos

As seguintes tarefas foram realizadas para alcance do objetivo proposto neste trabalho:

Estudar todos os processos da Planta Didática SMAR.

Estudar o sistema de banco de dados relacional do Sistema Provisório SMAR para compreender de que forma e onde os dados são alocados.

Desenvolver a interface gráfica do utilizador primária semelhante ao Sistema Supervisório SMAR, na linguagem Java com auxílio do NetBeans IDE, ambiente de desenvolvimento gratuito e open source, na qual acessará, de qualquer máquina que possua a interface instalada, os dados no banco de dados relacional MySQL, hospedado em um servidor online contratado pelo serviço de hospedagem web Hostinger.

Desenvolver, de maneira semelhante à anterior, a interface gráfica do utilizador secundária para acompanhamento da extração dos dados do banco de dados relacional Microsoft SQL Server da Planta Didática SMAR, para disponibilizá-los no servidor online para leitura pela interface primária.

Construir o banco de dados relacional onde serão alocados os dados provenientes do banco de dados da Planta Didática SMAR, com auxílio do MySQL Workbench, ambiente de desenvolvimento gratuito com ferramentas para visualização, desenvolvimento e administração de bancos de dados relacionais por meio da linguagem de programação SQL.

1.3 Justificativa

A possibilidade de controlar remotamente uma planta industrial, ou seja, de fora de seu local de serviço, para obtenção de maior conectividade e acessibilidade dos processos industriais, não limitando o controle e a supervisão da planta ao ambiente fabril.

Neste trabalho é utilizada a estrutura de banco de dados MySQL devido à sua popularidade e acessibilidade por diversos serviços de hospedagem. Para manipulação dos dados será utilizada a interface MySQL Workbench para facilitar o acesso à estrutura dos dados. Além disso, para a extração dos dados na máquina local e alocação em outro servidor será utilizada a linguagem de programação Java com o auxílio do ambiente de desenvolvimento NetBeans IDE, devido à facilidade de programação com ferramentas de arraste para funções como botões, caixas de texto, caixas de marcação, entre outras ferramentas essenciais para a programação de uma interface gráfica do usuário.

1.4 Estrutura de capítulos

O Capítulo 2 é composto por uma revisão bibliográfica dos conceitos, métodos e ferramentas utilizadas. O Capítulo 3 mostra a metodologia para a criação do sistema supervisório e do banco de dados. O Capítulo 4 mostra alguns resultados preliminares de um sistema simplificado. O Capítulo 5 traz algumas considerações.

2 FUNDAMENTAÇÃO TEÓRICA

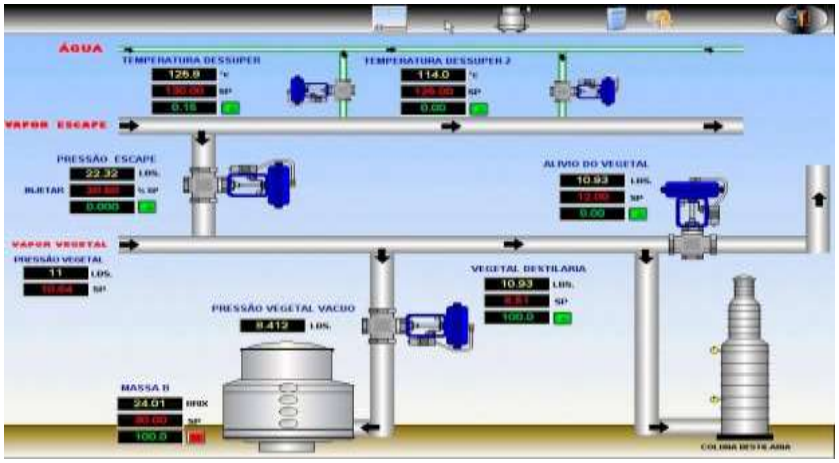
Neste capítulo serão abordados os conceitos teóricos para embasamento do trabalho.

2.1 Sistemas supervisórios

Martins (2012) diz que a automação teve seu início para facilitar nossas vidas em nossas atividades humanas em seus mais variados aspectos, desde as residências, ruas e trabalho até os meios de produção em diversas escalas. Tal evolução na automação industrial aumentou muito a complexidade e a necessidade de controle e monitoramento dos processos industriais, levando ao surgimento dos sistemas supervisórios, que são programas que ilustram de maneira mais simples e intuitiva o comportamento de um processo, desviando dos códigos de controle que necessitam de maiores conhecimentos para serem compreendidos por leigos. Tais tipos de sistemas são tão essenciais atualmente por ser possível ter todas as informações e todo o controle em um único local, agregando maior segurança e confiabilidade em todo o processo. Na Figura 1 é possível observar um exemplo de um sistema supervisório.

Os sistemas supervisórios necessitam de alguns *drivers* para possibilitar a comunicação entre os meios analógicos e reais para os meios digitais, um banco de dados para armazenagem das informações de todos os sensores e uma interface gráfica para que o usuário seja capaz de compreender o que está sendo transmitido de forma clara e concisa.

Figura 1 – Exemplo de um sistema supervisório



Fonte: site Branqs Automação

2.2 Banco de dados

De acordo com Takai, Italiano e Ferreira (2005), um banco de dados é um conjunto ou uma coleção de informações que são gravadas digitalmente em um computador de modo lógico, podendo ser acessadas por Sistemas Gerenciadores de Banco de Dados (RDBMS). Tais sistemas tiveram seu início no fim da década de 60, quando foi criado o primeiro RDBMS. Desde então, diversos outros modelos surgiram para otimizar o acesso seja de gravação, extração ou manipulação das informações armazenada. O modelo mais utilizado atualmente é o modelo relacional, entretanto existem alguns outros como hierárquico, em redes e orientado a objetos.

Em um modelo relacional as informações são armazenadas em tabelas que podem possuir “n” atributos, que são o que representará as informações dos conjuntos de dados dispostos nesta tabela, podendo também ser relacionados entre outras tabelas por meio de variáveis definidas como chaves primárias, que são variáveis estáticas únicas para cada registro, como é mostrado no exemplo da Figura 2. No exemplo são mostradas informações de contas bancárias, onde é possível observar três tabelas que podem ser relacionadas de acordo com o código do cliente e número da conta corrente, sendo possível obter demais informações como nome do cliente, saldo, rua e cidade com apenas uma das informações dentre código e número. Uma vez que é sabido o código, uma das tabelas conecta-o ao número da conta corrente, podendo ligá-lo à informação do saldo.

Figura 2 – Exemplo de modelo relacional de dados

Cod_Cliente	Nome	Rua	Cidade
1	Pedro	A	São Paulo
2	Maria	B	Jundiai

Num_CC	Saldo
20121	1200
21582	1320
21352	652

Cod_Cliente	Num_CC
1	20121
2	21582
2	21352

Fonte: TAKAI; ITALIANO; FERREIRA, 2005

Em um software industrial, como os sistemas supervisórios, é possível acessar estes dados alocados em um servidor de banco de dados de diversos pontos, caso possua as permissões e credenciais necessárias, podendo compartilhar e manipular tais informações entre diversos pontos.

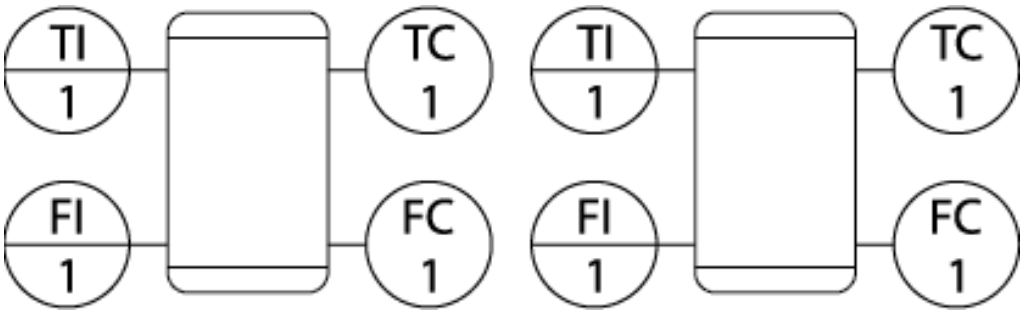
2.3 Programação orientada a objetos

A construção de softwares modernos requer o emprego de técnicas especializadas para garantir confiabilidade, disponibilidade, desempenho, segurança, agilidade e, segundo Rubira (2009), o modelo de objetos apresenta-se promissor nestes aspectos. Neste tipo de programação, as estruturas de processos podem ser assemelhadas como objetos do mundo real que possuem interação entre si. Dentre as linguagens mais comuns estão C#, C++ e Java, lançada em 1995 como uma linguagem puramente orientada a objetos, sucedendo a linguagem C++.

3 METODOLOGIA

Neste Capítulo é mostrado como pode ser desenvolvida a aplicação, de maneira simplificada, nas ferramentas MySQL Workbench e NetBeans IDE baseando-se na construção de um modelo genérico, abordando todas as etapas do processo de criação. O modelo a ser construído serão dois tanques com dois controladores de temperatura e vazão.

Figura 3 – Modelo de simulação



Fonte: Autoria Própria

De modo geral, os passos a serem seguidos para a construção da interface e estruturação do servidor são os seguintes:

1. Construção da interface gráfica;
2. Programação das funcionalidades da interface gráfica;
3. Escolha e contratação do serviço de hospedagem web;
4. Estruturação do servidor para comunicação com a interface gráfica.

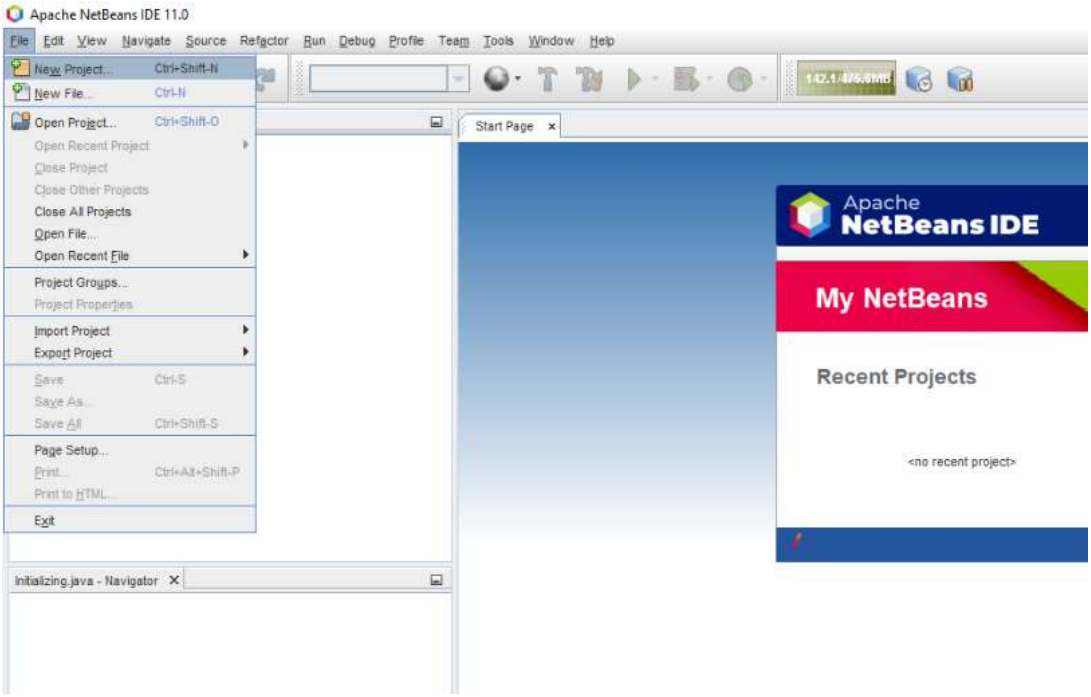
3.1 Construção das interfaces gráficas

Neste tópico será abordado o desenvolvimento das interfaces gráficas para interação com os dados do servidor.

3.1.1 Criando um projeto no *Netbeans IDE*

Ao instalar e executar o Netbeans pela primeira vez, a tela inicial irá aparecer com a *Start Page* aberta no painel central e o painel esquerdo de projetos vazio. Para criar um novo projeto basta ir em *File,* no menu superior, e selecionar *New Project* (Ctrl+Shift-N).

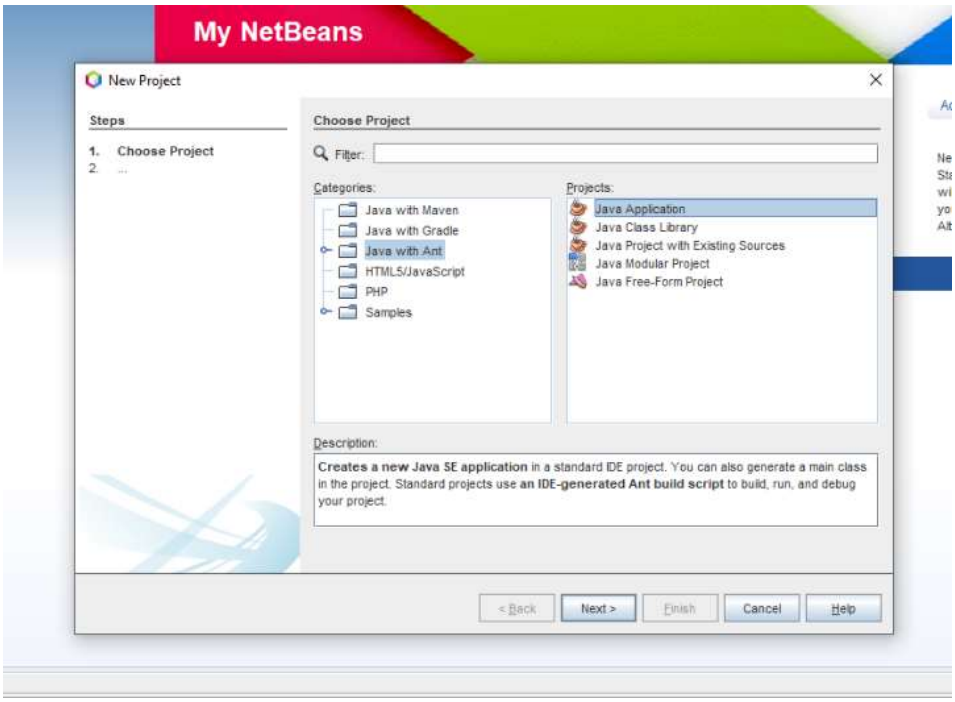
Figura 4 – Criação de novo projeto no *Netbeans IDE*



Fonte: Autoria Própria

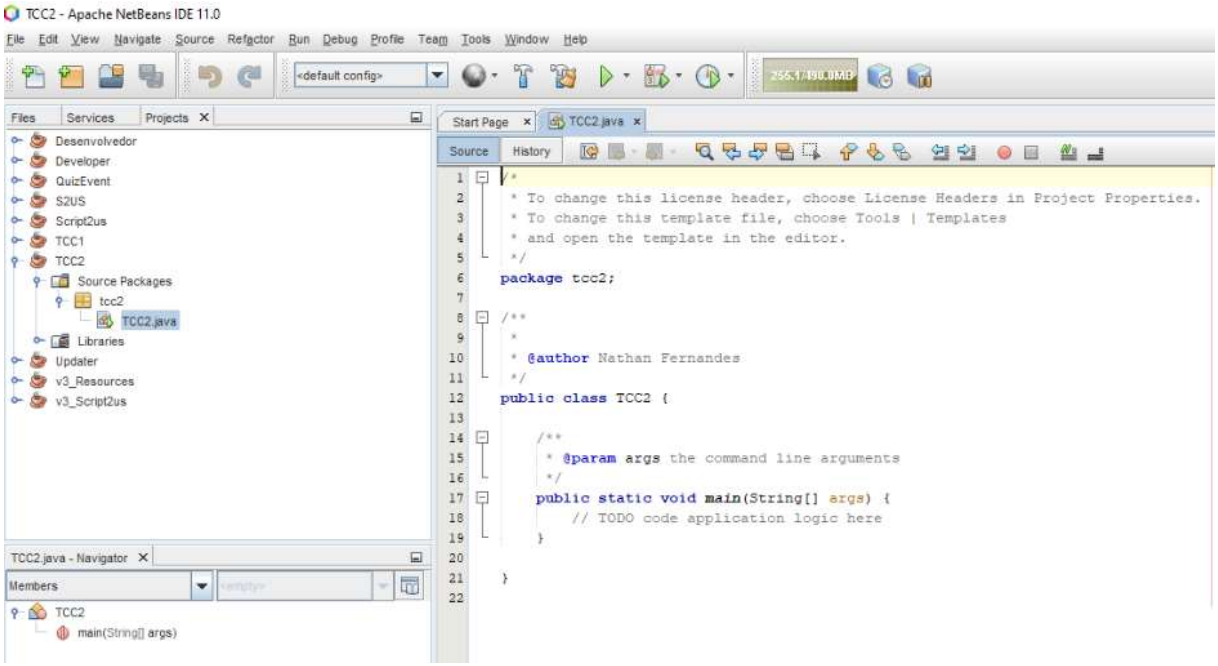
Para a criação do novo projeto, é necessário definir qual será a linguagem utilizada. Neste projeto será utilizada a categoria *Java with Ant* e o projeto será *Java Application*. Após escolher as opções, o projeto será aberto com a classe *Main* visível, que é a classe que será executada quando o projeto for compilado.

Figura 5 – Opções de novo projeto no *Netbeans IDE*



Fonte: Autoria Própria

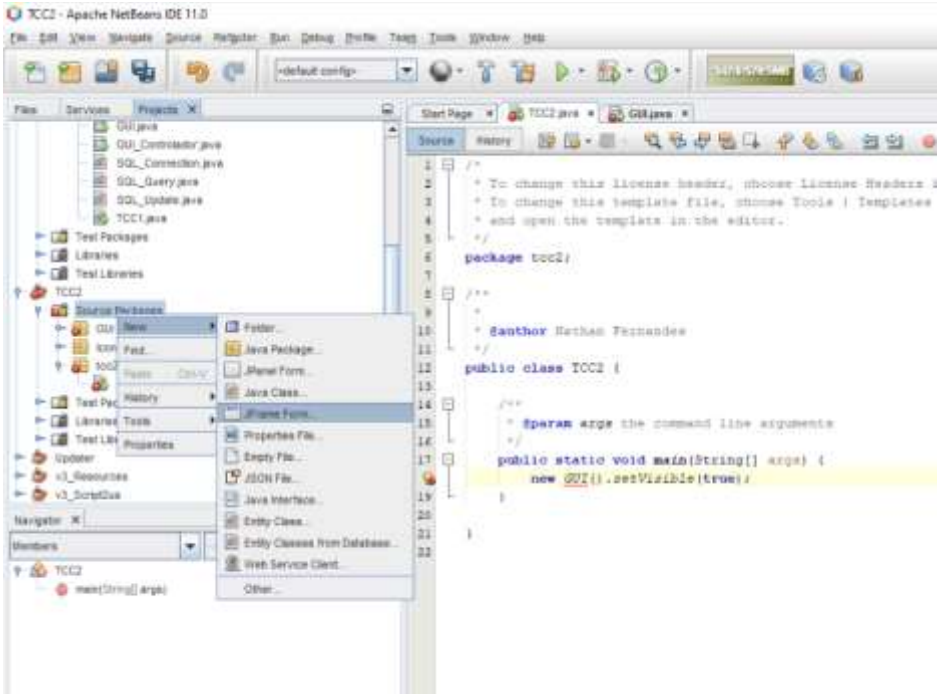
Figura 6 – Novo projeto criado no *Netbeans IDE*



Fonte: Autoria Própria

3.1.2 Criando uma interface gráfica no *Netbeans IDE*

Figura 7 – Criação de formulário no *Netbeans IDE*



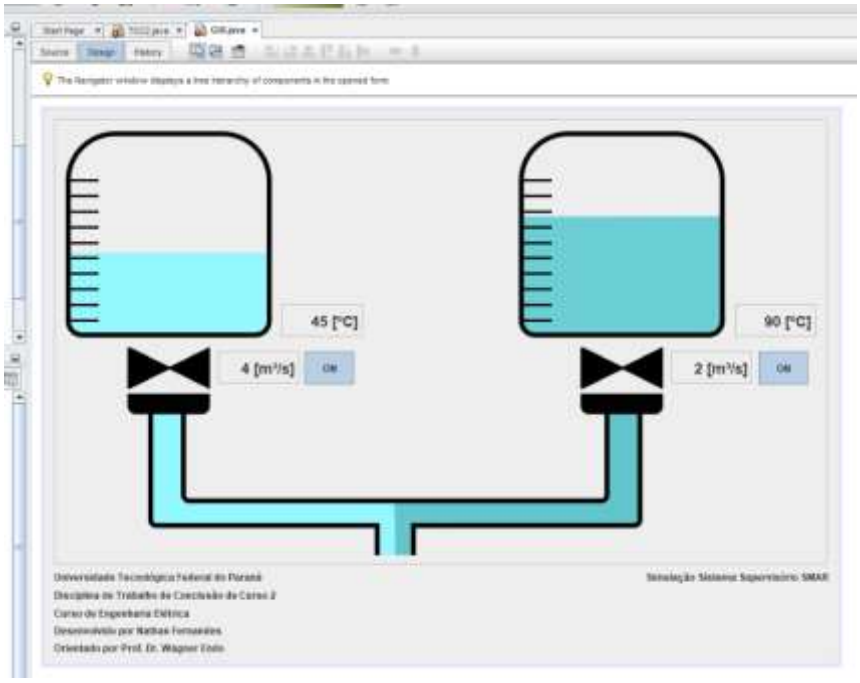
Fonte: Autoria Própria

Com o projeto criado, serão construídas novas classes para conter os códigos que farão a comunicação com o servidor, lendo e escrevendo entradas no banco de dados relacional; novos formulários que serão as interfaces para alocar as

ferramentas de visualização e comandos da aplicação, como caixas de texto, caixas de seleção, botões, menus, etc.

A primeira interface criada será a interface para registro dos dados no banco *SQL* e visualização das informações, que será a representação do sistema supervisório SMAR, recebendo as informações da planta e registrando-as no servidor local. Dessa forma, será construído um formulário *JFrame* nomeado *GUI1.java*, como mostrado na Figura 8, onde serão alocadas todas as funcionalidades e que será chamado pela classe principal *TCC2.java*, como mostrado na Figura 7.

Figura 8 – Interface para registro e controle da planta

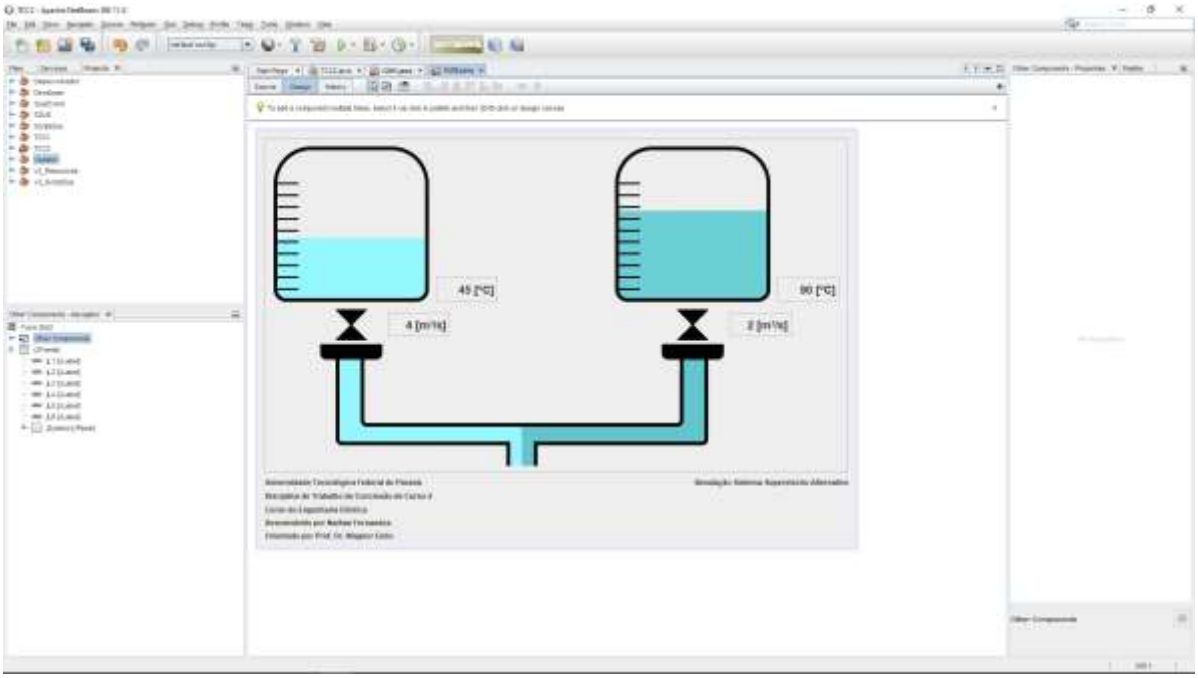


Fonte: Autoria Própria

Para construção da interface, basta mover os controles da paleta de controles para o projeto. Na interface mostrada na Figura 8, foram utilizados *Labels* para as imagens e textos, *Toogle Buttons* para abrir/fechar as válvulas e *Panels* para alocar os controles. As imagens contidas na interface não fazem parte da biblioteca do Netbeans e, portanto, foram todas desenhadas manualmente em um software de edição de imagem.

A segunda interface será uma cópia exata da primeira, porém os botões de controle serão removidos. Desta forma, essa interface será somente para realizar a leitura dos dados no banco SQL e visualização das informações, podendo ser acessada de qualquer local.

Figura 9 – Interface para leitura das informações da planta

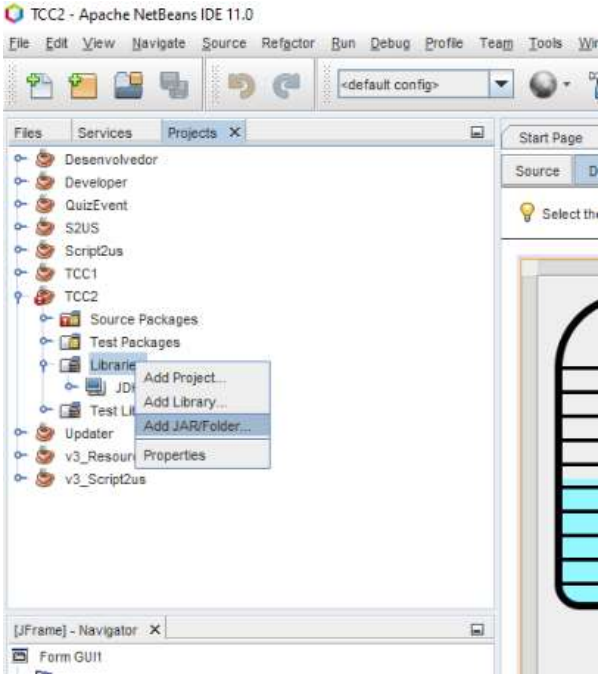


Fonte: Autoria Própria

3.1.3 Atribuindo funções para uma interface gráfica no *Netbeans IDE*

Com as interfaces gráficas construídas, é necessário escrever as funções que realizarão a conexão, registro e leitura dos dados no banco *SQL*. Essa etapa será realizada antes da estruturação do banco, uma vez que ele poderá ser construído posteriormente, mas respeitando as sintaxes utilizadas nestas funções.

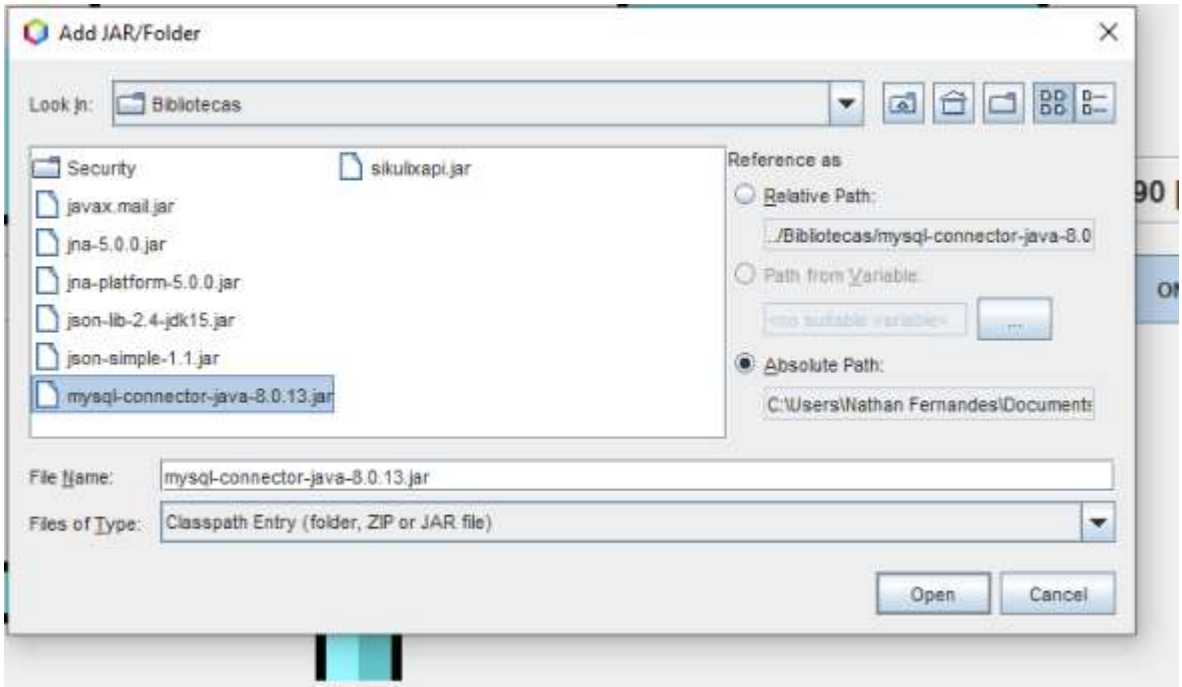
Figura 10 – Importando nova biblioteca para o projeto



Fonte: Autoria Própria

Para construção das funções que realizam a conexão, registro e leitura das informações no banco de dados, é necessário importar uma biblioteca específica com as sintaxes responsáveis por esse tipo de comunicação. Neste caso, a conexão com o servidor será realizada de modo direto, ou seja, a interface acessará o servidor diretamente, sem a necessidade de utilizar um serviço web (como PHP) para intermediar o acesso, que é uma prática mais segura. A biblioteca a ser utilizada para realizar a conexão é a *MySQL Connector Java 8.0.13*. Ela deverá ser baixada e importada na pasta *Libraries*, que está logo abaixo dos *Source Packages* no projeto *NetBeans*. As Figuras 10 e 11 mostram como é feita a importação de uma biblioteca para o projeto.

Figura 11 – Selecionando biblioteca para importação



Fonte: Autoria Própria

Para a interface 1, que realiza os registros no servidor e simula a ação do supervisório da planta, serão necessárias as seguintes funções:

1. Atualização do display de temperatura do tanque 1;
2. Atualização do display de temperatura do tanque 2;
3. Atualização da imagem com o volume de líquido presente no tanque 1;
4. Atualização da imagem com o volume de líquido presente no tanque 2;
5. Atualização do display de vazão da válvula 1 ao pressionar o botão para abrir a válvula 1;

6. Atualização do display de vazão da válvula 2 ao pressionar o botão para abrir a válvula 2;
7. Atualização da imagem da tubulação de acordo com as válvulas abertas;
8. Reabastecimento automático do líquido dos tanques sob certas e possíveis condições como válvula fechada e tanque vazio;
9. Reajuste da temperatura de acordo com o estado do controlador de temperatura;
10. Conexão com o banco de dados SQL;
11. Atualização dos valores das colunas com os registros de temperatura 1, temperatura 2, vazão 1 e vazão 2 na tabela de informações do banco SQL.

Para a interface 2, que realiza a leitura dos dados no servidor e simula um supervisório qualquer para acesso remoto, serão necessárias as seguintes funções:

1. Atualização do display de temperatura do tanque 1;
2. Atualização do display de temperatura do tanque 2;
3. Atualização da imagem com o volume de líquido presente no tanque 1;
4. Atualização da imagem com o volume de líquido presente no tanque 2;
5. Atualização do display de vazão da válvula 1;
6. Atualização do display de vazão da válvula 2;
7. Atualização da imagem da tubulação de acordo com as válvulas abertas;
8. Conexão com o banco de dados SQL;
9. Leitura dos valores das colunas com os registros de temperatura 1, temperatura 2, vazão 1 e vazão 2 na tabela de informações do banco SQL.

Como o sistema 1 será uma simulação de um supervisório, ele possuirá as seguintes condições:

- Os tanques irão perder 1ºC por segundo, com os controladores de temperatura desligados, até atingirem a temperatura de 25ºC;
- O tanque 1 irá ser aquecido 1°C por segundo, com o controlador de temperatura ligado, até atingir a temperatura de 45°C e manterá esta temperatura enquanto estiver ligado;

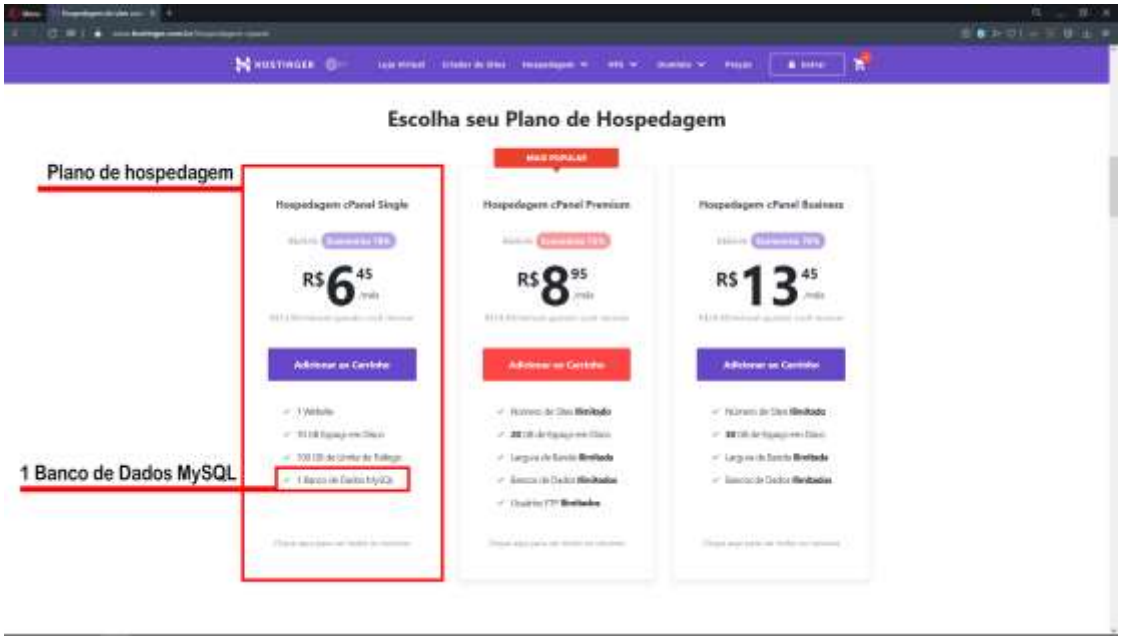
- O tanque 2 irá ser aquecido 1°C por segundo com o controlador de temperatura ligado, até atingir a temperatura de 90°C e manterá esta temperatura enquanto estiver ligado;
- Os tanques serão abastecidos a uma vazão de 1 m³/s até atingirem o limite de 100 m³ (limite da graduação nos tanques);
- Os tanques serão esvaziados a uma vazão definida pelo usuário, por meio dos botões de aumentar ou diminuir a vazão do tanque, podendo ser de no máximo 10 m³/s.

Todas as classes e funções criadas estão detalhadas no Anexo A e B deste trabalho.

3.2 Hospedagem Web

O serviço de hospedagem web é como alugar um computador para armazenar as informações pertinentes à sua aplicação. Nele é possível armazenar todos os dados como textos, imagens e vídeos de modo que seja possível acessá-los de qualquer parte do mundo por meio de um domínio registrado (site), que é o endereço de onde os dados estão hospedados. Na Figura 12 há um exemplo de um serviço de hospedagem com suporte para 1 banco de dados MySQL.

Figura 12 – Planos de hospedagem *Hostinger*

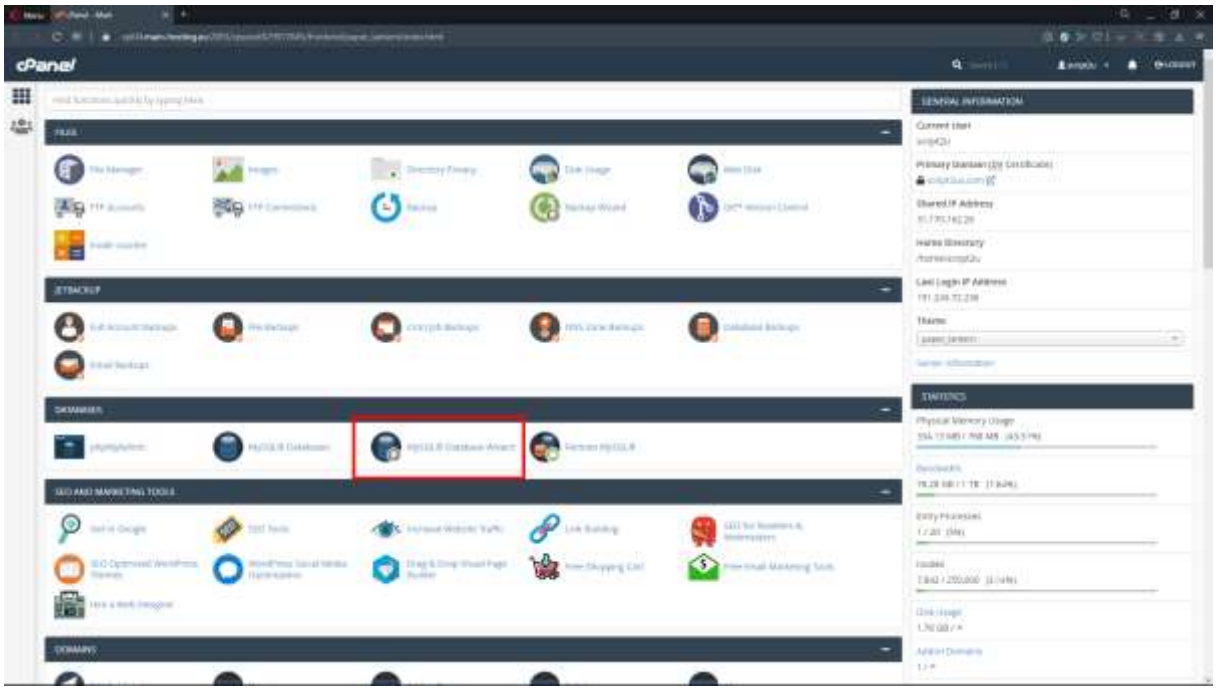


Fonte: Site da Hostinger

Existem diversos serviços de hospedagem web como *HostGator, GoDaddy, Hostinger, UOL Host, Google Cloud,* dentre outros. Para a construção do servidor SQL, é necessário contratar um serviço de hospedagem que ofereça suporte e acesso a tal função.

Após selecionar o serviço de hospedagem, o plano de hospedagem, realizar o pagamento e acessar a conta criada, deve ser acessado o *CPanel,* que é o painel de controle da hospedagem. Nele é possível realizar todas as ações disponíveis para sua hospedagem como criação de website, criação e gerenciamento de servidores ftp, criação e gerenciamento de e-mails, dentre diversas outras funções.

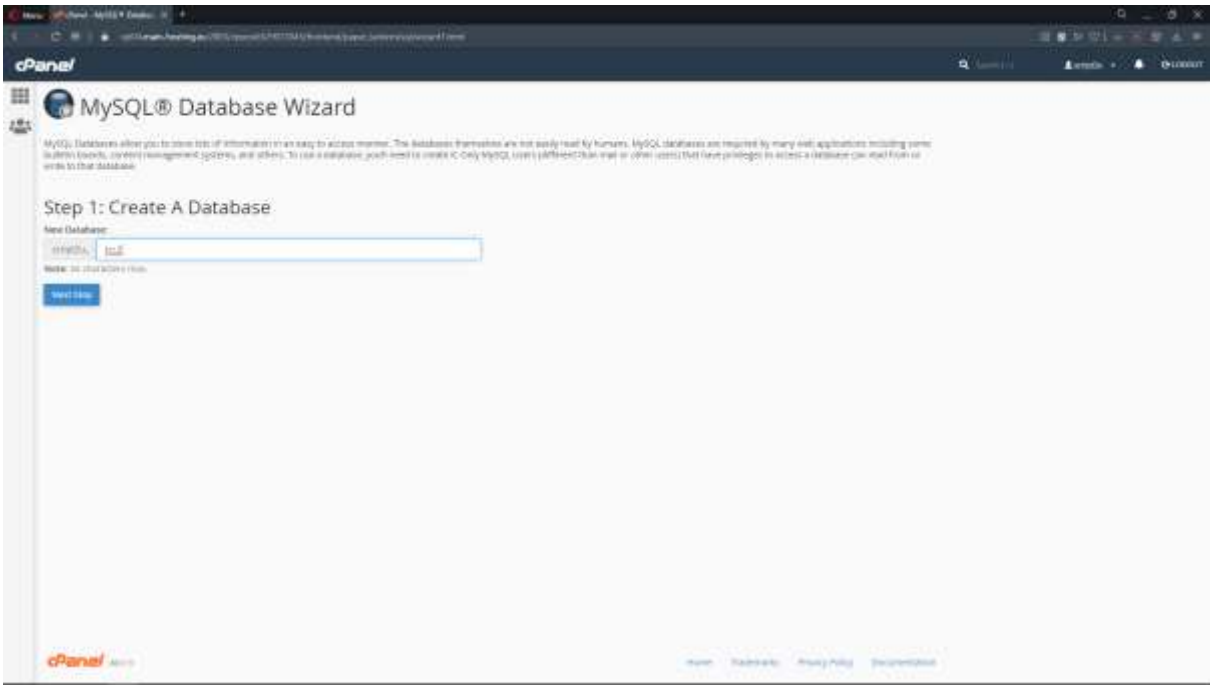
Figura 13 – CPanel e MySQL Database Wizard



Fonte: CPanel Hostinger

Para criar um novo servidor *SQL* no painel *CPanel*, é preciso clicar na opção *MySQL Database Wizard*, mostrada na Figura 13, que irá redirecionar para uma janela onde será iniciada a criação do banco de dados de forma simplificada. O primeiro passo pode ser visto na Figura 14, que é a escolha de um nome para o banco.

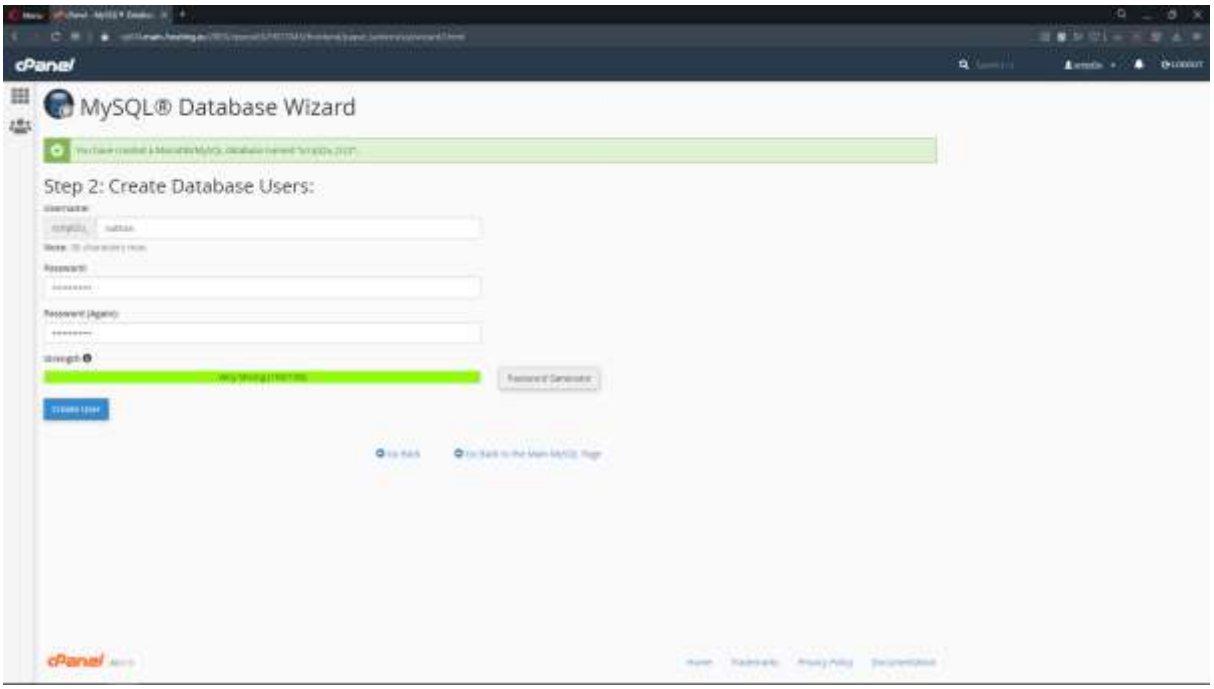
Figura 14 – Criação de banco de dados no CPanel



Fonte: CPanel Hostinger

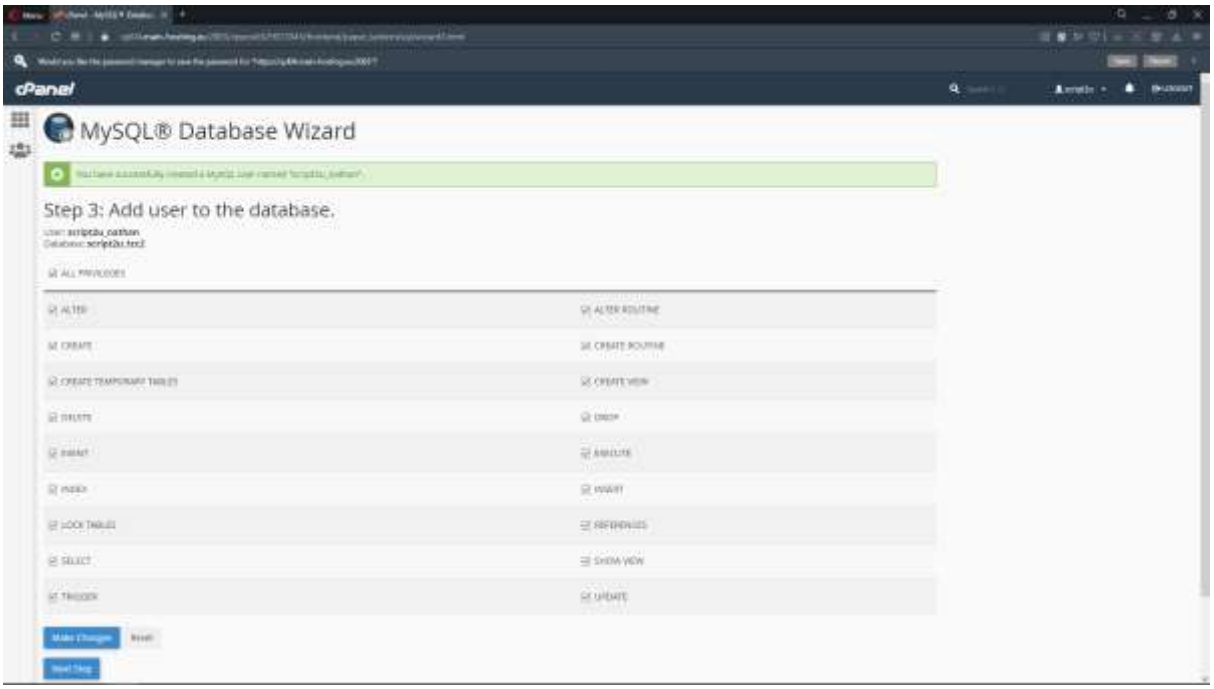
O banco de dados deverá possuir seu próprio nome e também um usuário e uma senha com os privilégios necessários para leitura e registro dos dados, como é possível observar na Figura 15 e na Figura 16.

Figura 15 – Criação de usuário para acesso ao banco de dados



Fonte: CPanel Hostinger

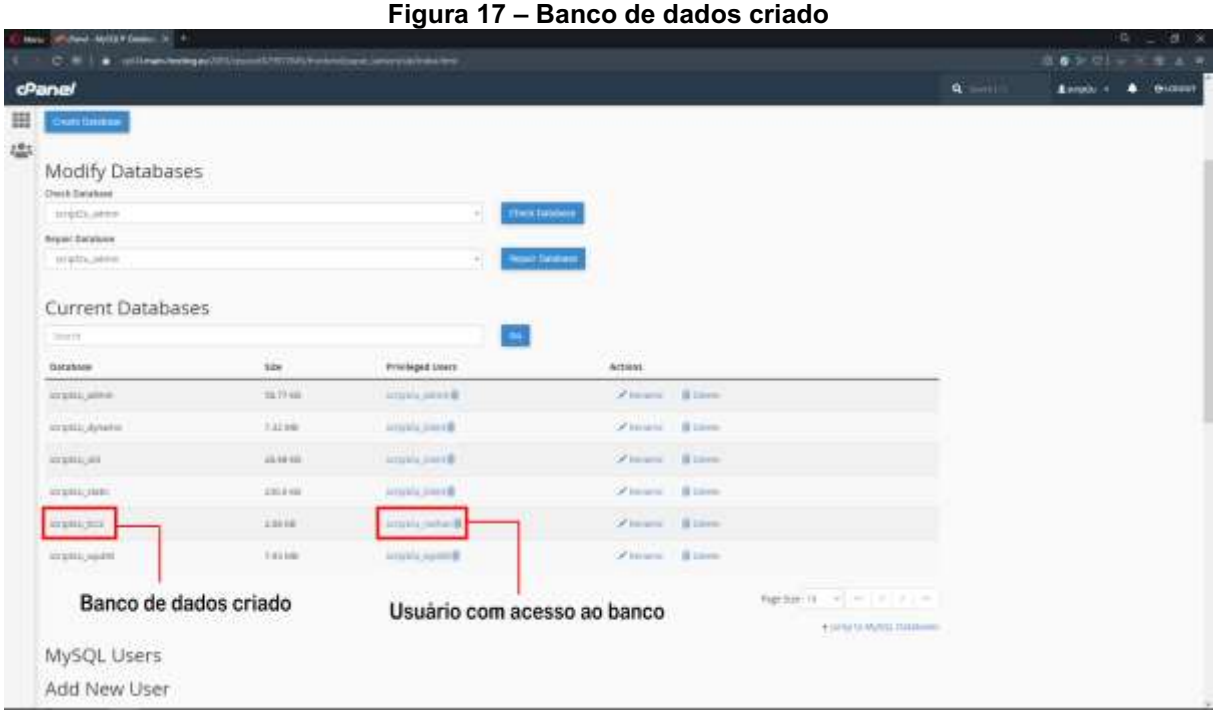
Figura 16 – Atribuição de privilégio ao usuário do banco de dados



Fonte: CPanel Hostinger

O banco a ser criado possui o nome *script2u_tcc2*, como mostrado na Figura 17. O usuário que será concedido acesso ao banco será o *script2u_nathan*, que terá todos os privilégios disponíveis para gerenciamento dos dados.

Figura 17 – Banco de dados criado



Fonte: CPanel Hostinger

3.3 MySQL Workbench

Neste tópico será abordado o desenvolvimento na ferramenta MySQL Workbench.

3.3.1 Conexão ao banco de dados

O banco de dados a ser utilizado é um banco contratado pela empresa de hospedagem web *Hostinger.*

Quadro 1 – Parâmetros de configuração da conexão no *MySQL Workbench*

Parâmetro	Descrição
Connection Name	Nome fictício para a conexão ao *RDBMS*
Connection Method	Método de conexão ao *RDBMS*
Hostname	Endereço *URL* ou *IP* do servidor
Port	Porta do protocolo *TCP*/*IP*
Username	Nome do usuário do *RDBMS*
Password	Senha do usuário do *RDBMS*

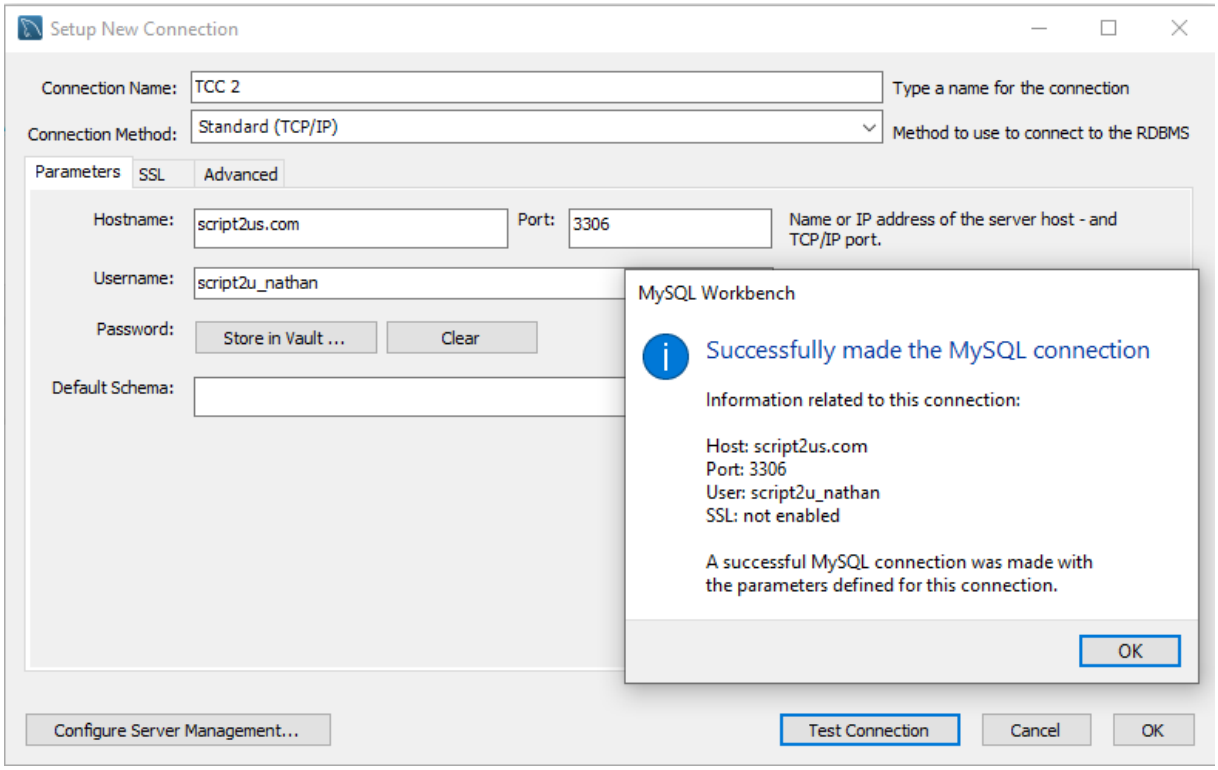
Fonte: Autoria Própria

Para acessar o servidor, é preciso ter todas as credenciais que, neste caso, são as seguintes:

- IP do servidor: script2us.com
- Nome do usuário: script2u_nathan
- Senha do usuário: -m-meB_IN-3l

As informações do banco contratado, disponíveis na Figura 18, serão utilizadas para acessá-lo pelo *MySQL Workbench*, para construção das tabelas onde serão armazenados os dados coletados.

Figura 18 – Criação de nova conexão ao banco de dados

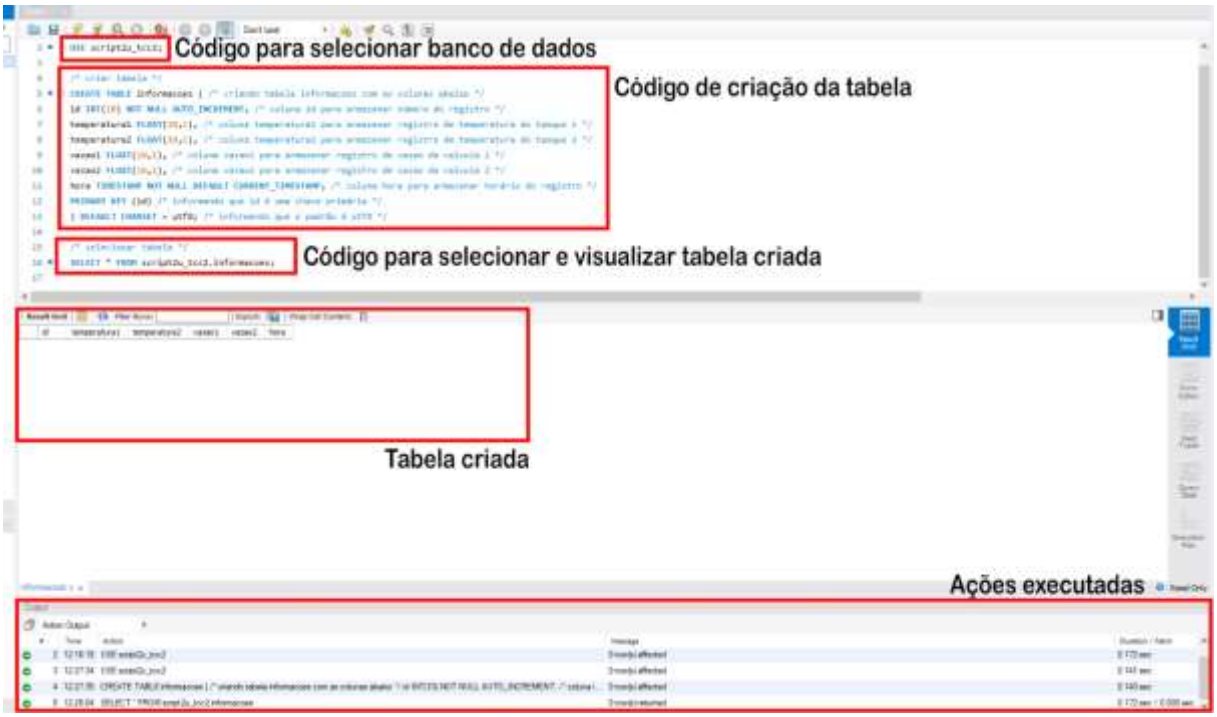


Fonte: Autoria Própria

3.3.2 Modelagem do banco de dados

Para a modelagem do banco, serão utilizadas sintaxes SQL para criação da tabela onde serão armazenados os dados relacionados aos instrumentos de medição, de acordo com o modelo proposto, como é mostrado na Figura 19. Nele, é criada apenas uma única tabela que conterá as informações de temperatura e vazão dos tanques e válvulas 1 e 2, e também o horário do registro. O código utilizado para a construção das tabelas é encontrado no Anexo A.

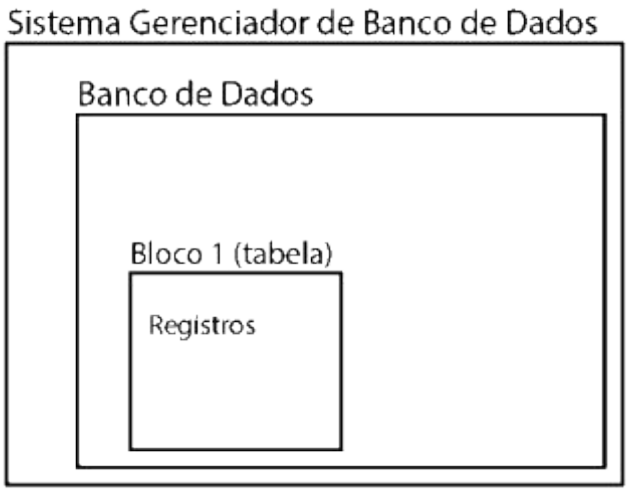
Figura 19 – Sintaxes SQL para construção do banco de dados



Fonte: Autoria Própria

A Figura 20 mostra um esquemático de como ficará a estrutura do banco de dados para o sistema simplificado.

Figura 20 – Esquemático do banco de dados simplificado



Fonte: Autoria Própria

4 ANÁLISE DE RESULTADOS

Com a simulação do sistema foi possível observar as alterações e registros dos valores das variáveis no banco de dados, a cada segundo, feitos pelo sistema 1, por meio da pesquisa pela tabela *registros* no MySQL Workbench. A Figura 21 mostra todos os registros feitos em um período de alguns segundos, onde a coluna *id* representa o índice ou número de registro da medição; a coluna *hora* representa o *timestamp* da medição, ou seja, a data e horário do registro; as colunas *temperatura1* e *temperatura2* representam a medição de temperatura dos tanques 1 e 2, respectivamente; as colunas *tc1* e *tc2* representam a chave de controle de temperatura, sendo 1 para *ON* e 0 para *OFF*; as colunas *volume1* e *volume2* representam o volume de líquido contido dentro dos tanques 1 e 2 e as colunas *vazao1* e *vazao2* representam a velocidade de vazão do líquido pelas válvulas 1 e 2.

Figura 21 – Dados no servidor

id	hora	temperatura1	tc1	temperatura2	tc2	volume1	vazao1	volume2	vazao2
148	2020-07-22 01:21:41	45.0	1	90.0	1	74.0	3.0	99.0	1.0
149	2020-07-22 01:21:43	45.0	1	90.0	1	72.0	3.0	99.0	0.0
150	2020-07-22 01:21:44	45.0	1	90.0	1	70.0	3.0	100.0	0.0
151	2020-07-22 01:21:45	45.0	1	90.0	1	68.0	3.0	100.0	0.0
152	2020-07-22 01:21:47	45.0	1	90.0	1	66.0	3.0	100.0	0.0
153	2020-07-22 01:21:48	45.0	1	90.0	1	64.0	3.0	100.0	0.0
154	2020-07-22 01:21:50	45.0	1	90.0	1	62.0	3.0	100.0	0.0
155	2020-07-22 01:21:51	44.0	0	90.0	1	60.0	3.0	100.0	0.0
156	2020-07-22 01:21:52	43.0	0	90.0	1	58.0	3.0	100.0	0.0
157	2020-07-22 01:21:54	42.0	0	90.0	1	56.0	3.0	100.0	0.0
158	2020-07-22 01:21:55	41.0	0	90.0	1	54.0	1.0	100.0	0.0
159	2020-07-22 01:21:56	40.0	0	90.0	1	55.0	0.0	100.0	0.0
160	2020-07-22 01:21:58	39.0	0	90.0	1	56.0	0.0	100.0	0.0
161	2020-07-22 01:21:59	40.0	1	90.0	1	57.0	0.0	100.0	1.0
162	2020-07-22 01:22:00	41.0	1	89.0	0	58.0	0.0	100.0	1.0
163	2020-07-22 01:22:02	42.0	1	90.0	1	59.0	0.0	100.0	1.0

Fonte: Autoria Própria

Note que os registros não foram feitos precisamente a cada segundo, isso se deve ao fato do código utilizado para o registro não ter enviado um valor de *timestamp* do momento em que a solicitação foi feita e, quando um novo dado é inserido sem um valor específico, é utilizado o valor padrão, que neste caso é o horário que o banco completou o registro das medições no banco de dados e não o horário que recebeu a solicitação de registro.

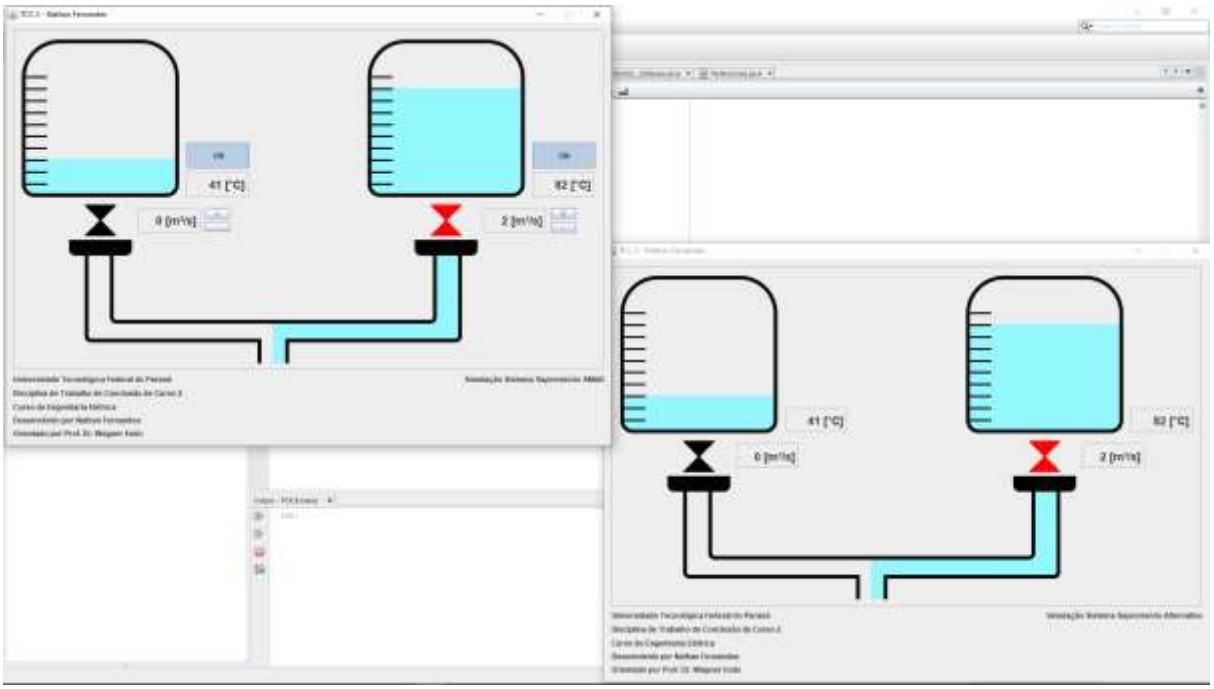
Como as variáveis foram simuladas, houve a necessidade de decrescer a temperatura a cada segundo até um limite definido em código, para simular a regulação térmica do líquido no tanque para a temperatura ambiente. O mesmo foi realizado com os valores de volume, já que o tanque deve ser preenchido para que

se esvazie por completo, acabando com a simulação. Sendo assim, foi definido que a temperatura é decrescida de 1°C por segundo quando o controle de temperatura está desligado e acrescida de 1°C quando o controle está ligado, já o volume do tanque é acrescido de 1 m³/s independente da válvula estar aberta ou não.

Dada as circunstâncias simuladas de temperatura e vazão, note que, quando o controle de temperatura é alterado de 1 para 0, a medição de temperatura é decrescida de 1°C, como é possível observar nos registros de *id 154* e *id 155* para a temperatura 1. Já para o volume nos tanques, como a vazão é sempre acrescida de 1 m³/s, independente do estado da válvula, até o limite de 100 m³ de cada tanque, a diferença entre um registro e o próximo será sempre o resultado do volume atual menos a vazão mais um. Veja que nos registros de vazão igual a 1 m³/s o volume permanece inalterado, enquanto que para vazões maiores que 1 m³/s o volume já passa a reduzir. Essas condições ocorreram apenas pelo fato do sistema 1 ter sido simulado, mas correspondem exatamente como foram programadas.

Enquanto isso, o sistema 2 foi capaz de ler o último dado registrado no servidor e atualizar os visualizadores.

Figura 22 – Sistemas de registro e leitura



Fonte: Autoria Própria

5 CONSIDERAÇÕES FINAIS

A princípio este estudo pretendia realizar a exportação dos dados de um Sistema Supervisório SMAR – uma planta didática com instrumentos de medição reais – para um servidor web, para que os dados pudessem ser lidos por qualquer outra aplicação em qualquer local geográfico, entretanto, devido às complicações como acessar o ambiente acadêmico neste período de pandemia e acessar os arquivos do Sistema SMAR sem o pendrive com sua chave de acesso, o trabalho se limitou a demonstrar como seriam realizadas as conexões e transferência dos dados, bem como a construção de todas as interfaces visuais e banco de dados.

No Sistema 1 simulou-se um supervisório, registrando dados fictícios, baseados nas condições e estados dos controladores, enquanto no Sistema 2 realizou-se somente a leitura do último registro no banco feito pelo Sistema 1.

O presente estudo teve como objetivo mostrar que é possível, de forma objetiva, realizar a exportação de informações para um banco hospedado em um servidor Web e realizar, posteriormente, a busca por estas informações de qualquer máquina conectada em qualquer rede, concedendo acesso e controle remoto à planta industrial, aumentando a sua conectividade.

Para trabalhos futuros, sugere-se o estudo da conexão e transmissão dos dados do supervisório para armazená-los em um ambiente fora da máquina local. Algumas dificuldades serão o bloqueio da conexão da universidade a alguns tipos de servidores externos e complexidade das informações do supervisório.

REFERÊNCIAS

COSTA, Elisângela Rocha. **Banco de Dados Relacionais**. Dissertação (Trabalho de Conclusão de Curso) – Faculdade de Tecnologia de São Paulo, 2011.

JACON, Pablo de Souza. **Estudo e Desenvolvimento de um Sistema de Supervisão de Processos Industriais Utilizando um Supervisório Open-Source e Protocolos de Comunicação Industriais Abertos.** Monografia (Trabalho de Conclusão de Curso) – Universidade Tecnológica Federal do Paraná, 2019.

KORTH, H. F.; SILBERSCHATZ, A.; SUDARSHAN, S. **Sistema de banco de dados.** 5ª edição – Rio de Janeiro: Elsevier, 2006.

MARTINS, Geomar Machado. **Princípios de Automação Industrial.** Universidade Federal de Santa Maria, Centro de Tecnologia – Santa Maria, 2012.

NATALE, F. **Automação Industrial.** São Paulo: Érica. 1996.

RIBEIRO, Marco Antônio. **Automação Industrial.** 4ª edição – Salvador, 2001.

RUBIRA, Cecília. **Introdução à Análise Orientada a Objetos e Projeto Arquitetural.** Universidade Estadual de Campinas, Instituto de Computação – Campinas, 2009.

SOMMERVILLE, Ian. **Engenharia de Software.** 9ª edição – São Paulo: Pearson Prentice Hall, 2011.

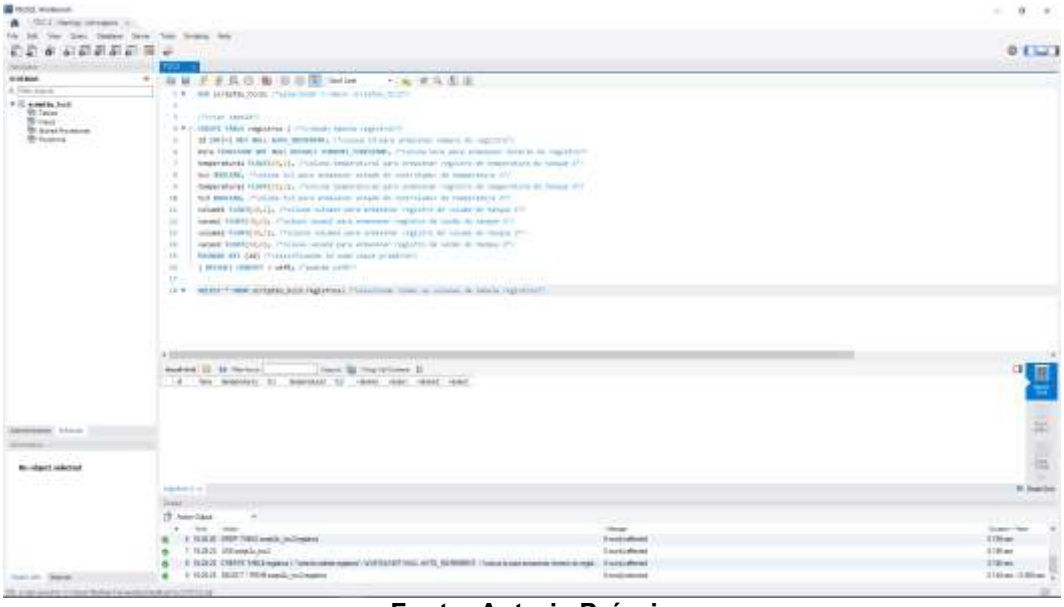
TAKAI, ITALIANO, FERREIRA, Osvaldo Kotaro, Isabel Cristina, João Eduardo. **Introdução a Banco de Dados.** DCC-IME-USP, 2005.

ANEXO A

CÓDIGO PARA CRIAÇÃO DA TABELA NO MYSQL

```
USE script2u_tcc2; /*selecionar o banco script2u_tcc2*/
CREATE TABLE registros ( /*criando tabela registros*/
id INT(4) NOT NULL AUTO_INCREMENT, /*coluna id para armazenar número do registro*/
hora TIMESTAMP NOT NULL DEFAULT CURRENT_TIMESTAMP, /*coluna hora para armazenar horário do registro*/
temperatura1 FLOAT(10,1), /*coluna temperatura1 para armazenar registro de temperatura do tanque 1*/
tc1 BOOLEAN, /*coluna tc1 para armazenar estado do controlador de temperatura 1*/
temperatura2 FLOAT(10,1), /*coluna temperatura2 para armazenar registro de temperatura do tanque 2*/
tc2 BOOLEAN, /*coluna tc2 para armazenar estado do controlador de temperatura 2*/
volume1 FLOAT(10,1), /*coluna volume1 para armazenar registro de volume do tanque 1*/
vazao1 FLOAT(10,1), /*coluna vazao1 para armazenar registro de vazão do tanque 1*/
volume2 FLOAT(10,1), /*coluna volume2 para armazenar registro de volume do tanque 2*/
vazao2 FLOAT(10,1), /*coluna vazao2 para armazenar registro de vazão do tanque 2*/
PRIMARY KEY (id) /*classificando id como chave primária*/
) DEFAULT CHARSET = utf8; /*padrão utf8*/
SELECT * FROM script2u_tcc2.registros; /*selecionar todas as colunas da tabela registros*/
```

Figura 23 – Tabela sendo criada no MySQL Workbench



Fonte: Autoria Própria

ANEXO B

NETBEANS CÓDIGO CLASSE PRINCIPAL

Esta é a classe principal, denominada TCC2.java, ela é a classe que será chamada ao iniciar a aplicação. Nela é escrito para tornar visível as duas interfaces GUI1.java e GUI2.java, de modo a tornar visível ambas as interfaces.

```
 1 package tcc2;
 2
 3 import GUI.GUI1;
 4 import GUI.GUI2;
 5
 6 /**
 7 *
 8 * @author Nathan Fernandes
 9 */
10 public class TCC2 {
11
12 /**
13 * @param args the command line arguments
14 */
15 public static void main(String[] args) {
16 new GUI1().setVisible(true);
17 new GUI2().setVisible(true);
18 }
19
20 }
```

ANEXO C

NETBEANS SISTEMA 1

Este é o código de criação da interface do sistema 1, o qual simula instrumentos de medição registrando seus valores no servidor. Note que nas linhas 10 e 11 são criados os objetos *run* e *p*, que fazem referência às classes *MySQL_Sintaxes* e *References*. A primeira contém os códigos para registrar e ler os dados no servidor, enquanto que a segunda é apenas para realizar a comunicação das variáveis de controle pelas funções das interfaces, evitando a criação de variáveis desnecessárias e tornando o código mais enxuto.

A classe *updateViewer* foi construída para atualizar todos os objetos de visualização, para ser chamada sempre que ocorrer uma alteração nas variáveis medidas.

Na linha 399 do código é possível observar uma ação com um timer de 1000 ms sendo iniciada logo após a interface ser iniciada. Esse timer tem o objetivo de modificar os valores de acordo com o estado dos controladores e lógica programada, além de registrar tais valores no servidor ao terminar de calculá-los.

A partir da linha 452 até a linha 482 são iniciadas as configurações dos botões de aumentar e diminuir a velocidade de vazão do líquido nos tanques 1 e 2.

```
1 package GUI;
2
3 import Resources.MySQL_Sintaxes;
4 import Resources.References;
5 import java.util.Timer;
6 import java.util.TimerTask;
7
8 public class GUI1 extends javax.swing.JFrame {
9
10 private final MySQL_Sintaxes run = new MySQL_Sintaxes();
11 private References p = new References();
12
13 public GUI1() {
14 initComponents();
15 p = run.readDatabase();
16 }
17
18 private void updateViewer() {
19 jTemperatura1.setText(p.getTemperatura1()+" [°C] ");
20 jTemperatura2.setText(p.getTemperatura2()+" [°C] ");
21 jVazao1.setText(p.getVazao1()+" [m³/s] ");
22 jVazao2.setText(p.getVazao2()+" [m³/s] ");
23 if (p.getVazao1()==0) {
```

```
24 jValvula1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_off.png")));
25 } else {
26 jValvula1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_on.png")));
27 }
28 if (p.getVazao2()==0) {
29 jValvula2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_off.png")));
30 } else {
31 jValvula2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_on.png")));
32 }
33 if (p.getVazao1()==0 && p.getVazao2()==0) {
34 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo0.png")));
35 } else if (p.getVazao1()>0 && p.getVazao2()==0) {
36 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo1.png")));
37 } else if (p.getVazao1()==0 && p.getVazao2()>0) {
38 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo2.png")));
39 } else if (p.getVazao1()>0 && p.getVazao2()>0) {
40 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo3.png")));
41 }
42 if (p.getVolume1()<10) {
43 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque0.png")));
44 } else if (p.getVolume1()<20) {
45 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque1.png")));
46 } else if (p.getVolume1()<30) {
47 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque2.png")));
48 } else if (p.getVolume1()<40) {
49 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque3.png")));
50 } else if (p.getVolume1()<50) {
51 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque4.png")));
52 } else if (p.getVolume1()<60) {
53 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque5.png")));
54 } else if (p.getVolume1()<70) {
55 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque6.png")));
56 } else if (p.getVolume1()<80) {
57 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque7.png")));
58 } else if (p.getVolume1()<90) {
59 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque8.png")));
60 } else if (p.getVolume1()<100) {
61 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque9.png")));
62 } else {
63 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque10.png")));
64 }
65 if (p.getVolume2()<10) {
66 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque0.png")));
67 } else if (p.getVolume2()<20) {
68 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque1.png")));
69 } else if (p.getVolume2()<30) {
70 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque2.png")));
71 } else if (p.getVolume2()<40) {
72 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque3.png")));
73 } else if (p.getVolume2()<50) {
74 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque4.png")));
75 } else if (p.getVolume2()<60) {
76 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque5.png")));
```

```
77 } else if (p.getVolume2()<70) {
78 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque6.png")));
79 } else if (p.getVolume2()<80) {
80 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque7.png")));
81 } else if (p.getVolume2()<90) {
82 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque8.png")));
83 } else if (p.getVolume2()<100) {
84 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque9.png")));
85 } else {
86 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque10.png")));
87 }
88 }
89
90 /**
91 * This method is called from within the constructor to initialize the form.
92 * WARNING: Do NOT modify this code. The content of this method is always
93 * regenerated by the Form Editor.
94 */
95 @SuppressWarnings("unchecked")
96 // <editor-fold defaultstate="collapsed" desc="Generated Code">
97 private void initComponents() {
98
99 jL1 = new javax.swing.JLabel();
100 jL2 = new javax.swing.JLabel();
101 jL3 = new javax.swing.JLabel();
102 jL4 = new javax.swing.JLabel();
103 jL5 = new javax.swing.JLabel();
104 jL6 = new javax.swing.JLabel();
105 jSystem = new javax.swing.JPanel();
106 jTanque1 = new javax.swing.JLabel();
107 jT1 = new javax.swing.JPanel();
108 jTemperatura1 = new javax.swing.JLabel();
109 jValvula1 = new javax.swing.JLabel();
110 jV1 = new javax.swing.JPanel();
111 jVazao1 = new javax.swing.JLabel();
112 jTanque2 = new javax.swing.JLabel();
113 jValvula2 = new javax.swing.JLabel();
114 jV2 = new javax.swing.JPanel();
115 jVazao2 = new javax.swing.JLabel();
116 jTubo = new javax.swing.JLabel();
117 jT2 = new javax.swing.JPanel();
118 jTemperatura2 = new javax.swing.JLabel();
119 jTC1 = new javax.swing.JToggleButton();
120 jTC2 = new javax.swing.JToggleButton();
121 jFC1_up = new javax.swing.JButton();
122 jFC1_down = new javax.swing.JButton();
123 jFC2_down = new javax.swing.JButton();
124 jFC2_up = new javax.swing.JButton();
125
126 setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
127 setTitle("TCC 2 - Nathan Fernandes");
128 setResizable(false);
129 addWindowListener(new java.awt.event.WindowAdapter() {
```

```
130 public void windowOpened(java.awt.event.WindowEvent evt) {
131 formWindowOpened(evt);
132 }
133 });
134
135 jL1.setText("Universidade Tecnológica Federal do Paraná");
136
137 jL2.setText("Disciplina de Trabalho de Conclusão de Curso 2");
138
139 jL3.setText("Curso de Engenharia Elétrica");
140
141 jL4.setText("Desenvolvido por Nathan Fernandes");
142
143 jL5.setText("Orientado por Prof. Dr. Wagner Endo");
144
145 jL6.setText("Simulação Sistema Supervisório SMAR");
146
147 jSystem.setBorder(javax.swing.BorderFactory.createEtchedBorder());
148 jSystem.setPreferredSize(new java.awt.Dimension(960, 540));
149
150 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/tanque0.png"))); //
NOI18N
151
152 jT1.setBorder(javax.swing.BorderFactory.createEtchedBorder());
153 jT1.setPreferredSize(new java.awt.Dimension(100, 40));
154
155 jTemperatura1.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
156 jTemperatura1.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
157 jTemperatura1.setPreferredSize(new java.awt.Dimension(96, 36));
158
159 javax.swing.GroupLayout jT1Layout = new javax.swing.GroupLayout(jT1);
160 jT1.setLayout(jT1Layout);
161 jT1Layout.setHorizontalGroup(
162 jT1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
163 .addComponent(jTemperatura1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
164 );
165 jT1Layout.setVerticalGroup(
166 jT1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
167 .addComponent(jTemperatura1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
168 );
169
170 jValvula1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/valvula_off.png")));
// NOI18N
171
172 jV1.setBorder(javax.swing.BorderFactory.createEtchedBorder());
173 jV1.setPreferredSize(new java.awt.Dimension(100, 40));
174
175 jVazao1.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
176 jVazao1.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
```

```
177 jVazao1.setPreferredSize(new java.awt.Dimension(96, 36));
178
179 javax.swing.GroupLayout jV1Layout = new javax.swing.GroupLayout(jV1);
180 jV1.setLayout(jV1Layout);
181 jV1Layout.setHorizontalGroup(
182 jV1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
183 .addGroup(jV1Layout.createSequentialGroup()
184 .addComponent(jVazao1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
185 .addGap(0, 0, Short.MAX_VALUE))
186 );
187 jV1Layout.setVerticalGroup(
188 jV1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
189 .addComponent(jVazao1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
190 );
191
192 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/tanque0.png"))); //
NOI18N
193
194 jValvula2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/valvula_off.png")));
// NOI18N
195
196 jV2.setBorder(javax.swing.BorderFactory.createEtchedBorder());
197 jV2.setPreferredSize(new java.awt.Dimension(100, 40));
198
199 jVazao2.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
200 jVazao2.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
201 jVazao2.setPreferredSize(new java.awt.Dimension(96, 36));
202
203 javax.swing.GroupLayout jV2Layout = new javax.swing.GroupLayout(jV2);
204 jV2.setLayout(jV2Layout);
205 jV2Layout.setHorizontalGroup(
206 jV2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
207 .addComponent(jVazao2, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
208 );
209 jV2Layout.setVerticalGroup(
210 jV2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
211 .addComponent(jVazao2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
212 );
213
214 jTubo.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
215 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/tubo0.png"))); //
NOI18N
216 jTubo.setVerticalAlignment(javax.swing.SwingConstants.TOP);
217 jTubo.setPreferredSize(new java.awt.Dimension(650, 196));
218
219 jT2.setBorder(javax.swing.BorderFactory.createEtchedBorder());
220 jT2.setPreferredSize(new java.awt.Dimension(100, 40));
```

```
221
222 jTemperatura2.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
223 jTemperatura2.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
224 jTemperatura2.setPreferredSize(new java.awt.Dimension(96, 36));
225
226 javax.swing.GroupLayout jT2Layout = new javax.swing.GroupLayout(jT2);
227 jT2.setLayout(jT2Layout);
228 jT2Layout.setHorizontalGroup(
229 jT2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
230 .addComponent(jTemperatura2, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
231 );
232 jT2Layout.setVerticalGroup(
233 jT2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
234 .addComponent(jTemperatura2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
235 );
236
237 jTC1.setSelected(true);
238 jTC1.setText("ON");
239 jTC1.setPreferredSize(new java.awt.Dimension(60, 40));
240
241 jTC2.setSelected(true);
242 jTC2.setText("ON");
243 jTC2.setPreferredSize(new java.awt.Dimension(60, 40));
244
245 jFC1_up.setFont(new java.awt.Font("Dialog", 0, 10)); // NOI18N
246 jFC1_up.setText("+");
247 jFC1_up.setPreferredSize(new java.awt.Dimension(40, 15));
248 jFC1_up.addActionListener(new java.awt.event.ActionListener() {
249 public void actionPerformed(java.awt.event.ActionEvent evt) {
250 jFC1_upActionPerformed(evt);
251 }
252 });
253
254 jFC1_down.setFont(new java.awt.Font("Dialog", 0, 10)); // NOI18N
255 jFC1_down.setText("-");
256 jFC1_down.setPreferredSize(new java.awt.Dimension(40, 15));
257 jFC1_down.addActionListener(new java.awt.event.ActionListener() {
258 public void actionPerformed(java.awt.event.ActionEvent evt) {
259 jFC1_downActionPerformed(evt);
260 }
261 });
262
263 jFC2_down.setFont(new java.awt.Font("Dialog", 0, 10)); // NOI18N
264 jFC2_down.setText("-");
265 jFC2_down.addActionListener(new java.awt.event.ActionListener() {
266 public void actionPerformed(java.awt.event.ActionEvent evt) {
267 jFC2_downActionPerformed(evt);
268 }
269 });
270
```

```
271 jFC2_up.setFont(new java.awt.Font("Dialog", 0, 10)); // NOI18N
272 jFC2_up.setText("+");
273 jFC2_up.addActionListener(new java.awt.event.ActionListener() {
274 public void actionPerformed(java.awt.event.ActionEvent evt) {
275 jFC2_upActionPerformed(evt);
276 }
277 });
278
279 javax.swing.GroupLayout jSystemLayout = new javax.swing.GroupLayout(jSystem);
280 jSystem.setLayout(jSystemLayout);
281 jSystemLayout.setHorizontalGroup(
282 jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
283 .addGroup(jSystemLayout.createSequentialGroup()
284 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
285 .addGroup(jSystemLayout.createSequentialGroup()
286 .addContainerGap()
287 .addComponent(jTanque1)
288 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
289 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
290 .addComponent(jT1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
291 .addComponent(jTC1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
292 .addGap(188, 188, 188)
293 .addComponent(jTanque2)
294 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
295 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
296 .addComponent(jT2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
297 .addComponent(jTC2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)))
298 .addGroup(jSystemLayout.createSequentialGroup()
299 .addGap(87, 87, 87)
300 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
301 .addGroup(jSystemLayout.createSequentialGroup()
302 .addComponent(jValvula1)
303 .addGap(10, 10, 10)
304 .addComponent(jV1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
305 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
306 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
307 .addComponent(jFC1_up, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
308 .addComponent(jFC1_down, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
309 .addGap(294, 294, 294)
310 .addComponent(jValvula2)
311 .addGap(10, 10, 10)
312 .addComponent(jV2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
313 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
314 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING,
false)
315 .addComponent(jFC2_up, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
316 .addComponent(jFC2_down, javax.swing.GroupLayout.PREFERRED_SIZE, 40,
javax.swing.GroupLayout.PREFERRED_SIZE)))
317 .addComponent(jTubo, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))))
318 .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
319 );
320 jSystemLayout.setVerticalGroup(
321 jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
322 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jSystemLayout.createSequentialGroup()
323 .addContainerGap()
324 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
325 .addComponent(jTanque1)
326 .addGroup(jSystemLayout.createSequentialGroup()
327 .addComponent(jTC1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
328 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
329 .addComponent(jT1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
330 .addGroup(jSystemLayout.createSequentialGroup()
331 .addComponent(jTC2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
332 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
333 .addComponent(jT2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
334 .addComponent(jTanque2))
335 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
336 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.CENTER)
337 .addComponent(jV2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
338 .addComponent(jValvula2)
339 .addComponent(jV1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
340 .addComponent(jValvula1)
341 .addGroup(jSystemLayout.createSequentialGroup()
342 .addComponent(jFC1_up, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
343 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
344 .addComponent(jFC1_down, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
345 .addGroup(jSystemLayout.createSequentialGroup()
346 .addComponent(jFC2_up, javax.swing.GroupLayout.PREFERRED_SIZE, 16,
javax.swing.GroupLayout.PREFERRED_SIZE)
347 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
348 .addComponent(jFC2_down, javax.swing.GroupLayout.PREFERRED_SIZE, 16,
javax.swing.GroupLayout.PREFERRED_SIZE)))
349 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
```

```
350 .addComponent(jTubo, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
351 .addContainerGap())
352 );
353
354 javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
355 getContentPane().setLayout(layout);
356 layout.setHorizontalGroup(
357 layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
358 .addGroup(layout.createSequentialGroup()
359 .addContainerGap()
360 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
361 .addGroup(layout.createSequentialGroup()
362 .addComponent(jSystem, javax.swing.GroupLayout.PREFERRED_SIZE, 940,
javax.swing.GroupLayout.PREFERRED_SIZE)
363 .addGap(0, 0, Short.MAX_VALUE))
364 .addGroup(layout.createSequentialGroup()
365 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
366 .addComponent(jL1)
367 .addComponent(jL4)
368 .addComponent(jL3)
369 .addComponent(jL2)
370 .addComponent(jL5))
371 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
372 .addComponent(jL6)))
373 .addContainerGap())
374 );
375 layout.setVerticalGroup(
376 layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
377 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
378 .addContainerGap()
379 .addComponent(jSystem, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
380 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
381 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
382 .addComponent(jL6)
383 .addComponent(jL1))
384 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
385 .addComponent(jL2)
386 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
387 .addComponent(jL3)
388 .addGap(5, 5, 5)
389 .addComponent(jL4)
390 .addGap(5, 5, 5)
391 .addComponent(jL5)
392 .addContainerGap())
393 );
394
395 pack();
396 setLocationRelativeTo(null);
397 }// </editor-fold>
398
```

```
399 private void formWindowOpened(java.awt.event.WindowEvent evt) {
400 updateViewer();
401
402 // Timer de 1000 ms para atualizar e registrar valores
403 final long segundos = 1000; // time in milliseconds
404 final Timer timer = new Timer();
405 final TimerTask task;
406 task = new TimerTask() {
407 @Override
408 public void run() {
409 // Atualizando temperatura 1
410 p.setTC1(jTC1.isSelected());
411 if (p.getTC1()) {
412 if (p.getTemperatura1()<45) {
413 p.setTemperatura1(p.getTemperatura1()+1);
414 }
415 } else {
416 if (p.getTemperatura1()>25) {
417 p.setTemperatura1(p.getTemperatura1()-1);
418 }
419 }
420 // Atualizando temperatura 2
421 p.setTC2(jTC2.isSelected());
422 if (p.getTC2()) {
423 if (p.getTemperatura2()<90) {
424 p.setTemperatura2(p.getTemperatura2()+1);
425 }
426 } else {
427 if (p.getTemperatura2()>25) {
428 p.setTemperatura2(p.getTemperatura2()-1);
429 }
430 }
431 // Atualizando volume 1
432 p.setVolume1(p.getVolume1() + 1 - p.getVazao1());
433 if (p.getVolume1()<0) {
434 p.setVolume1(0);
435 } else if (p.getVolume1()>100) {
436 p.setVolume1(100);
437 }
438 // Atualizando volume 2
439 p.setVolume2(p.getVolume2() + 1 - p.getVazao2());
440 if (p.getVolume2()<0) {
441 p.setVolume2(0);
442 } else if (p.getVolume2()>100) {
443 p.setVolume2(100);
444 }
445 run.insertDatabase(p);
446 updateViewer();
447 }
448 };
449 timer.scheduleAtFixedRate(task, 0, segundos);
450 }
451
```

```
452 private void jFC1_upActionPerformed(java.awt.event.ActionEvent evt) {
453 p.setVazao1(p.getVazao1()+1);
454 if (p.getVazao1()>10) {
455 p.setVazao1(10);
456 }
457 updateViewer();
458 }
459
460 private void jFC1_downActionPerformed(java.awt.event.ActionEvent evt) {
```

452 private void jFC1_upActionPerformed(java.awt.event.ActionEvent evt) {
460 private void jFC1_downActionPerformed(java.awt.event.ActionEvent evt) {

```
461 p.setVazao1(p.getVazao1()-1);
462 if (p.getVazao1()<0) {
463 p.setVazao1(0);
464 }
465 updateViewer();
466 }
467
468 private void jFC2_upActionPerformed(java.awt.event.ActionEvent evt) {
469 p.setVazao2(p.getVazao2()+1);
470 if (p.getVazao2()>10) {
471 p.setVazao2(10);
472 }
473 updateViewer();
474 }
475
476 private void jFC2_downActionPerformed(java.awt.event.ActionEvent evt) {
477 p.setVazao2(p.getVazao2()-1);
478 if (p.getVazao2()<0) {
479 p.setVazao2(0);
480 }
481 updateViewer();
482 }
483
484 /**
485 * @param args the command line arguments
486 */
487 public static void main(String args[]) {
488 /* Set the Nimbus look and feel */
489 //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
490 /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
491 * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
492 */
493 try {
494 for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
495 if ("Nimbus".equals(info.getName())) {
496 javax.swing.UIManager.setLookAndFeel(info.getClassName());
497 break;
498 }
499 }
500 } catch (ClassNotFoundException | InstantiationException | IllegalAccessException |
javax.swing.UnsupportedLookAndFeelException ex) {
```

```
501 java.util.logging.Logger.getLogger(GUI1.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
502 }
503 //</editor-fold>
504 //</editor-fold>
505 //</editor-fold>
506 //</editor-fold>
507
508 //</editor-fold>
509
510 /* Create and display the form */
511 java.awt.EventQueue.invokeLater(() -> {
512 new GUI1().setVisible(true);
513 });
514 }
515
516 // Variables declaration - do not modify
517 private javax.swing.JButton jFC1_down;
518 private javax.swing.JButton jFC1_up;
519 private javax.swing.JButton jFC2_down;
520 private javax.swing.JButton jFC2_up;
521 private javax.swing.JLabel jL1;
522 private javax.swing.JLabel jL2;
523 private javax.swing.JLabel jL3;
524 private javax.swing.JLabel jL4;
525 private javax.swing.JLabel jL5;
526 private javax.swing.JLabel jL6;
527 private javax.swing.JPanel jSystem;
528 private javax.swing.JPanel jT1;
529 private javax.swing.JPanel jT2;
530 private javax.swing.JToggleButton jTC1;
531 private javax.swing.JToggleButton jTC2;
532 private javax.swing.JLabel jTanque1;
533 private javax.swing.JLabel jTanque2;
534 private javax.swing.JLabel jTemperatura1;
535 private javax.swing.JLabel jTemperatura2;
536 private javax.swing.JLabel jTubo;
537 private javax.swing.JPanel jV1;
538 private javax.swing.JPanel jV2;
539 private javax.swing.JLabel jValvula1;
540 private javax.swing.JLabel jValvula2;
541 private javax.swing.JLabel jVazao1;
542 private javax.swing.JLabel jVazao2;
543 // End of variables declaration
544 }
```

ANEXO D

NETBEANS SISTEMA 2

Este é o código de criação da interface do sistema 2, o qual faz somente a leitura dos dados no servidor, semelhante a um sistema qualquer para acesso remoto. Observe que este código é basicamente uma réplica do sistema 1, porém sem os botões e funções para modificação das variáveis e registros no servidor.

```
1 package GUI;
2
3 import Resources.MySQL_Sintaxes;
4 import Resources.References;
5 import java.util.Timer;
6 import java.util.TimerTask;
7
8 public class GUI2 extends javax.swing.JFrame {
9
10 private final MySQL_Sintaxes run = new MySQL_Sintaxes();
11 private References p = new References();
12
13 public GUI2() {
14 initComponents();
15 }
16
17 private void updateViewer() {
18 jTemperatura1.setText(p.getTemperatura1()+" [°C] ");
19 jTemperatura2.setText(p.getTemperatura2()+" [°C] ");
20 jVazao1.setText(p.getVazao1()+" [m³/s] ");
21 jVazao2.setText(p.getVazao2()+" [m³/s] ");
22 if (p.getVazao1()==0) {
23 jValvula1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_off.png")));
24 } else {
25 jValvula1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_on.png")));
26 }
27 if (p.getVazao2()==0) {
28 jValvula2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_off.png")));
29 } else {
30 jValvula2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/valvula_on.png")));
31 }
32 if (p.getVazao1()==0 && p.getVazao2()==0) {
33 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo0.png")));
34 } else if (p.getVazao1()>0 && p.getVazao2()==0) {
35 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo1.png")));
36 } else if (p.getVazao1()==0 && p.getVazao2()>0) {
37 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo2.png")));
38 } else if (p.getVazao1()>0 && p.getVazao2()>0) {
39 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tubo3.png")));
40 }
41 if (p.getVolume1()<10) {
42 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque0.png")));
43 } else if (p.getVolume1()<20) {
```

```
44 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque1.png")));
45 } else if (p.getVolume1()<30) {
46 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque2.png")));
47 } else if (p.getVolume1()<40) {
48 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque3.png")));
49 } else if (p.getVolume1()<50) {
50 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque4.png")));
51 } else if (p.getVolume1()<60) {
52 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque5.png")));
53 } else if (p.getVolume1()<70) {
54 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque6.png")));
55 } else if (p.getVolume1()<80) {
56 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque7.png")));
57 } else if (p.getVolume1()<90) {
58 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque8.png")));
59 } else if (p.getVolume1()<100) {
60 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque9.png")));
61 } else {
62 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque10.png")));
63 }
64 if (p.getVolume2()<10) {
65 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque0.png")));
66 } else if (p.getVolume2()<20) {
67 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque1.png")));
68 } else if (p.getVolume2()<30) {
69 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque2.png")));
70 } else if (p.getVolume2()<40) {
71 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque3.png")));
72 } else if (p.getVolume2()<50) {
73 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque4.png")));
74 } else if (p.getVolume2()<60) {
75 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque5.png")));
76 } else if (p.getVolume2()<70) {
77 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque6.png")));
78 } else if (p.getVolume2()<80) {
79 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque7.png")));
80 } else if (p.getVolume2()<90) {
81 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque8.png")));
82 } else if (p.getVolume2()<100) {
83 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque9.png")));
84 } else {
85 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/icons/tanque10.png")));
86 }
87 }
88
89 /**
90 * This method is called from within the constructor to initialize the form.
91 * WARNING: Do NOT modify this code. The content of this method is always
92 * regenerated by the Form Editor.
93 */
94 @SuppressWarnings("unchecked")
95 // <editor-fold defaultstate="collapsed" desc="Generated Code">
96 private void initComponents() {
```

```
 97
 98 jLabel9 = new javax.swing.JLabel();
 99 jButton1 = new javax.swing.JButton();
100 jL1 = new javax.swing.JLabel();
101 jL2 = new javax.swing.JLabel();
102 jL3 = new javax.swing.JLabel();
103 jL4 = new javax.swing.JLabel();
104 jL5 = new javax.swing.JLabel();
105 jL6 = new javax.swing.JLabel();
106 jSystem = new javax.swing.JPanel();
107 jTanque1 = new javax.swing.JLabel();
108 jT1 = new javax.swing.JPanel();
109 jTemperatura1 = new javax.swing.JLabel();
110 jValvula1 = new javax.swing.JLabel();
111 jV1 = new javax.swing.JPanel();
112 jVazao1 = new javax.swing.JLabel();
113 jTanque2 = new javax.swing.JLabel();
114 jValvula2 = new javax.swing.JLabel();
115 jV2 = new javax.swing.JPanel();
116 jVazao2 = new javax.swing.JLabel();
117 jTubo = new javax.swing.JLabel();
118 jT2 = new javax.swing.JPanel();
119 jTemperatura2 = new javax.swing.JLabel();
120
121 jLabel9.setText("jLabel9");
122
123 jButton1.setText("jButton1");
124
125 setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
126 setTitle("TCC 2 - Nathan Fernandes");
127 setResizable(false);
128 addWindowListener(new java.awt.event.WindowAdapter() {
129 public void windowOpened(java.awt.event.WindowEvent evt) {
130 formWindowOpened(evt);
131 }
132 });
133
134 jL1.setText("Universidade Tecnológica Federal do Paraná");
135
136 jL2.setText("Disciplina de Trabalho de Conclusão de Curso 2");
137
138 jL3.setText("Curso de Engenharia Elétrica");
139
140 jL4.setText("Desenvolvido por Nathan Fernandes");
141
142 jL5.setText("Orientado por Prof. Dr. Wagner Endo");
143
144 jL6.setText("Simulação Sistema Supervisório Alternativo");
145
146 jSystem.setBorder(javax.swing.BorderFactory.createEtchedBorder());
147 jSystem.setPreferredSize(new java.awt.Dimension(960, 540));
148
```

```
149 jTanque1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/tanque0.png"))); //
NOI18N
150
151 jT1.setBorder(javax.swing.BorderFactory.createEtchedBorder());
152 jT1.setPreferredSize(new java.awt.Dimension(100, 40));
153
154 jTemperatura1.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
155 jTemperatura1.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
156 jTemperatura1.setPreferredSize(new java.awt.Dimension(96, 36));
157
158 javax.swing.GroupLayout jT1Layout = new javax.swing.GroupLayout(jT1);
159 jT1.setLayout(jT1Layout);
160 jT1Layout.setHorizontalGroup(
161 jT1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
162 .addComponent(jTemperatura1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
163 );
164 jT1Layout.setVerticalGroup(
165 jT1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
166 .addComponent(jTemperatura1, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
167 );
168
169 jValvula1.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/valvula_off.png")));
// NOI18N
170
171 jV1.setBorder(javax.swing.BorderFactory.createEtchedBorder());
172 jV1.setPreferredSize(new java.awt.Dimension(100, 40));
173
174 jVazao1.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
175 jVazao1.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
176 jVazao1.setPreferredSize(new java.awt.Dimension(96, 36));
177
178 javax.swing.GroupLayout jV1Layout = new javax.swing.GroupLayout(jV1);
179 jV1.setLayout(jV1Layout);
180 jV1Layout.setHorizontalGroup(
181 jV1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
182 .addGroup(jV1Layout.createSequentialGroup()
183 .addComponent(jVazao1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
184 .addGap(0, 0, Short.MAX_VALUE))
185 );
186 jV1Layout.setVerticalGroup(
187 jV1Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
188 .addComponent(jVazao1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
189 );
190
191 jTanque2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/tanque0.png"))); //
NOI18N
192
```

```
193 jValvula2.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/valvula_off.png")));
// NOI18N
194
195 jV2.setBorder(javax.swing.BorderFactory.createEtchedBorder());
196 jV2.setPreferredSize(new java.awt.Dimension(100, 40));
197
198 jVazao2.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
199 jVazao2.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
200 jVazao2.setPreferredSize(new java.awt.Dimension(96, 36));
201
202 javax.swing.GroupLayout jV2Layout = new javax.swing.GroupLayout(jV2);
203 jV2.setLayout(jV2Layout);
204 jV2Layout.setHorizontalGroup(
205 jV2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
206 .addComponent(jVazao2, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
207 );
208 jV2Layout.setVerticalGroup(
209 jV2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
210 .addComponent(jVazao2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
211 );
212
213 jTubo.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
214 jTubo.setIcon(new javax.swing.ImageIcon(getClass().getResource("/Icons/tubo0.png"))); //
NOI18N
215 jTubo.setVerticalAlignment(javax.swing.SwingConstants.TOP);
216 jTubo.setPreferredSize(new java.awt.Dimension(650, 196));
217
218 jT2.setBorder(javax.swing.BorderFactory.createEtchedBorder());
219 jT2.setPreferredSize(new java.awt.Dimension(100, 40));
220
221 jTemperatura2.setFont(new java.awt.Font("Dialog", 1, 18)); // NOI18N
222 jTemperatura2.setHorizontalAlignment(javax.swing.SwingConstants.RIGHT);
223 jTemperatura2.setPreferredSize(new java.awt.Dimension(96, 36));
224
225 javax.swing.GroupLayout jT2Layout = new javax.swing.GroupLayout(jT2);
226 jT2.setLayout(jT2Layout);
227 jT2Layout.setHorizontalGroup(
228 jT2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
229 .addComponent(jTemperatura2, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
230 );
231 jT2Layout.setVerticalGroup(
232 jT2Layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
233 .addComponent(jTemperatura2, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
234 );
235
236 javax.swing.GroupLayout jSystemLayout = new javax.swing.GroupLayout(jSystem);
237 jSystem.setLayout(jSystemLayout);
```

```
238 jSystemLayout.setHorizontalGroup(
239 jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
240 .addGroup(jSystemLayout.createSequentialGroup()
241 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
242 .addGroup(jSystemLayout.createSequentialGroup()
243 .addContainerGap()
244 .addComponent(jTanque1)
245 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
246 .addComponent(jT1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
247 .addGap(188, 188, 188)
248 .addComponent(jTanque2)
249 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
250 .addComponent(jT2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
251 .addGroup(jSystemLayout.createSequentialGroup()
252 .addGap(87, 87, 87)
253 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
254 .addGroup(jSystemLayout.createSequentialGroup()
255 .addComponent(jValvula1)
256 .addGap(10, 10, 10)
257 .addComponent(jV1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
258 .addGap(340, 340, 340)
259 .addComponent(jValvula2)
260 .addGap(10, 10, 10)
261 .addComponent(jV2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))
262 .addComponent(jTubo, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE))))
263 .addContainerGap(javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE))
264 );
265 jSystemLayout.setVerticalGroup(
266 jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
267 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING,
jSystemLayout.createSequentialGroup()
268 .addContainerGap()
269 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
270 .addComponent(jTanque1, javax.swing.GroupLayout.Alignment.TRAILING)
271 .addComponent(jT1, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
272 .addComponent(jT2, javax.swing.GroupLayout.Alignment.TRAILING,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)
273 .addComponent(jTanque2, javax.swing.GroupLayout.Alignment.TRAILING))
274 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.UNRELATED)
275 .addGroup(jSystemLayout.createParallelGroup(javax.swing.GroupLayout.Alignment.CENTER)
276 .addComponent(jV2, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
277 .addComponent(jValvula2)
278 .addComponent(jV1, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
```

```
279 .addComponent(jValvula1))
280 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
281 .addComponent(jTubo, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
282 .addContainerGap())
283 );
284
285 javax.swing.GroupLayout layout = new javax.swing.GroupLayout(getContentPane());
286 getContentPane().setLayout(layout);
287 layout.setHorizontalGroup(
288 layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
289 .addGroup(layout.createSequentialGroup()
290 .addContainerGap()
291 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
292 .addGroup(layout.createSequentialGroup()
293 .addComponent(jSystem, javax.swing.GroupLayout.PREFERRED_SIZE, 940,
javax.swing.GroupLayout.PREFERRED_SIZE)
294 .addGap(0, 0, Short.MAX_VALUE))
295 .addGroup(layout.createSequentialGroup()
296 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
297 .addComponent(jL1)
298 .addComponent(jL4)
299 .addComponent(jL3)
300 .addComponent(jL2)
301 .addComponent(jL5))
302 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED,
javax.swing.GroupLayout.DEFAULT_SIZE, Short.MAX_VALUE)
303 .addComponent(jL6)))
304 .addContainerGap())
305 );
306 layout.setVerticalGroup(
307 layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
308 .addGroup(javax.swing.GroupLayout.Alignment.TRAILING, layout.createSequentialGroup()
309 .addContainerGap()
310 .addComponent(jSystem, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
311 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
312 .addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.TRAILING)
313 .addComponent(jL6)
314 .addComponent(jL1))
315 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
316 .addComponent(jL2)
317 .addPreferredGap(javax.swing.LayoutStyle.ComponentPlacement.RELATED)
318 .addComponent(jL3)
319 .addGap(5, 5, 5)
320 .addComponent(jL4)
321 .addGap(5, 5, 5)
322 .addComponent(jL5)
323 .addContainerGap())
324 );
325
326 pack();
327 setLocationRelativeTo(null);
```

```
328 }// </editor-fold>
329
330 private void formWindowOpened(java.awt.event.WindowEvent evt) {
331 final long segundos = 1000; // time in milliseconds
332 final Timer timer = new Timer();
333 final TimerTask task;
334 task = new TimerTask() {
335 @Override
336 public void run() {
337 p = run.readDatabase();
338 updateViewer();
339 }
340 };
341 timer.scheduleAtFixedRate(task, 0, segundos);
342 }
343
344 /**
345 * @param args the command line arguments
346 */
347 public static void main(String args[]) {
348 /* Set the Nimbus look and feel */
349 //<editor-fold defaultstate="collapsed" desc=" Look and feel setting code (optional) ">
350 /* If Nimbus (introduced in Java SE 6) is not available, stay with the default look and feel.
351 * For details see http://download.oracle.com/javase/tutorial/uiswing/lookandfeel/plaf.html
352 */
353 try {
354 for (javax.swing.UIManager.LookAndFeelInfo info :
javax.swing.UIManager.getInstalledLookAndFeels()) {
355 if ("Nimbus".equals(info.getName())) {
356 javax.swing.UIManager.setLookAndFeel(info.getClassName());
357 break;
358 }
359 }
360 } catch (ClassNotFoundException | InstantiationException | IllegalAccessException |
javax.swing.UnsupportedLookAndFeelException ex) {
361 java.util.logging.Logger.getLogger(GUI2.class.getName()).log(java.util.logging.Level.SEVERE,
null, ex);
362 }
363 //</editor-fold>
364 //</editor-fold>
365 //</editor-fold>
366 //</editor-fold>
367 //</editor-fold>
368 //</editor-fold>
369 //</editor-fold>
370 //</editor-fold>
371
372 //</editor-fold>
373
374 /* Create and display the form */
375 java.awt.EventQueue.invokeLater(() -> {
376 new GUI2().setVisible(true);
377 });
```

```
378 }
379
380 // Variables declaration - do not modify
381 private javax.swing.JButton jButton1;
382 private javax.swing.JLabel jL1;
383 private javax.swing.JLabel jL2;
384 private javax.swing.JLabel jL3;
385 private javax.swing.JLabel jL4;
386 private javax.swing.JLabel jL5;
387 private javax.swing.JLabel jL6;
388 private javax.swing.JLabel jLabel9;
389 private javax.swing.JPanel jSystem;
390 private javax.swing.JPanel jT1;
391 private javax.swing.JPanel jT2;
392 private javax.swing.JLabel jTanque1;
393 private javax.swing.JLabel jTanque2;
394 private javax.swing.JLabel jTemperatura1;
395 private javax.swing.JLabel jTemperatura2;
396 private javax.swing.JLabel jTubo;
397 private javax.swing.JPanel jV1;
398 private javax.swing.JPanel jV2;
399 private javax.swing.JLabel jValvula1;
400 private javax.swing.JLabel jValvula2;
401 private javax.swing.JLabel jVazao1;
402 private javax.swing.JLabel jVazao2;
403 // End of variables declaration
404 }
```

ANEXO E

NETBEANS CLASSE PARA ABRIR/FECHAR CONEXÃO COM O BANCO DE DADOS

Este é o código de abertura e fechamento da conexão com o banco de dados, nele existem 4 funções, onde a primeira é utilizada para realizar a conexão com o banco de dados para registro e/ou leitura, enquanto as demais são para fechar essa conexão com o servidor.

```
1 package Resources;
 2
 3 import java.sql.Connection;
 4 import java.sql.DriverManager;
 5 import java.sql.PreparedStatement;
 6 import java.sql.ResultSet;
 7 import java.sql.SQLException;
 8 import java.util.logging.Level;
 9 import java.util.logging.Logger;
10
11 public class MySQL_Connection {
12
13 public static Connection getConnection() {
14 try {
15 Class.forName("com.mysql.cj.jdbc.Driver");
16 String host = "jdbc:mysql://script2us.com";
17 String user = "script2u_nathan";
18 String password = "-m-meB_IN-3l";
19 return DriverManager.getConnection(host, user, password);
20 } catch (ClassNotFoundException | SQLException ex) {
21 Logger.getLogger(MySQL_Connection.class.getName()).log(Level.SEVERE, null, ex);
22 }
23 return null;
24 }
25
26 public static void closeConnection(Connection conn) {
27 if (conn!=null) {
28 try {
29 conn.close();
30 } catch (SQLException ex) {
31 Logger.getLogger(MySQL_Connection.class.getName()).log(Level.SEVERE, null, ex);
32 }
33 }
34 }
35
36 public static void closeConnection(Connection conn, PreparedStatement stmt) {
37 closeConnection(conn);
38 if (stmt!=null) {
39 try {
40 stmt.close();
41 } catch (SQLException ex) {
```

```
42 Logger.getLogger(MySQL_Connection.class.getName()).log(Level.SEVERE, null, ex);
43 }
44 }
45 }
46
47 public static void closeConnection(Connection conn, PreparedStatement stmt, ResultSet rs) {
48 closeConnection(conn,stmt);
49 if (rs!=null) {
50 try {
51 rs.close();
52 } catch (SQLException ex) {
53 Logger.getLogger(MySQL_Connection.class.getName()).log(Level.SEVERE, null, ex);
54 }
55 }
56 }
57
58 }
```

ANEXO F

NETBEANS CLASSE PARA LER E REGISTRAR DADOS NO BANCO DE DADOS

Este é o código com as funções para registro e leitura dos dados no banco. Note que a sintaxe SQL é utilizada para obter os valores da tabela e que há também a classe intermediária *References* sendo usada para armazenar as variáveis em uma única função, para que seja possível retorná-la com todas as variáveis inclusas, evitando assim a criação de variáveis em todas as funções. Essa prática mantém o código mais enxuto, fácil e seguro de realizar alterações.

```
 1 package Resources;
 2
 3 import java.sql.Connection;
 4 import java.sql.PreparedStatement;
 5 import java.sql.ResultSet;
 6 import java.sql.SQLException;
 7 import java.util.logging.Level;
 8 import java.util.logging.Logger;
 9
10 public class MySQL_Sintaxes {
11
12 public References readDatabase() {
13 Connection conn = null;
14 PreparedStatement stmt = null;
15 ResultSet rs = null;
16 References p = new References();
17 conn = MySQL_Connection.getConnection();
18 try {
19 stmt = conn.prepareStatement("SELECT MAX(id) FROM script2u_tcc2.registros");
20 rs = stmt.executeQuery();
21 int id = 0;
22 while (rs.next()) {
23 id = rs.getInt(1);
24 }
25 if (id==0) {
26 p.setTemperatura1(25);
27 p.setTC1(false);
28 p.setTemperatura2(25);
29 p.setTC2(false);
30 p.setVolume1(100);
31 p.setVazao1(0);
32 p.setVolume2(100);
33 p.setVazao2(0);
34 } else {
35 stmt = conn.prepareStatement("SELECT
temperatura1,tc1,temperatura2,tc2,volume1,vazao1,volume2,vazao2 FROM script2u_tcc2.registros
ORDER BY id DESC LIMIT 1");
36 rs = stmt.executeQuery();
```

```
37 while (rs.next()) {
38 p.setTemperatura1(rs.getInt(1));
39 p.setTC1(rs.getBoolean(2));
40 p.setTemperatura2(rs.getInt(3));
41 p.setTC2(rs.getBoolean(4));
42 p.setVolume1(rs.getInt(5));
43 p.setVazao1(rs.getInt(6));
44 p.setVolume2(rs.getInt(7));
45 p.setVazao2(rs.getInt(8));
46 }
47 }
48 return p;
49 } catch (SQLException ex) {
50 Logger.getLogger(MySQL_Sintaxes.class.getName()).log(Level.SEVERE, null, ex);
51 } finally {
52 MySQL_Connection.closeConnection(conn, stmt, rs);
53 }
54 return null;
55 }
56
57 public void insertDatabase(References p) {
58 Connection conn = null;
59 PreparedStatement stmt = null;
60 ResultSet rs = null;
61 conn = MySQL_Connection.getConnection();
62 try {
63 stmt = conn.prepareStatement(
64 "INSERT INTO script2u_tcc2.registros"
65 + "(temperatura1,tc1,temperatura2,tc2,volume1,vazao1,volume2,vazao2)"
66 + "VALUES"
67 + "(?,?,?,?,?,?,?,?)");
68 stmt.setInt(1,p.getTemperatura1());
69 stmt.setBoolean(2,p.getTC1());
70 stmt.setInt(3,p.getTemperatura2());
71 stmt.setBoolean(4,p.getTC2());
72 stmt.setInt(5,p.getVolume1());
73 stmt.setInt(6,p.getVazao1());
74 stmt.setInt(7,p.getVolume2());
75 stmt.setInt(8,p.getVazao2());
76 stmt.executeUpdate();
77 } catch (SQLException ex) {
78 Logger.getLogger(MySQL_Sintaxes.class.getName()).log(Level.SEVERE, null, ex);
79 } finally {
80 MySQL_Connection.closeConnection(conn, stmt, rs);
81 }
82 }
83
84 }
```

ANEXO G

NETBEANS CLASSE PARA LEITURA E REGISTRO DAS VARIÁVEIS DENTRO DOS SISTEMAS

Esta é a classe que contém as funções para registro e leitura das variáveis manipuladas. Esta classe foi construída para facilitar o retorno das variáveis lidas no banco de dados pela função de leitura, de modo a retornar apenas a função como objeto, minimizando a quantidade de variáveis criadas e mantendo o código responsivo.

```
 1 package Resources;
 2
 3 public class References {
 4
 5 private int temperatura1;
 6 private Boolean tc1;
 7 private int temperatura2;
 8 private Boolean tc2;
 9 private int volume1;
10 private int vazao1;
11 private int volume2;
12 private int vazao2;
13
14 public int getTemperatura1() {
15 return temperatura1;
16 }
17 public void setTemperatura1(int temperatura1) {
18 this.temperatura1 = temperatura1;
19 }
20
21 public int getTemperatura2() {
22 return temperatura2;
23 }
24 public void setTemperatura2(int temperatura2) {
25 this.temperatura2 = temperatura2;
26 }
27
28 public Boolean getTC1() {
29 return tc1;
30 }
31 public void setTC1(Boolean tc1) {
32 this.tc1 = tc1;
33 }
34
35 public Boolean getTC2() {
36 return tc2;
37 }
38 public void setTC2(Boolean tc2) {
39 this.tc2 = tc2;
```

```
40 }
41
42 public int getVolume1() {
43 return volume1;
44 }
45 public void setVolume1(int volume1) {
46 this.volume1 = volume1;
47 }
48
49 public int getVolume2() {
50 return volume2;
51 }
52 public void setVolume2(int volume2) {
53 this.volume2 = volume2;
54 }
55
56 public int getVazao1() {
57 return vazao1;
58 }
59 public void setVazao1(int vazao1) {
60 this.vazao1 = vazao1;
61 }
62
63 public int getVazao2() {
64 return vazao2;
65 }
66 public void setVazao2(int vazao2) {
67 this.vazao2 = vazao2;
68 }
69 }
```