UNIVERSIDADE FEDERAL DA BAHIA

ESCOLA POLITÉCNICA

MATHEUS REIS SANTIAGO

SISTEMA SUPERVISÓRIO PARA UM SISTEMA DE ELEVAÇÃO POR BOMBEIO CENTRÍFUGO SUBMERSO

Salvador

2023

MATHEUS REIS SANTIAGO

SISTEMA SUPERVISÓRIO PARA UM SISTEMA DE ELEVAÇÃO POR BOMBEIO CENTRÍFUGO SUBMERSO

Trabalho de conclusão de curso apresentado ao colegiado do curso de Engenharia de Controle e Automação de Processos da Universidade Federal da Bahia como requisito parcial para obtenção do título Engenheiro de Controle e Automação de Processos.

Orientador: Prof. Dr. Daniel Diniz Santana

Salvador

2023

MATHEUS REIS SANTIAGO

SISTEMA SUPERVISÓRIO PARA UM SISTEMA DE ELEVAÇÃO POR BOMBEIO CENTRÍFUGO SUBMERSO

Trabalho de conclusão de curso apresentado ao colegiado do curso de Engenharia de Controle e Automação de Processos da Universidade Federal da Bahia como requisito parcial para obtenção do título de Bacharel em Engenharia de Controle e Automação de Processos.

13 de dezembro de 2023.

Comissão examinadora

gov.br Documento assinado digitalmente
DANIEL DINIZ SANTANA
Data: 22/12/2023 17:10:25-0300
Verifique em https://validar.iti.gov.br

Prof° Dr. Daniel Diniz Santana Doutor em Engenharia Industrial Universidade Federal da Bahia

gov.br Documento assinado digitalmente
LEONARDO SILVA DE SOUZA
Data: 22/12/2023 17:24:56-0300
Verifique em https://validar.iti.gov.br

Prof° Dr. Leonardo Silva de Souza Doutor em Engenharia de Processos QuimicosUniversidade Federal do Rio de Janeiro

gov.br Documento assinado digitalmente
ODILON SANTANA LUIZ DE ABREU
Data: 27/12/2023 11:07:13-0300
Verifique em https://validar.iti.gov.br

Prof° Me. Odilon Santana Luiz de AbreuMestre em Mecatrônica Universidade Federal da Bahia

Salvador

2023

RESUMO

A proposta aqui abordada tem como objetivo a construção de uma interface que permita o acompanhamento de informações do BCS instalado no Laboratório de Elevação Artificial (LEA) do Centro de Capacitação Tecnológica e Industrial (CTAI) da Universidade Federal da Bahia. Visando monitorar os dados referentes aos indicadores de desempenho, bem como as variáveis envolvidas no processo de extração de petróleo que têm impacto direto ou indireto nestes indicadores. Isto é feito através da utilização da linguagem de programação Python, na qual faz-se o uso da biblioteca Plotly, que se trata de um framework de aplicação web que permite visualizações gráficas interativas dos dados juntamente com as outras vantagens que o próprio python proporciona. Serão apresentados guia de uso da ferramenta e o detalhamento da solução proposta que consiste numa interface gráfica capaz de exibir as variáveis de entradas, saídas, indicadores de desempenho e o envelope de operaçâo do BCS – LEA.

Palavras-chave: Monitoramento, indicadores de desempenho, plotly, BCS, supervisóri

ABSTRACT

The proposal addressed here aims to build an interface that allows the monitoring of information from the BCS installed in the Artificial Elevation Laboratory (LEA) of the Technological and Industrial Training Center (CTAI) of the Federal University of Bahia. Aiming to monitor data relating to performance indicators, as well as the variables involved in the oil extraction process that have a direct or indirect impact on these indicators. This is done through the use of the Python programming language, which uses the Plotly library, which is a web application framework that allows interactive graphical visualizations of data along with the other advantages that Python itself provides. A guide for using the tool and details of the proposed solution will be presented, which consists of a graphical interface capable of displaying the input variables, outputs, performance indicators and the operating envelope of the BCS – LEA.

Keywords: Monitoring, performance indicators, plotly, BCS, supervisory.

SUMÁRIO

1 INTRODUÇÃO

De acordo com Soares (2011), em diversas áreas da produção humana em escala industrial pode-se encontrar a automação, e na exploração de petróleo não é diferente. Dentre as aplicações da automação na área de Oil & gas (O&G) está o controle da extração de petróleo, que visa a obtenção da maior rentabilidade possível da produção em relação aos recursos utilizados, ou seja, uma relação custo-benefício viável. Ainda segundo este autor, num cenário onde exista considerável quantidade de poços distribuídos em uma região de exploração, a utilização de um sistema supervisório que obtém os dados dos maquinários envolvidos na exploração é essencial para o funcionamento e otimização do sistema de produção de petróleo.

O propósito de um sistema de supervisão é fazer o processamento das informações do processo e torná-las acessíveis ao operador ou a qualquer pessoa com acesso ao programa supervisório (SANTOS, 2019). Além disso, pode-se fazer o armazenamento das informações obtidas em um banco de dados para consultas futuras do histórico de operações do processo, tornando possível, por exemplo, uma análise comparativa entre metodologias abordadas no controle do processo em diferentes momentos da exploração dos poços de petróleo, e a identificação de possíveis condições anormais através da interpretação dos dados apresentados pelo sistema supervisório.

Além de sua utilização a nível da operação, os sistemas supervisórios têm se mostrado bastante úteis na gestão do processo produtivo. Por esta razão, são vistos como ferramentas que passam pelas questões relacionadas a engenharia, chegando até a estruturação da gestão da empresa. Por se tratar de uma ferramenta extremamente útil para o acompanhamento da produção, seja na área petrolífera ou em qualquer setor industrial que demanda automação, o sistema supervisório pode oferecer três tipos de serviços para processos industriais automatizados, são eles: supervisão, operação e controle. De acordo Santos (2019), uma única interface supervisória pode oferecer todos esses serviços ou pode conter apenas um deles, dependendo da demanda necessária na operação.

Ao verificar mudanças significativas no desempenho em uma bombeio centrífugo submerso, pode-se fazer o rastreamento da causa raiz de um determinado problema, seja ele operacional ou de produção. No entanto, ainda segundo Santos (2019), para fazer esse monitoramento é necessário um contato direto com o hardware de controle, o que pode ser um trabalho custoso dependendo da distância de onde o equipamento de operação se encontra, assim existe a necessidade de tornar os dados disponíveis remotamente e a interface supervisória atende a esta necessidade.

No contexto da elevação de petróleo por bombeio centrífugo submerso (BCS), um sistema

supervisório pode conter informações das variáveis operacionais, acompanhar o melhor ponto de operação da BCS, em inglês, *best efficiency point* (BEP), pois, segundo Borges (2018), o BEP é uma grandeza dinâmica que varia conforme a frequência de acionamento da bomba, na qual é efetuada por um inversor de frequência ou por mudanças na viscosidade do fluido bombeado, e também, apresentar os indicadores de desempenho. Estes indicadores podem ser de natureza operacional ou de gestão e conhecê-los é fundamental para se chegar a um uso eficiente do sistema BCS (BORGES, 2018). A importância dos indicadores de desempenho está relacionada com a identificação de condições anormais no processo de extração, mais especificamente voltada aos equipamentos que compõem a BCS, assim como, no aperfeiçoamento do rendimento na extração de petróleo, pois fornecem informações que podem ser usadas na aplicação da otimização do processo. É importante ressaltar que, para cada bomba fabricada, existem particularidades em seus indicadores, por isso, faz-se necessário o acompanhamento destes indicadores em um sistema supervisória visando o funcionamento adequado da BCS durante sua operação (BORGES, 2018).

Nesse cenário, este trabalho terá como foco o sistema de elevação artificial por BCS do Laboratório de Elevação Artificial do Centro de Capacitação Tecnológica e Industrial (CTAI) da Universidade Federal da Bahia, e tem como objetivo principal o desenvolvimento uma interface, a nível de sistema supervisório, para acompanhar indicadores de desempenho, sejam eles operacionais e gerenciais, do sistema em questão.

1.1 SOLUÇÃO PROPOSTA

Uma interface capaz de apresentar as variáveis de desempenho e operacionais relacionadas ao bombeio centrífugo submerso (BCS) localizada no Laboratório de Elevação Artificial (LEA) da Universidade Federal da Bahia (UFBA), com funcionalidade de consulta em tempo real ao banco de dados para obtenção dos dados das variáveis de entradas e saídas do sistema. As variáveis de saída de interesse são: (i) a frequência de operação da bomba, (ii) abertura da válvula *choke*, (iii) pressão no reservatório, e (iv) pressão no manifold. As variáveis de entrada de interesse são: (i) pressão na *choke*, (ii) vazão média, (iii) nível do anular, e (iv) diferencial de pressão inserida pela bomba. Essas variáveis de entradas e saídas são apresentadas na interface através de gráficos interativos, além disso, encontra-se também os gráficos dos indicadores de desempenho (Head, Eficiência e Potência) da bomba, bem como o envelope de operação da BCS.

Do ponto de vista de controle, o sistema BCS -LEA possui duas variáveis manipuladas e diversas controladas dependendo da configuração de controle que se se deseja estabelecer. Entre essas variáveis, as manipuladas são, respectivamente, frequência de operação do motor da bomba

e abertura da válvula choque (LIMA; COSTA; SANTOS; FONTANA; ABREU; SILVA., 2020).

Dito isto, se faz necessário projetar uma arquitetura de comunicação que define o fluxo de dados, através do uso de ferramentas como banco de dados, protocolo de comunicação OPC (Open Platform Communications) e browser para exibição da interface.

1.2 OBJETIVO GERAL

O objetivo é a construção de uma interface gráfica a nível de sistema supervisório que permita o monitoramento de informações referentes aos indicadores de desempenho, bem como das variáveis envolvidas na operação da BCS - LEA. A interface é construída através da utilização da linguagem de programação Python, na qual faz-se o uso do framework Plotly.

1.3 OBJETIVOS ESPECÍFICOS

Destacam-se os seguintes objetivos específicos do trabalho:

a) Analisar as variáveis envolvidas no processo de elevação artificial através da BCS;
b) Elaborar uma interface, a nível de sistema supervisório, que permita o acompanhamento dessas variáveis;
c) Possibilidade de visualização das informações em tempo real, bem como a realização da consulta de histórico dos dados;
d) Construir uma arquitetura de fluxo de dados que faça o uso de ferramentas *open source* para fácil divulgação do produto.

1.4 ESTRUTURA DO TRABALHO

A estrutura deste trabalho é a seguinte: o segundo capítulo abordará de forma concisa os conceitos teóricos relacionados fluxo, armazenamento e tratamento de dados, bem como a exibição destes dados processados. No terceiro capítulo, será introduzida a ferramenta proposta. O quarto capítulo detalhará os principais recursos empregados na sua criação, juntamente com alguns aspectos da implementação e suas limitações. O capítulo 5 apresentará um guia de utilização, incluindo exemplos nos quais a a ferramenta foi testada.

2 CONCEITOS TEÓRICOS

Nesta seção, serão apresentados os princípios relacionados à bomba centrífuga submersa, além do uso de sistemas de supervisão na indústria, com foco especial no setor de óleo e gás (O&G). Também serão discutidos os elementos necessários para construção de uma interface gráfica a nível de camada supervisória, através da aplicação da linguagem de programação Python, onde se realizará o processamento e análise de dados com ênfase no uso da biblioteca

Plotly. Conceitos importantes como protocolos de comunicação, banco de dados e a arquitetura de comunicação entre as ferramentas pertinetes a construção e visualização da interface, serão igualmente abordados.

2.1 BOMBA CENTRÍFUGA

De acordo Varon (2013), bombas são dispositivos que convertem energia mecânica em energia hidráulica. A fonte de energia mecânica é fornecida por um motor ou outro elemento motriz, que transforma essa energia em energia hidráulica através da ação da bomba. A função primordial da bomba é movimentar diversos tipos de fluidos, o que a torna um componente vital em várias indústrias, incluindo a indústria de lubrificantes, combustíveis, alimentos, bebidas, produtos de limpeza, entre outras.

Segundo Fonseca (2019), existem dois tipos principais de bombas: as dinâmicas e as de deslocamento positivo. Nas bombas dinâmicas, a energia motriz é transmitida apenas de forma rotacional. Seu componente essencial para a transmissão de energia é o impulsor ou rotor, e seu princípio de funcionamento é governado pelas equações de Euler. Por outro lado, as bombas de deslocamento positivo, também conhecidas como bombas de ação recíproca e rotativa, têm uma característica distintiva: a transmissão de energia ocorre por meio de deslocamentos volumétricos gerados por êmbolos, engrenagens e outros componentes semelhantes (FONSECA, 219).

As bombas centrífugas são máquinas projetadas para transferir fluidos por meio da força centrífuga gerada pela rotação de um rotor dentro de uma carcaça. Esse movimento rotacional do fluido resulta em um aumento simultâneo na pressão e na energia cinética do fluido. Posteriormente, no difusor, parte dessa energia cinética é convertida em um aumento adicional de pressão, conseguindo isso por meio do aumento da área de fluxo (CASTELLANOS, 2019).

2.1.1 BOMBEIO CENTRÍFUGO SUBMERSO (BCS)

Segundo Castellanos (2019), o bombeio centrífugo submerso é um equipamento fundamental na indústria do petróleo, usado como um sistema de elevação artificial para extrair petróleo dos poços. Seu princípio de funcionamento envolve uma bomba centrífuga de vários estágios, instalada em profundidade dentro do poço. Esta bomba atua como um motor para misturar os fluidos extraídos do reservatório. Ao fornecer energia adicional ao reservatório, a

bomba eleva a mistura até a superfície para posterior tratamento e armazenamento. A Figura 1, ilustra a estrutura báscica que compõe uma BCS.

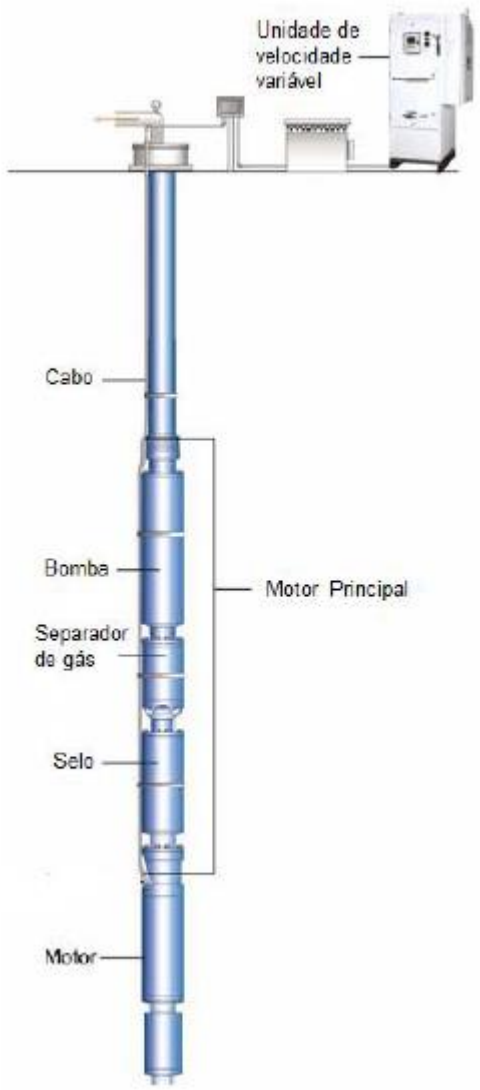

Figura 1: Configuração básica do equipamento BCS (REIS, 2018)

Segundo Reis (2018), o funcionamento da bomba é controlado por um motor acoplado, que geralmente está equipado com sensores para monitorar o desempenho do motor e algumas variáveis do comportamento da BCS. Em grandes empresas que fornecem BCS para a indústria do petróleo, esses sensores medem a pressão de sucção e descarga da bomba, a temperatura do fluido e do motor, a vibração, bem como vazamentos ou perdas de corrente. Essas informações são transmitidas pelo mesmo cabo elétrico que alimenta o motor, sendo posteriormente filtradas para leitura e registro histórico na superfície. Além disso, possibilitam o controle e a proteção do equipamento.

2.1.2 EQUIPAMENTOS DE SUBSUPERFÍCIE DA BCS

Os componentes de subsuperfície são posicionados dentro do poço. A seguir, serão apresentados os principais dispositivos.

- BOMBA DE ESTÁGIO MULTIPLOS

De acordo Varon (2013), a Bomba Centrífuga Submersa (BCS) geralmente é composta por vários estágios dispostos em série, cuja quantidade é determinada pelas necessidades de elevação do poço onde será instalada. Cada estágio consiste em um rotor (impeller) e um difusor (diffuser), conforme mostrado na Figura 2. Nessa representação, é possível observar o caminho percorrido pelo fluido em cada estágio.

Ainda de acordo com Varon (2013), quando o fluido passa pelo rotor, é sujeito a um campo centrífugo e move-se na direção radial antes de entrar na zona do difusor. No difusor, a energia na forma de velocidade é convertida em pressão, continuando esse processo no estágio seguinte de maneira semelhante. O ganho de pressão é progressivo ao longo da série de estágios da bomba, culminando na elevação total necessária para a descarga do fluido.

Figura 2: Conjunto Impelidor-Difusor (BORGES, 2018)

- ADMISSÃO DA BOMBA (Intake)

A entrada da bomba está situada na sua parte inferior, marcando o ponto onde o fluido inicialmente penetra no primeiro estágio do sistema. Essa entrada pode assumir duas formas: uma versão simples ou um tipo de separador de gás. A finalidade é evitar que a bomba sugue gás livre de maneira indesejada. A escolha entre essas opções de entrada depende da configuração do sistema, que requer uma avaliação cuidadosa dos parâmetros, incluindo a série específica da bomba, a taxa de fluxo do líquido a ser produzido e a proporção entre gás e líquido.

- MOTOR ELÉTRICO

O motor elétrico desempenha a importante função de transformar energia elétrica em energia mecânica, que é essencial para acionar a bomba centrífuga. Geralmente operando como um motor trifásico, ele mantém uma velocidade constante na faixa de 3500 RPM a uma frequência de 60 Hz (VARON, 2013). O eixo do motor está cuidadosamente conectado ao eixo do protetor, ao sistema de admissão e ao impulsor, formando um eixo contínuo que deve estar perfeitamente alinhado para garantir o funcionamento adequado. Esses motores são projetados para operar em ambientes desafiadores, sujeitos a altas pressões e temperaturas. Além disso, eles são projetados para serem imersos em fluidos viscosos, mostrando sua robustez e capacidade de lidar com condições adversas de operação.

- PROTETOR OU SELO

O protetor está posicionado entre o motor elétrico e a bomba centrífuga, exercendo várias funções essenciais. Ele atua como uma barreira protetora, impedindo que o fluido entre no motor, ao mesmo tempo em que equilibra a pressão interna do motor com a pressão do fluido produzido. Além disso, o protetor conecta a carcaça do motor à carcaça da bomba, evitando variações de pressão no motor e proporcionando um espaço adequado para a expansão e contração do óleo presente no motor.

- CABOS ELÉTRICOS

Conforme Araújo (2015), a energia elétrica é transferida da superfície para o motor por meio de um cabo elétrico trifásico. O dimensionamento desses cabos depende de vários fatores, como a distância entre o painel de controle e o motor, a temperatura operacional, a tensão

fornecida pela rede elétrica, o tipo de fluido a ser bombeado e o espaço disponível entre a coluna de produção e o revestimento. Em geral, utiliza-se um cabo redondo em áreas com espaço anular mais amplo (entre a coluna de produção e o revestimento) e um cabo chato em regiões onde o espaço é limitado. Essas escolhas são feitas para garantir uma transmissão eficiente de eletricidade, levando em consideração as condições específicas do ambiente e do sistema em questão.

2.2.1 EQUIPAMENTOS DE SUPERFÍCIE DA BCS

Na superfície, estão instalados equipamentos essenciais, compostos principalmente pelo quadro de comandos, transformador, caixa de ventilação e cabeça de produção. Cada um desses componentes será discutido em detalhes nos próximos ítens.

- QUADRO DE COMANDOS

O quadro de comandos assume a responsabilidade de operar e regular os dispositivos localizados no fundo do poço. Ele é essencialmente composto por dois compartimentos distintos. O primeiro, de média tensão, abriga os transformadores de corrente e de controle, além dos fusíveis de proteção que interrompem o circuito elétrico em situações de corrente excessiva. Também inclui uma chave seletora. O segundo compartimento, de baixa tensão, contém os relés eletromecânicos, o amperímetro e o temporizador (ARAÚJO, 2015).

- TRANSFORMADOR

De acordo Varon (2013), o transformador desempenha um papel crucial ao receber a tensão padrão da rede elétrica e convertê-la na voltagem necessária para alimentar o motor, levando em consideração as perdas associadas ao cabo elétrico. Existem dois tipos principais de transformadores: (i) o transformador de alimentação principal, usado para reduzir a alta voltagem da fonte primária para a voltagem mais baixa necessária pelo variador de frequência; e (ii) o transformador de saída, empregado quando há um variador de frequência instalado. Sua função é aumentar a voltagem de saída do variador de frequência até atingir a voltagem exigida pelo motor submerso.

- CAIXA DE VENTILAÇÃO

Segundo Araújo (2015), a função primordial da caixa de ventilação é assegurar a

ventilação dos cabos elétricos, permitindo a liberação do gás livre proveniente do poço para a atmosfera. Suas principais características incluem a conexão do cabo proveniente do transformador de saída ao conector superior da cabeça. Além disso, a caixa é facilmente acessível para a revisão elétrica dos dispositivos subterrâneos e serve como local para realizar medições das condições de isolamento e extensão do cabo.

- CABEÇA DE PRODUÇÃO

Conforme explicado por Borges (2018), a cabeça de produção é um componente especial que inclui uma passagem para a coluna de produção e outra para o cabo elétrico. Em poços terrestres, onde a pressão é baixa, costuma-se usar um flange bipartido com vedação de borracha. Nesse caso, a vedação é realizada por meio de placas que comprimem as borrachas em torno do cabo e da coluna de produção. Já em poços marítimos, sujeitos a altas pressões, emprega-se um mandril rosqueado no tubing hanger, com condutores elétricos em seu interior para a passagem da corrente elétrica. A escolha da cabeça de produção para um determinado poço envolve considerações como o diâmetro do revestimento e da coluna de produção, a espessura e o tipo do cabo, bem como as pressões presentes no sistema.

2.3 DESEMPENHO DA BCS

A performace do BCS é avaliado por meio de representações gráficas chamadas curvas de desempenho. Estas curvas são criadas em relação à taxa de fluxo de líquido e mostram a altura manométrica ou elevação (*Head*), a potência mecânica necessária para operar a bomba e sua eficiência. Esses dados são coletados em condições padrão específicas, onde o fluido tem uma densidade de 1.0 (com a água a 20ºC) e a bomba opera a uma velocidade constante de 3500 rotações por minuto (RPM) (REIS, 2018). A Figura 3 ilustra um exemplo dessas curvas de desempenho para uma BCS.

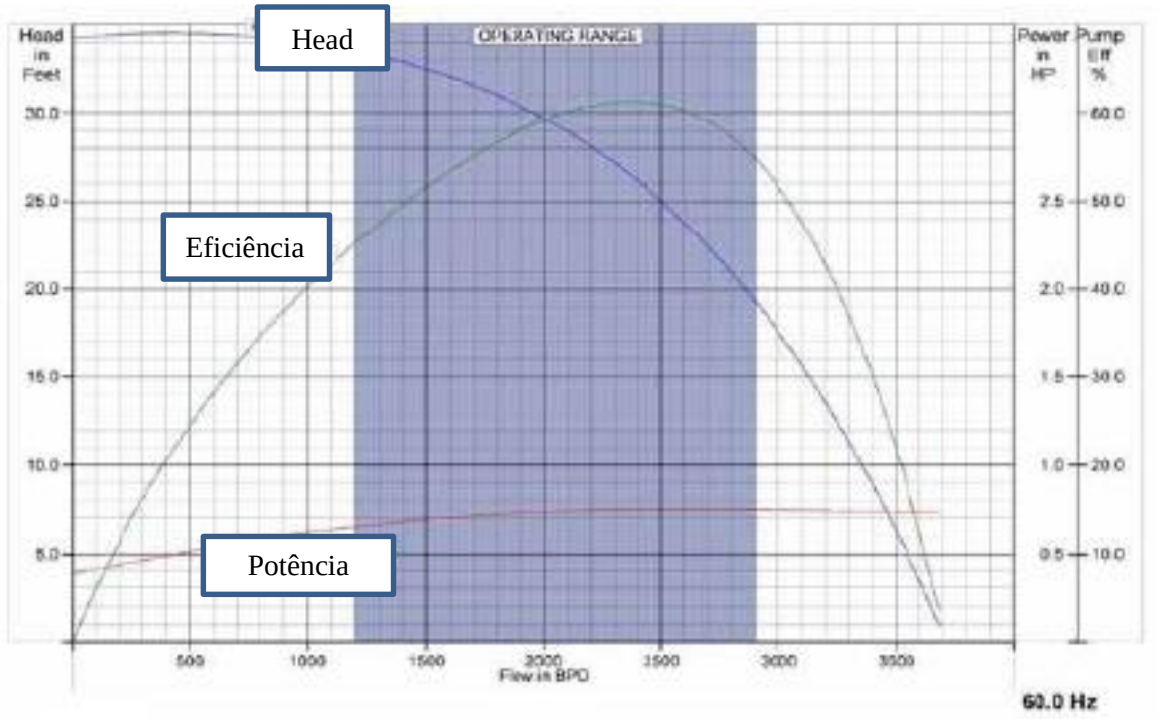

Figura 3: Exemplo de curvas de desempenho de uma BCS (REIS, 2018)

De acordo Pavlov et al (2014), a produção deve estar dentro do chamado envelope operacional do BCS, comumente definidos em relação às restrições do *head* e da vazão do BCS desta relação, tambem obtém-se a curva do sistema, que é uma métrica que relaciona a vazão e o head.

O termo "Head" de uma bomba, conforme descrito por Araújo (2015), representa a quantidade de energia que a bomba pode fornecer ao fluido por unidade de peso, para uma vazão específica como mostrado na Equação 1:

$$H = \frac{\Delta P}{\rho g}, \qquad (1)$$

em que, g representa a gravidade, ρ a massa específica do fluido e Δp diferença de pressão, que representa a pressão na descarga menos a pressão na sucção, produzida pela bomba, é convertida em unidades de comprimento, geralmente expressas em metros ou pés de coluna de fluido, levando em consideração a densidade do fluido ρ. Quando o "Head" é expresso em termos lineares, ele indica a altura manométrica que a bomba é capaz de superar para uma determinada vazão.

Apresentada na Equação 2, a potência absorvida pela bomba (BHP - Brake HorsePower) refere-se à energia que o motor deve fornecer ao eixo da bomba. Chamada de potência

hidráulica (P_h), essa quantidade está relacionada à vazão volumétrica do fluido (q) e à diferença de pressão (Δp) (VARON, 2013):

$$P_h = q\Delta P \qquad (2)$$

A eficiência (Ƞ) é calculada como a relação entre a energia útil transmitida ao fluido e a energia absorvida pela bomba, como pode ser visto na Equação 3:

$$\eta = \frac{P_h}{\omega T_{eixo}} = \frac{q\Delta P}{\omega T_{eixo}}, \qquad (3)$$

em que P_h é a potência hidrúlica, ω é velocidade angular de rotação do eixo e T_{eixo} que é o torque exercido pelo eixo. Ao analisar a curva de eficiência, é possível identificar as vazões em que a bomba apresenta o melhor desempenho (ARAÚJO, 2015).

2.4 SISTEMA SUPERVISÓRIO

De acordo com Santos (2019), com a crescente automação de diversos sistemas, tornou-se essencial supervisionar os equipamentos usados na automação de poços. Ainda segundo Santos (2019), esse monitoramento é crucial para identificar dispositivos defeituosos ou aprimorar o desempenho dos processos. Foi a necessidade gerada pela indústria que impulsionou o surgimento dos sistemas supervisórios. Esses sistemas desempenham um papel vital ao adquirir variáveis do processo, exibi-las e controlar dispositivos de automação que impactam diretamente nos procedimentos operacionais (SANTOS, 2019).

Os sistemas de supervisão precisam ter a capacidade de processar dados do processo e apresentá-los ao operador ou a outros usuários do software de supervisão. Além disso, eles podem realizar tarefas de controle em um nível de supervisão e, com a ajuda de mecanismos específicos integrados ao sistema computacional, tomar decisões e executar ações no processo de forma automática (SANTOS, 2019).

Em relação a aplicação desta tecnologia na indústria de *O&G*, segundo Franca (2011), devido à distribuição geográfica extensa dos poços e à necessidade de manter a produção de forma contínua, a automação tornou-se uma ferramenta indispensável. Nas empresas modernas, a automação desempenha um papel crucial, proporcionando suporte vital para melhorar a eficiência no uso da matéria-prima, reduzir os custos de produção e aprimorar a qualidade dos

produtos. Além disso, ela permite criar planos de manutenção que minimizam as interrupções no processo e oferecem respostas rápidas às flutuações na demanda. E ainda como destacado por Franca (2011), um sistema local de controle do processo de elevação de petróleo é de grande valor. Ele não apenas mantém o processo no ponto ótimo de operação, mas também identifica descontinuidades operacionais, restaura rapidamente a operação normal após perturbações, recuperando a produção o mais rápido possível. Além disso, tem a capacidade de diagnosticar as causas dos problemas, transmitindo alertas ao sistema de supervisão para a tomada de providências necessárias, como intervenções de limpeza, manutenção em equipamentos e outras medidas corretivas.

Os sistemas de supervisão tornaram-se essenciais na gestão empresarial, deixando de ser apenas ferramentas operacionais ou de engenharia para se transformarem em fontes valiosas de informação. Sua importância na estrutura organizacional das empresas é indiscutível (SNTOS, 2019).

2.5 PYTHON PARA ANÁLISE DE DADOS

Apesar de ser uma linguagem de programação consolidada, amplamente adotada por desenvolvedores experientes, a simplicidade do Python possibilita que profissionais e acadêmicos a utilizem sem a necessidade de uma vasta experiência em programação. A versatilidade do Python é uma de suas características mais marcantes, pois é uma linguagem multiparadigma, integrando vários estilos de programação, incluindo orientação a objetos e programação estruturada (FORMIGONI, 2021).

Ainda sengundo Formigoni (2021), essa facilidade de aprendizado faz do Python uma escolha popular para análise de dados. Além disso, sua comunidade ativa contribui continuamente para o desenvolvimento de ferramentas cada vez mais eficientes por meio da linguagem. Para facilitar ainda mais a análise de dados, existem diversos pacotes e bibliotecas de alto nível disponíveis, como por exemplo o Pandas, que oferece uma ampla variedade de ferramentas de programação para manipulação e análise de dados (MCKINNEY,W., E OUTROS, 2010)

De acordo Souza (2023), python possui uma notável capacidade de integração com outras tecnologias amplamente empregadas na análise de dados. Por exemplo, é viável extrair dados de bancos de dados SQL utilizando bibliotecas como o SQLAlchemy (BAYER, M., 2012).

Além disso, é possível combinar Python com ferramentas de Big Data, como o Apache Spark, para realizar análises distribuídas em vastos conjuntos de dados.

2.5.1 PLOTLY

Segundo Sievert, C. (2020), o Plotly é uma poderosa biblioteca de visualização de dados compatível com Python, Javascript e R. Esta ferramenta versátil oferece uma gama diversificada de produtos, desde a criação de dashboards até soluções para clientes SQL . O grande diferencial do Plotly é sua capacidade de permitir a integração fácil e eficiente de gráficos em diversas aplicações, incluindo os populares Jupyter Notebooks. Além disso, o Plotly se destaca por sua vasta variedade de visualizações pré-configuradas e sua capacidade massiva de personalização, tornando-o uma escolha ideal para profissionais que buscam criar representações visuais de dados altamente flexíveis e atraentes (VASCONCELLOS, 2023).

Ainda de acordo a Vasconcellos (2023), o Plotly oferece duas versões distintas: uma online e outra offline. Na primeira versão, seus gráficos são hospedados na plataforma da Plotly, tornando-os publicamente acessíveis e prontos para serem incorporados em sites e aplicativos de forma fácil e direta. Já na segunda versão, todos os gráficos são gerados diretamente em seu notebook e ficam disponíveis apenas lá. Essa opção é especialmente vantajosa quando é necessário compartilhar informações sensíveis restritas apenas aos membros de sua empresa, garantindo a confidencialidade dos dados apresentados.

2.6 BANCO DE DADOS

Segundo Souza (2022), um banco de dados é uma estrutura fundamental para organizar e armazenar informações sobre um domínio específico. Em termos simples, ele consiste no agrupamento de dados relacionados que precisam ser preservados para futuras referências ou auditorias. Empresas frequentemente lidam com uma ampla gama de informações que devem ser organizadas internamente para consulta posterior por suas equipes e gerentes. É aqui que entra a necessidade de um Sistema de Gerenciamento de Banco de Dados (SGBD). Na Figura 4, tem-se um exemplo simplificado de um sistema de banco de dados.

Figura 4: Ambiente simplificado de um sistema de banco de dados (ALVES, 2023).

Esse sistema permite a manipulação eficiente das informações, simplificando significativamente a rotina operacional da empresa. Atualmente, existem diversos tipos de SGBDs, cada um adaptado às necessidades específicas dos clientes. Entre alguns exemplos tem-se o Oracle, DB2, MySQL, SQL Server, PostgreSQL, entre outros. A escolha do SGBD adequado desempenha um papel crucial na eficiência e na segurança das operações de uma empresa, facilitando o acesso às informações vitais quando necessário (SOUZA, 2023).

2.7 PROTOCOLOS DE COMUNICAÇÃO

Em projetos de automação, onde se busca aprimorar a automação e a gestão de dados em um ambiente industrial dinâmico, a escolha cuidadosa dos protocolos de comunicação desempenha um papel central. Nesse contexto, a combinação estratégica dos protocolos OPC-DA (OLE for Process Control - Data Access) e ODBC (Open Database Connectivity) emerge como uma solução integral para atender às demandas específicas da aplicação proposta. A seguir serão apresentados os dois protocolos uteis para a realização do trabalho, a saber o protoloco OPC-DA e o ODBC.

2.7.1 OPC-DA

O Protocolo de Comunicação OPC-DA (OLE for Process Control - Data Access) tem desempenhado um papel significativo na interconexão de sistemas na automação industrial, proporcionando uma estrutura para a troca eficiente de dados em ambientes de controle de processos. Conforme abordado por Silva (2019), o surgimento do OPC-DA representou uma extensão do modelo OLE (Object Linking and Embedding), visando superar desafios de interoperabilidade entre dispositivos diversos.

Embora o OPC-DA tenha sido amplamente adotado, ele não está isento de desafios. Silva (2019) destaca a ausência de suporte para funcionalidades avançadas, como segurança robusta e capacidade de lidar com tipos complexos de dados, impulsionando a transição para protocolos mais modernos, como o OPC-UA (OLE for Process Control - Unified Architecture).

2.7.2. ODBC

No complexo panorama dos sistemas de gerenciamento de bancos de dados relacionais, o Protocolo de Comunicação ODBC (Open Database Connectivity) emerge como uma peça central para a integração, proporcionando uma abordagem padronizada para a comunicação entre aplicativos e diversas fontes de dados. Conforme destacado por Hughes (2023) o ODBC atua como um facilitador da interoperabilidade, simplificando o acesso aos dados e promovendo a eficiência no desenvolvimento de software.

A arquitetura do ODBC, ainda conforme Hughes (2023) é fundamentada em camadas, estabelecendo uma interface que conecta aplicativos a diferentes tipos de bancos de dados. Essa estrutura modular oferece uma flexibilidade única, permitindo que aplicativos acessem dados independentemente da complexidade e localização dos bancos de dados. O ODBC utiliza drivers específicos para cada sistema de gerenciamento de banco de dados, garantindo uma integração eficaz em ambientes heterogêneos.

3 DESENVOLVIMENTO DO SISTEMA SUPERVISÓRIO PARA BCS

A interface comporta as informações operacionais da BCS por meio de um conjunto de recursos como medidores, visores, indicadores, interruptores e etc. Nela exibe-se as principais informações de desempenho da bomba como *head*, eficiência e potência, que serão fundamentais para o projeto da interface em questão, bem como o envelope de operação e a curva do sistema, exibe também as saídas do sistema (frequência, abertura da válvula choke,

pressão no reservatório e pressão no manifold) e as entradas (pressão na choke, vazão média, nível do anular e diferencial de pressão). Após a visão geral da versão final da interface na Figura 5, será realizado o detalhamento das partes componentes da interface, destacando suas funcionalidades.

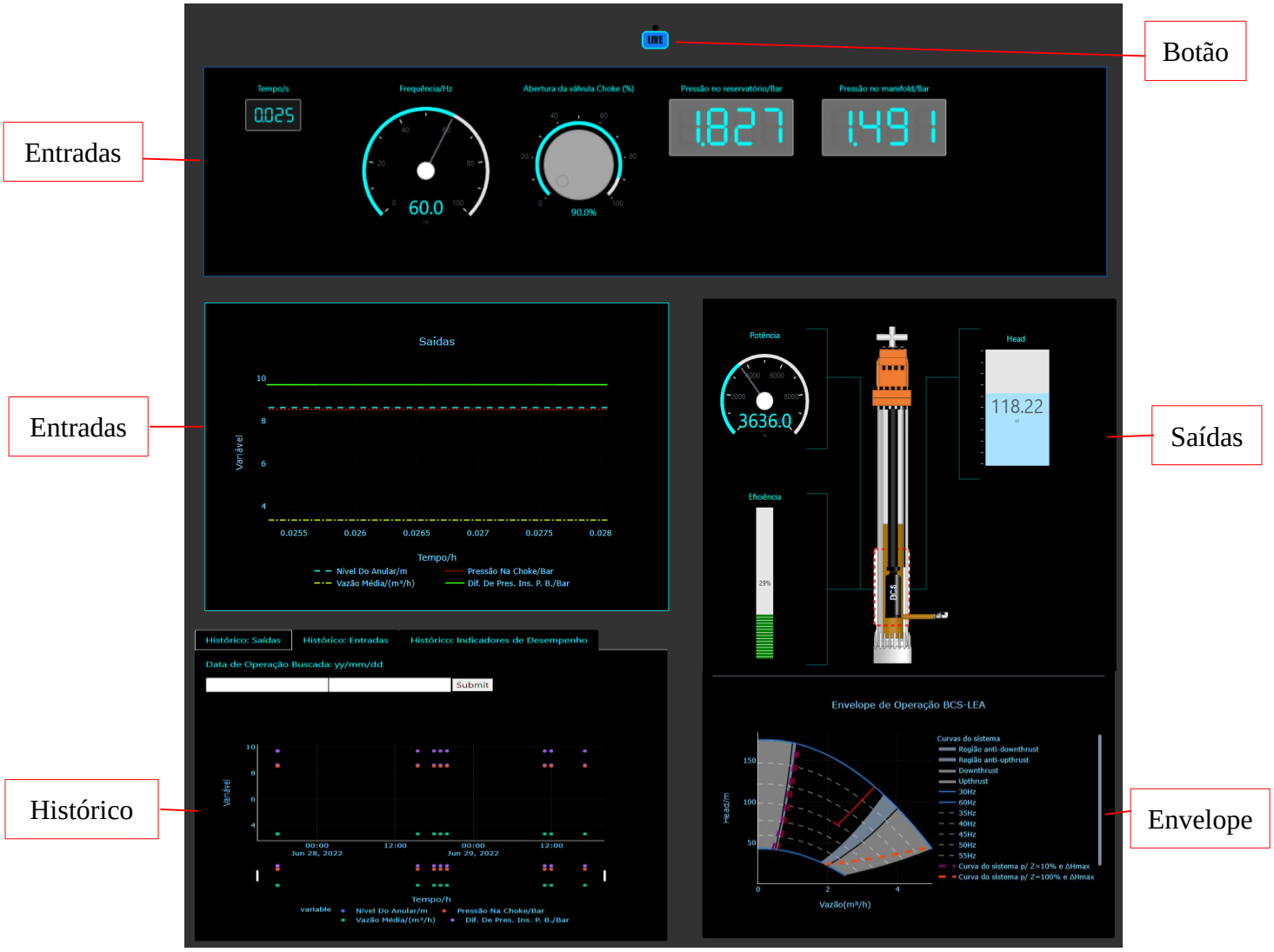

Figura 5: Página principal da interface

Uma aplicação relevante a ser destacada é que a interface dispõe da possibilidade de consulta em tempo real ao banco de dados para obtenção dos dados das variáveis de entrada, saída e de desempenho do sistema. Para começar a usar a interface em tempo real, basta clicar no botão 'ao vivo' localizado no centro superior da interface, conforme mostrado na Figura 4. Ao clicar neste botão, o sistema inicia automaticamente a busca por novos valores das variáveis

no banco de dados a cada segundo. Isso possibilita a exibição, na interface, de dados atualizados das variáveis, conforme ilustrado nas Figuras 6 a 8, no mesmo intervalo de tempo.

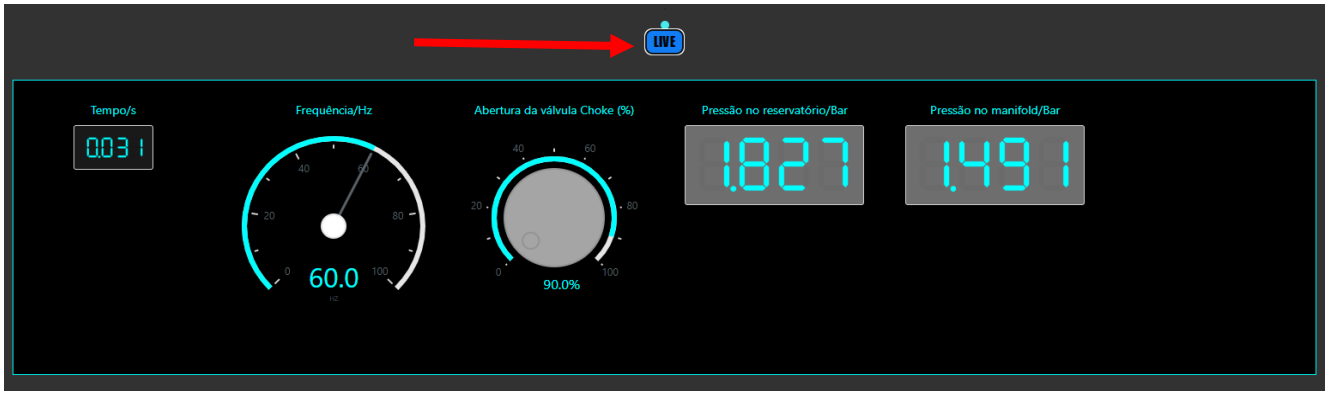

Figura 6: Dados das entradas e botão de ativação da aplicação em tempo real.

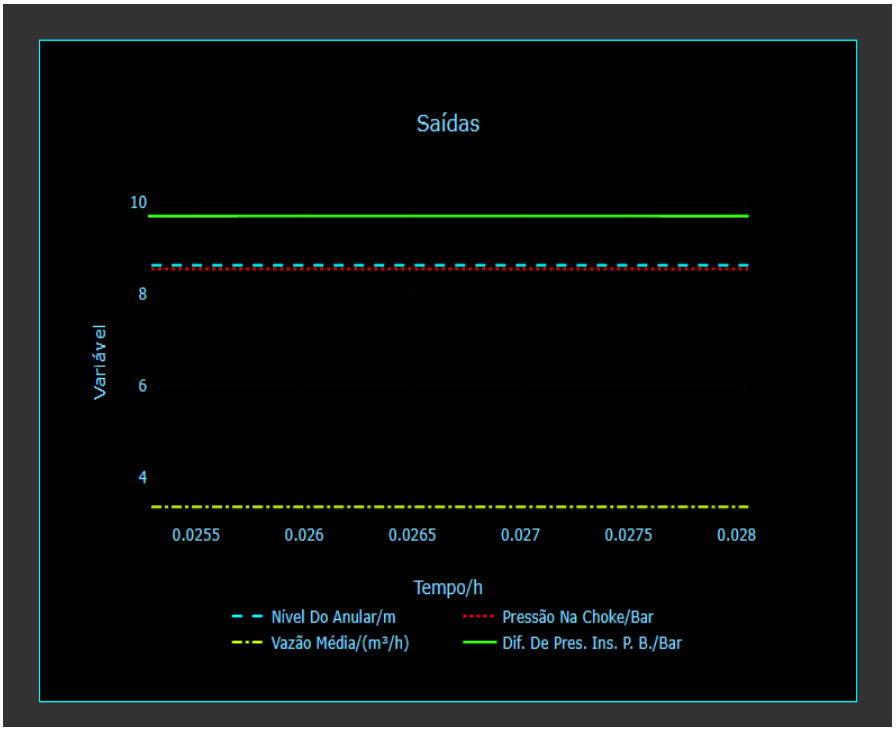

Figura 7: Exibição das saídas

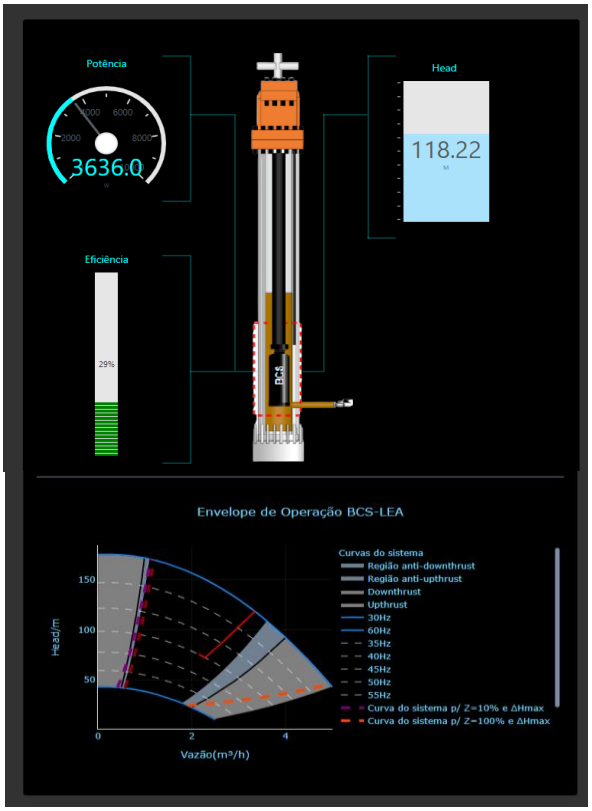

Figura 8: Indicadores de desempenho e envelope de operação da BCS.

Tendo em vista que a interface apresenta o uso em tempo real, fez-se necessário a inclusão da possibilidade de verificar o histórico de operações referente aos indicadores de desempenho, as entradas e as saídas do sistema, caso o usuário deseje informações de períodos anteriores, como pode ser observado nas Figuras 9 a 11.

Figura 9: Consulta ao histórico de saídas

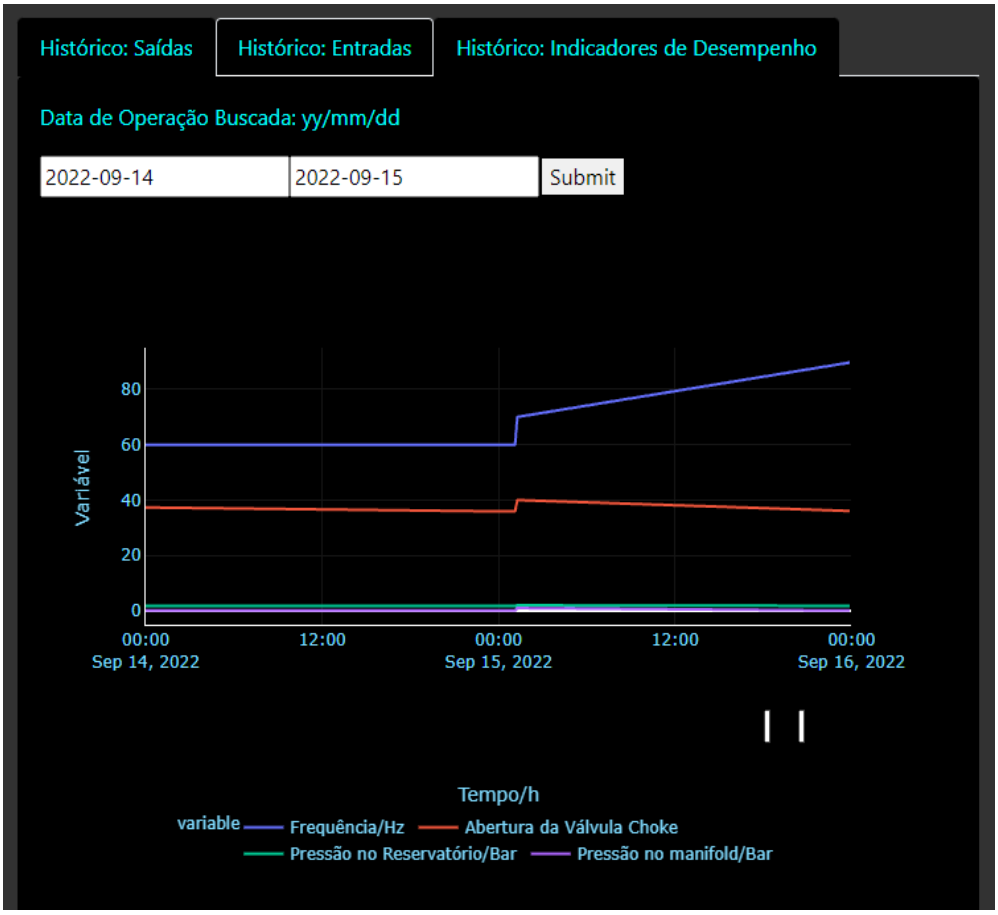

Figura 10: Consulta ao histórico de entradas.

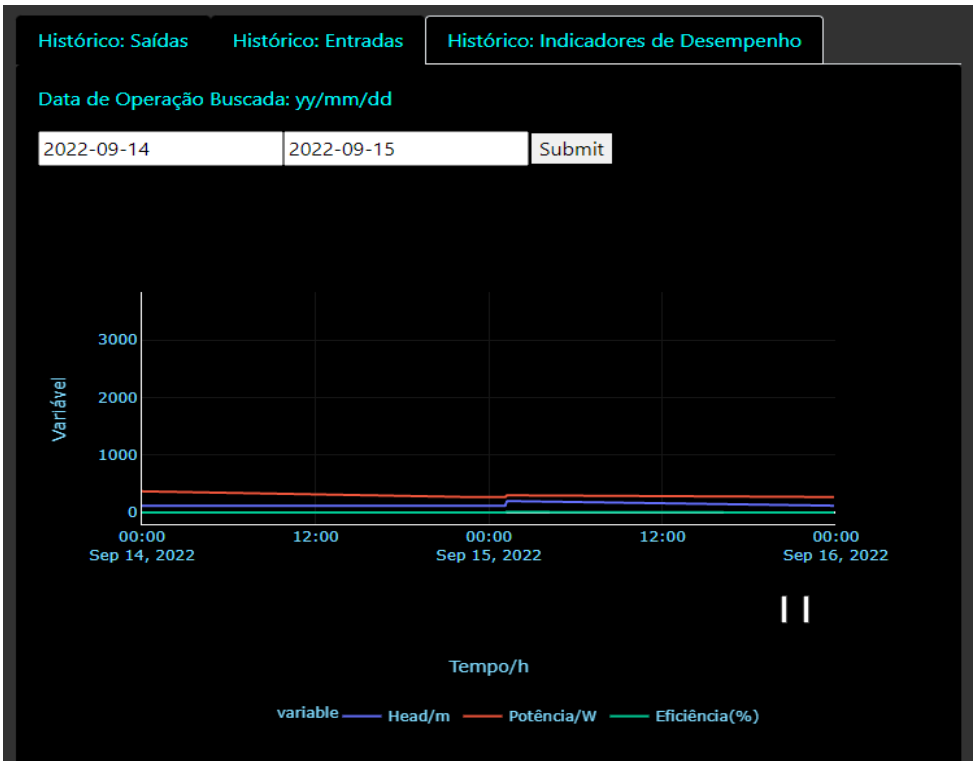

Figura 11: Consulta ao histórico dos indicadores de desempenho.

Sob o aspecto da exibição gráfica da interface, em relação a consulta de histórico, tem-se em uma região do gráfico a existência de três abas (Figura 9) contendo os gráficos com as informações das saídas, entradas e indicadores de desempenho. Dentre alguns recursos para manipulação e visualização dos dados, pode ser destacado a seção de manipulação dos gráficos mostrado na Figura 9 onde há opções como de zoom, seleção de curvas específicas e a opção de *download* dos gráficos.

4 PROJETO DA FERRAMENTA

4.1 CONTEXTUALIZAÇÃO

Para contextualizar o ambiente de utilização da interface proposta, faz-se necessário conhecer a estrutura física da planta, também destacando as variáveis envolvidas no processo de monitoramento da BCS.

O primeiro aspecto que pode-se destacar é a intalação da bomba, que foi feita pela empresa Baker Hughes. A BCS-LEA é composta por um sensor de fundo da série 450 (Centinel 3) e um motor da série 450 (FMHX) de 18 HP. O selo, o separador de gás e a bomba pertencem à série 400, sendo a bomba configurada com 15 estágios do modelo 15P4 PMXSSD (LIMA; COSTA; SANTOS; FONTANA; ABREU; SILVA., 2020).

Ainda de acordo com Lima, Costa, Santos, Fontana, Abreu e Silva. (2020), na configuração atual, a produção na cabeça do poço é monitorada por meio de instrumentos de pressão (PT-103), temperatura (TIT-103) e vazão (FT-102). Além disso, a planta possui uma válvula choke (FCV-102) que permite a realização de experimentos mais próximos da realidade, especialmente em poços com maiores profundidades. Na superfície, um tanque de óleo com capacidade de 1500L (LUBRAX- XP10) foi instalado para abastecer a coluna de produção. A Figura 12 fornece uma representação visual dos elementos mencionados na planta BCS e do sistema de aquisição de dados.

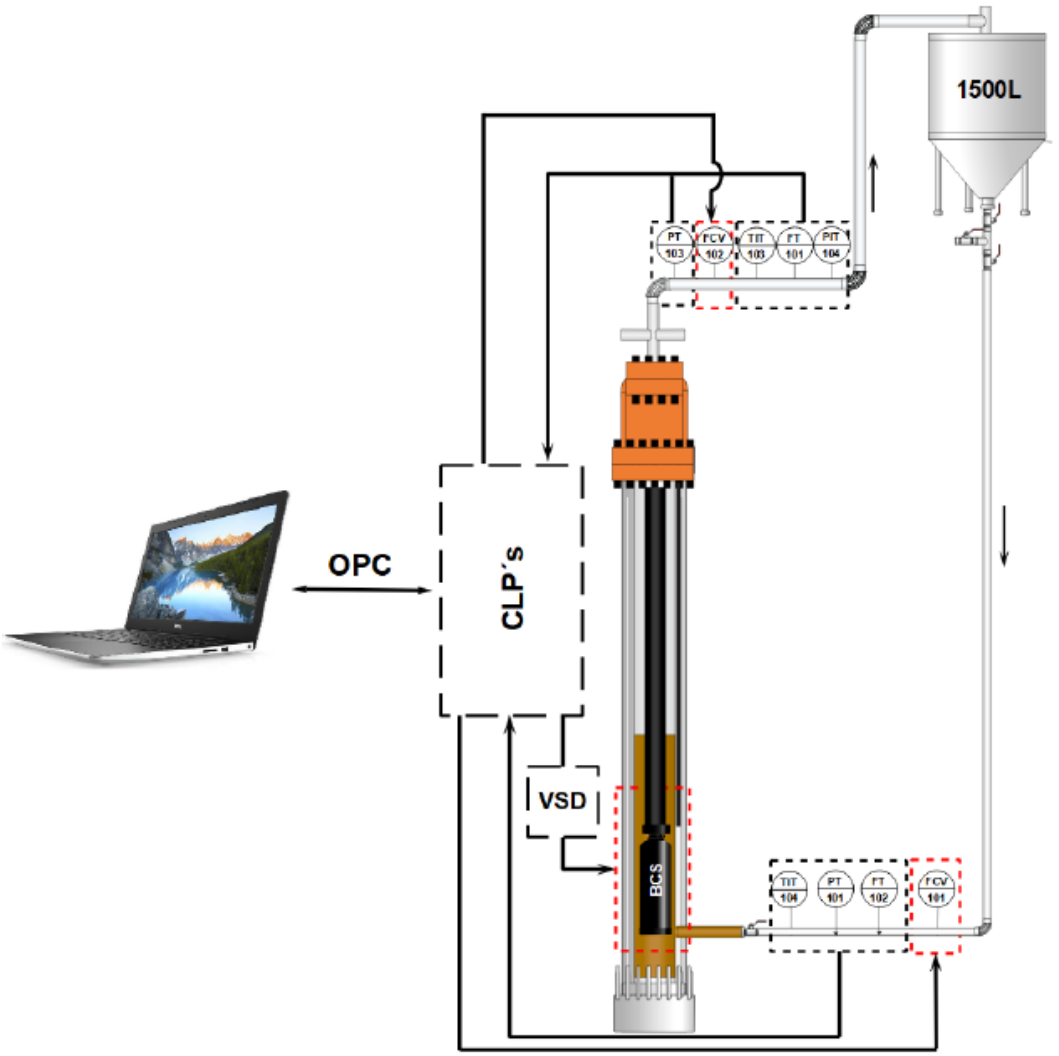

Figura 12: Representação da automação BCS LEA (LIMA; COSTA; SANTOS; FONTANA; ABREU; SILVA., 2020).

Conhecido o sistema no qual se deseja aplicar a ferramenta, vale destacar as variáveis de entradas e saídas pertinentes ao monitoramento da BCS, bem como seu envelope de operação, tais informações são destacadas na Figura 13 e 14.

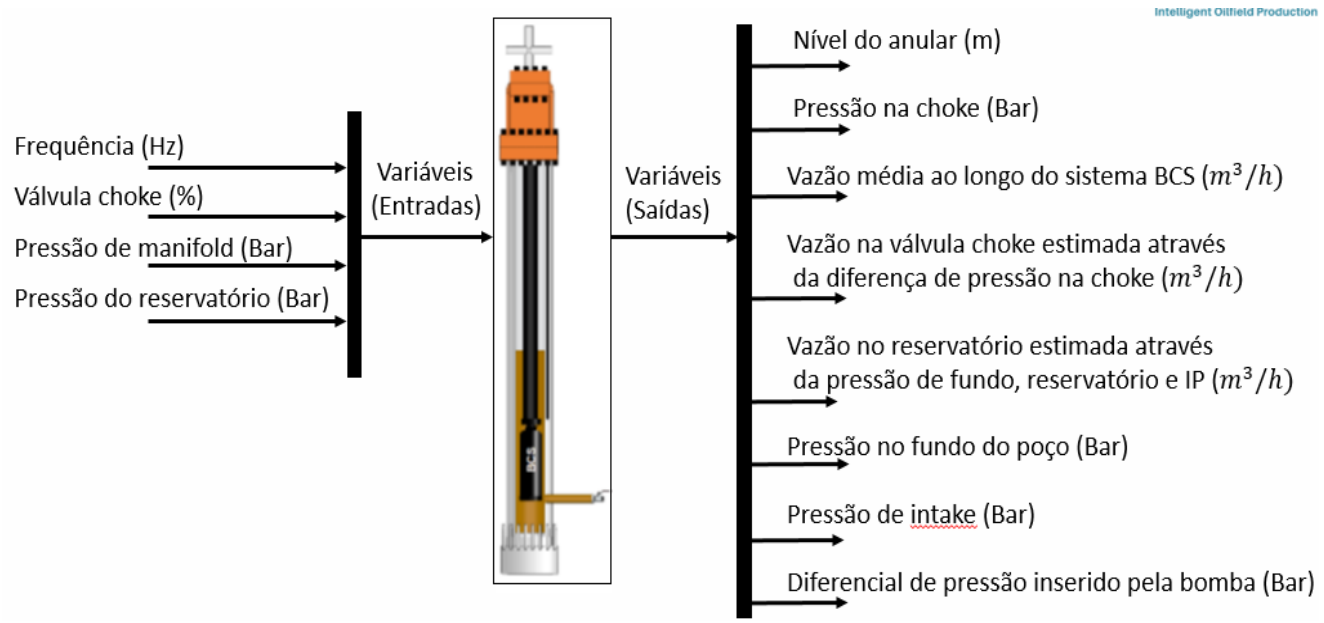

Figura 13: Variáveis de entradas e saídas da BCS (LIMA; COSTA; SANTOS; FONTANA; ABREU; SILVA., 2020).

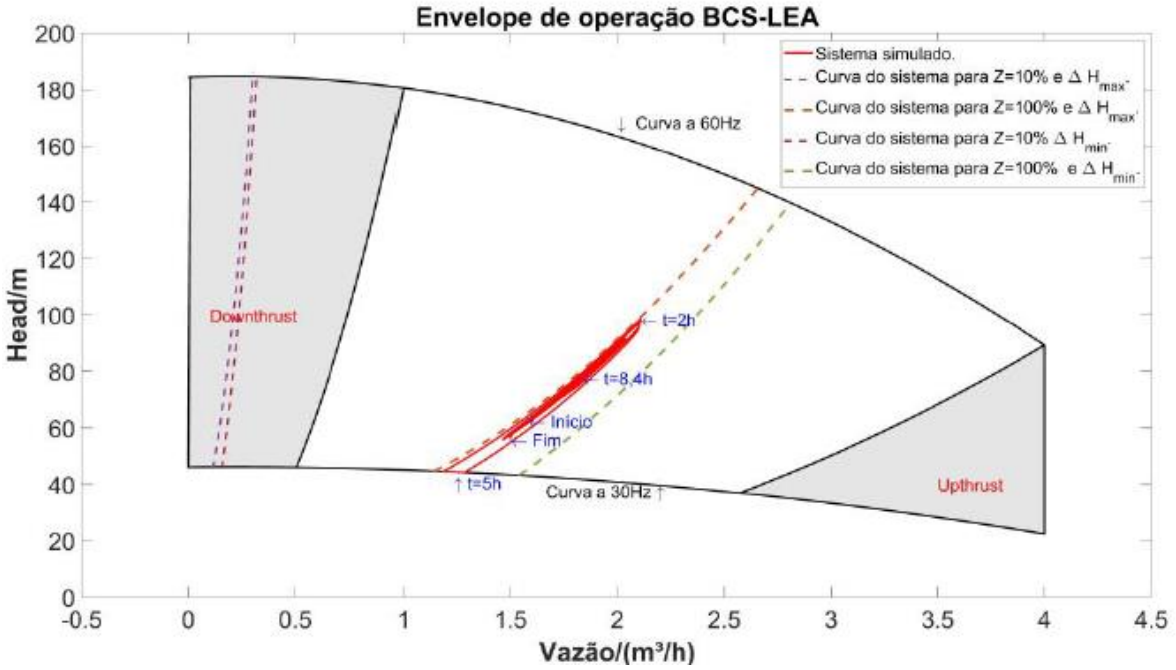

Figura 14: Exemplo do envelope de operação BCS-LEA (LIMA; COSTA; SANTOS; FONTANA; ABREU; SILVA., 2020).

É importante ressaltar que, para fins de construção e validação das aplicações da ferramente, fez-se uso de rotinas criadas em MatLab/Simulink, que simulam a operação da BCS-LEA, permitindo assim o acesso a dados gráficos e númericos, que viabilizaram a

programação da interface.

4.2 FERRAMENTAS

Para a execução da interface, destaca-se as seguintes ferramentas instaladas no computador: Banco de dados MariaDB, Python 3.9, o programa *Pycharm* para criação e execução de código em python, servidor OPC (*OLE for Process Control*) TOP Server 6.11 e o HeidiSQL para gerenciar o banco de dados e um navegador de internet.

Desta forma, se faz necessário projetar uma arquitetura de comunicação que define o fluxo de dados, considerando as coletas dos dados feitas pelos sensores da planta BCS, dados que são enviados para o servidor OPC e em seguida o banco de dados recebe os valores das variáveis enviadas por um cliente OPC, no final a interface irá fazer a aquisição dos dados automaticamente. Dito isto, a arquitetura da automação utilizada na composição da interface gráfica pode ser vista na Figura 15, onde os destaques em vermelho servem para explicitar o paralelismo entre o armazenamento dos dados e a consulta ao banco de dados, ou seja, estas etapas ocorrem de forma independente

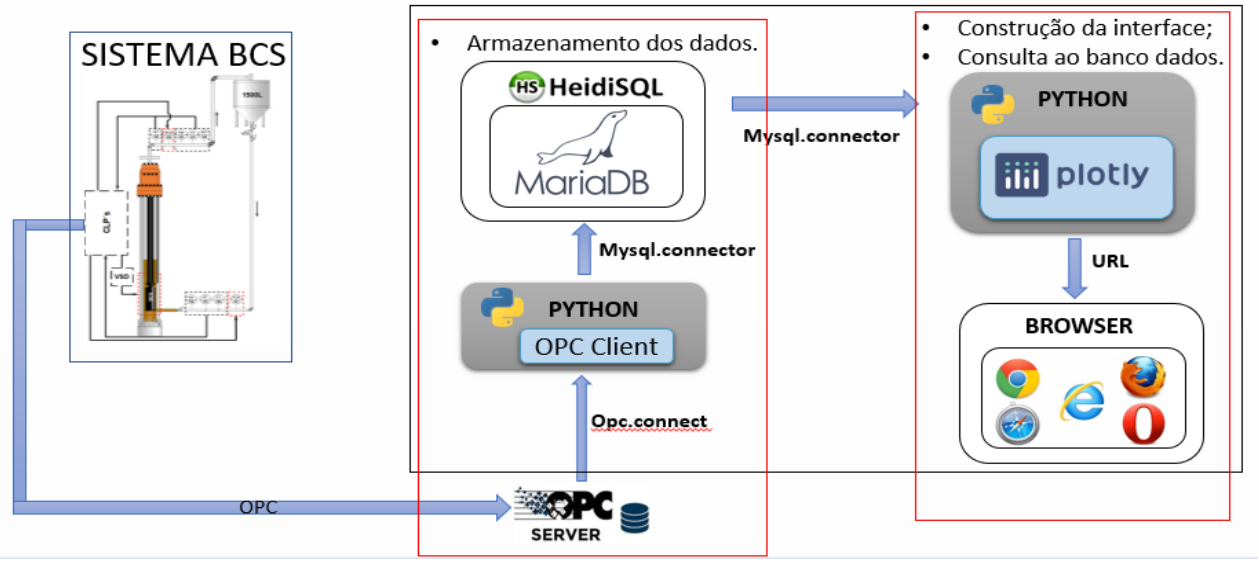

Figura 15: Arquitetura proposta do fluxo de dados.

Na prática, os dados podem ser escritos no banco em mais de uma maneira, ou seja, além do protocolo OPC, o registro no banco de dados pode ocorrer também via protocolo ODBC com o *software* MatLab, sendo provável também a existência de outros *softwares* que podem realizar este procedimento. Desta forma, vale destacar as funcionalidades e especificidades das ferramentas necessárias para execução e uso da ferramenta, como mostrado

a seguir.

- HeidiSQL: De acordo com Becker (2023), o HeidiSQL (Figura 16) é um software gratuito projetado para ser fácil de aprender. Com ele você pode visualizar e editar dados e estruturas em computadores que executam um dos sistemas de banco de dados, como MariaDB, MySQL, Microsoft SQL, PostgreSQL e SQLite. Desenvolvido por Ansgar em 2002, o HeidiSQL é uma das ferramentas mais populares para MariaDB e MySQL em todo o mundo.

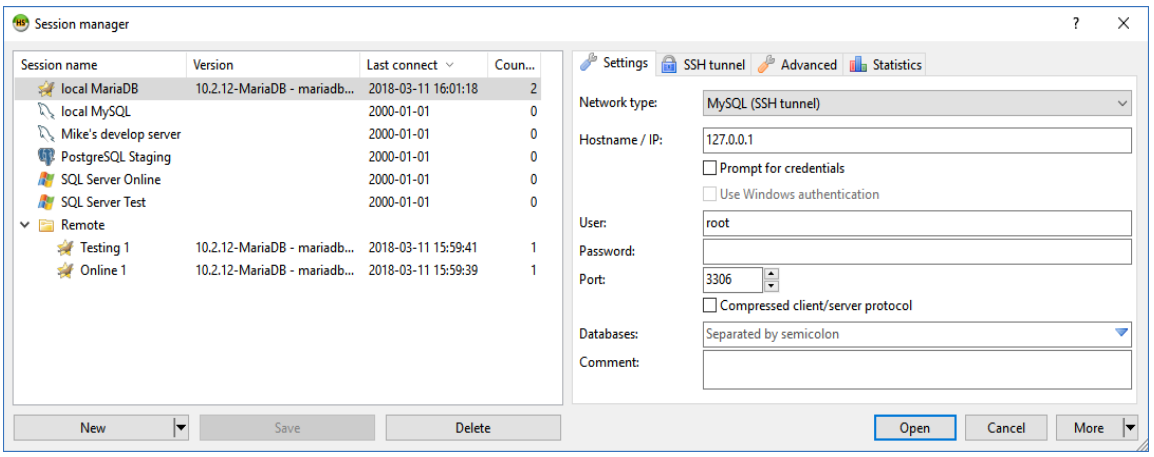

Figura 16: Interface de configuração do HeidiSQL (BECKER, 2023)

- Top Server 6.12: O TOP Server conecta seus vários sistemas de hardware a uma grande variedade de softwares de automação e controle industrial, incluindo, entre outros, HMI, SCADA, MES, Historians e muitos outros através de sua ampla gama de interfaces padrão, incluindo OPC UA, OPC DA, AVEVA SuiteLink (SOFTWARE TOOLBOX, 2023).

- PyCharm: Criada pela JetBrains, essa ferramenta oferece recursos como análise de código, depuração, e autocompletar, simplificando o processo de desenvolvimento de código (SILVA, 2023)

Ainda conforme Silva (2023), com esta IDE Python, vista na Figura 17, os desenvolvedores têm a flexibilidade de criar seus próprios processos, utilizando plugins e aproveitando as APIs disponíveis na plataforma. Além disso, é possível interagir diretamente com diversos bancos de dados sem sair da própria IDE, proporcionando uma experiência integrada e eficiente no desenvolvimento.

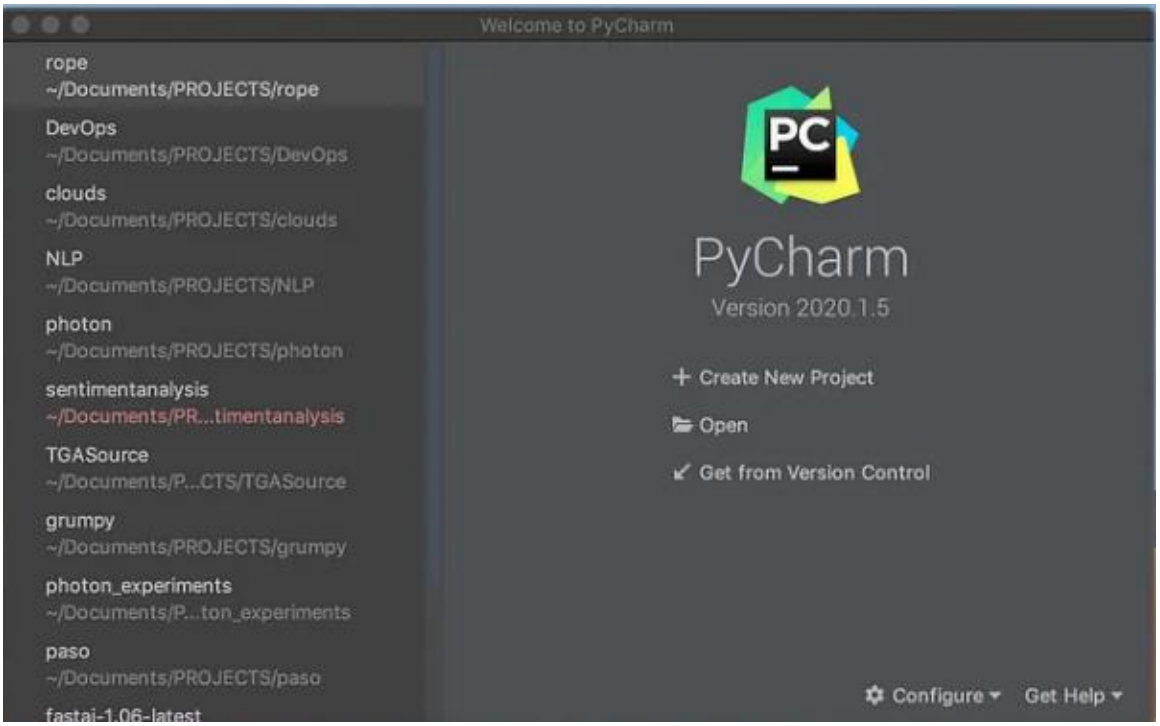

Figura 17: Interface Pycharm (COTTMAN, 2021)

- Navegadores: Para a realização deste trabalho, foi utilizado o Google Chrome e o Monzila, como ambientes para a renderização da interface.
- MariaDB: O MariaDB é um dos bancos de dados mais conhecidos do mundo, criado pelos mesmos desenvolvedores do MySQL, que mantiveram a estrutura de código aberto. Sua principal característica é a rapidez, escalabilidade e robustez de suas ferramentas, plugins e, claro, capacidade de armazenamento (SOUZA, 2020).

4.3 IMPLEMENTAÇÃO

Para dar início a construção da interface proposta neste trabalho, foi feito o uso de simulação do funcionamento dinâmico da BCS – LEA, no software MatLab/Simulink, através da utilização de uma rotina denominada de Principal, como mostrado na Figura 18, que emula o sistema em malha aberta (LIMA; COSTA; SANTOS; FONTANA; ABREU; SILVA., 2020).

Com o uso dessa simulação, foi possível obter informações coerentes relacionadas ao processo de extração realizado pela BCS - LEA. Os dados e as informações obtidas foram essenciais para viabilizar a construção da interface, pois permitiu que testes fossem executados e assim, posssibilitando a percepção dos melhores meios para condução e correção do projeto.

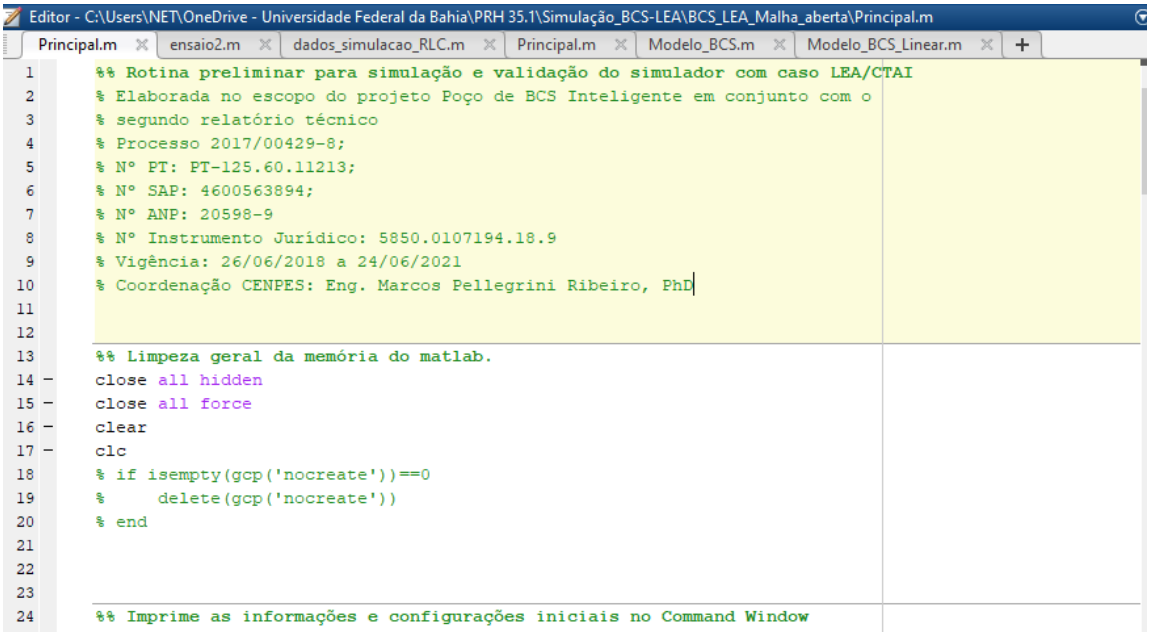

Figura 18: Simulador BCS – LEA.

Uma vez que é possível coletar os dados (simulados) da planta, dar-se início ao armazenamento desses dados com o intuito de consultá-los e no final exibíl-los na interface. O uso das informações obtidas após simulação tem como propósito validar o funcionamento do software desenvolvido, como exibições gráficas e numéricas, bem como a validação da estrutura de comunicação e fluxo de dados. A Listagem 1, mostra o código criado em MatLab que tem como função escrever no MariaDB (o banco de dados usado neste projeto) os dados obtidos na simulação.

```
time_delay = 1;

save Saidas;
load('Saidas.mat');

save Entradas;
load('Entradas.mat');

load("Envelope_BCS_LEA_Completo.mat");

cont=0
h=1000
conn=database('Localhost','root','23022021')
for c=1:h
    pause(time_delay)
    conn.AutoCommit = "on";

    Nivel_Do_Anular = Saidas(c,1).*P.L_poco*(randi(3)*0.5);
    Pressao_Na_Choke = Saidas(c,2).*P.DeltaP_max*(randi(3)*0.5);
    Vazao_Media_Ao_Longo_Do_Sistema_BCS = Saidas(c,3).*P.Delta_vazao*3.6*(randi(3)*0.5);
    Vazao_Na_Valv_Choke_Estimada_Atraves_Da_Diferenca_De_P_Na_Choke = Saidas(c,4)*3.6*(randi(3)*0.5);
    Vazao_No_Resertorio_Estimada_Atraves_Da_P_De_Fundo_Reser_E_IP = Saidas(c,5)*3.6*(randi(3)*0.5);
    Pressao_No_Fundo_Do_Poco = Saidas(c,6)*(randi(3)*0.5);
    Pressao_De_Intake = Saidas(c,7)*(randi(3)*0.5);
    Diferencial_De_Pressao_Inserido_Pela_Bomba= Saidas(c,8)*(randi(3)*0.5);

    %Tempo = dados.tempo_hora(1,c)
    %Tempo = Tempo'
    Datatime = datetime();
    saidas_x_tempo = table(Nivel_Do_Anular,Pressao_Na_Choke,Vazao_Media_Ao_Longo_Do_Sistema_BCS,...
    Vazao_Na_Valv_Choke_Estimada_Atraves_Da_Diferenca_De_P_Na_Choke,Vazao_No_Resertorio_Estimada_Atraves_Da_P_De_Fundo_Reser_E_IP,
    Pressao_No_Fundo_Do_Poco,Pressao_De_Intake,Diferencial_De_Pressao_Inserido_Pela_Bomba,Datatime)
    d=2;
    Frequencia = Entradas.Frequencia(c,1)*(randi(3)*0.5);
    Abertura_valvula_choke  = Entradas.Abertura_valvula_choke(c,1)*100*(randi(3)*0.5);
    Pressao_reservatorio  = fix(Entradas.Pressao_reservatorio(c,1)*10^d)/10^d*(randi(3)*0.5);
    Pressao_manifold = fix(Entradas.Pressao_manifold(c,1)*10^d)/10^d*(randi(3)*0.5);
    T_entradas = table(Frequencia,Abertura_valvula_choke,Pressao_reservatorio,Pressao_manifold,Datatime)

    Head =  dados.simulacao.Head(c,1)*(randi(3)*0.5); %[m]
    BHP = Saidas(c,1)*1000*(randi(3)*0.5); % PotÃªncia W
    Eficiencia = Saidas(c,1); % EficiÃªncia Procentagem

    desempenho = table(Head,BHP,Eficiencia,Datatime)
    tablename3 = "desempenho"
    sqlwrite(conn,tablename3,desempenho)

    tablename1 = "entradas";
    sqlwrite(conn,tablename1,T_entradas)

    tablename = "saidas_x_tempo";
    sqlwrite(conn,tablename,saidas_x_tempo)
    commit(conn)
    cont=cont+1
end
close(conn)
```

Listagem 1: Armazenamento dos dados no MariaDB.

Na Listagem 1, é mostrado o código utilizado para fazer as inserções das informações no banco de dados através do protocolo de comunicação ODBC. Com isso foi possível testar vários

recursos para a contrução da interface, inclusive a consulta em tempo real ao banco de dados feita pela interface, pois na listagem acima, simula-se a inserção desses dados a cada 1 (um) segundo com o intuito de representar o cenário real de coleta dos valores das varáveis.

Na construção a interface através do python, o framework plotly desempenha um papel central. De forma objetiva será destacada os principais recursos utilizados na construção dainterface.

- ESTRUTURA DO CÓDIGO:

O layout do código para a criação dos gráficos e suas estilizações é composto pela biblioteca Dash, do qual obtem-se o Dash HTML Components (que simplificou a utilização do HTML) e o Dash Core Components (usado para implementar recursos de interatividade com o usuário). As bibliotecas Plotly Express e Graph Objetcs foram usadas para aconstrução dos gráficos. Na Figura 19, tem-se uma representação do ecossitema plotly.

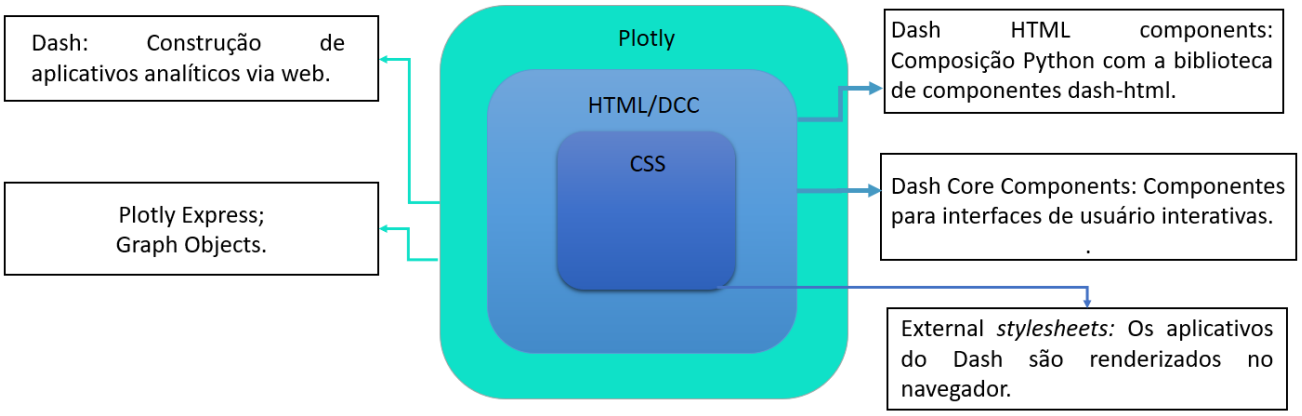

Figura 19: Ecossistema Plotly.

Para a estilização da interface, foi utilizado o estilo CSS próprio do plotly que é o `dbc.themes.BOOTSTRAP`. O plotly permite as principais funcionalidades do Bootstrap em um aplicativo web, simplificando assim o desenvolvimento de aplicativos web (NOKERI, 2023). A Listagem 2 mostra o emprego da estilização na a interface através da variável `app`.

```
# O CSS é um mecanismo para adicionar estilo a um documento web.
# CSS (Cascading Style Sheets ou Folhas de Estilo em Cascata) é uma linguagem de estilo (en-US)
# usada para descrever a apresentação de um documento escrito em HTML ou em XML (incluindo várias linguagens em
# XML como SVG, MathML ou XHTML).

app = dash.Dash(__name__, external_stylesheets=[dbc.themes.BOOTSTRAP])
```

Listagem 2: Estilização da interface.

- CONSULTA AO BANCO DE DADOS

Para para conectar a interface ao banco de dados MariaDB e consultar os valores das variáveis contidas nele, foi feito o uso da biblioteca `mysql.connector` (BAWEJA, 2023). Como mostra a Listagem 3, os parâmetros definidos neste trabalho para para acessar o banco foram o nome do usuário (`user`), a senha para permitir a consulta (`passoword`), o número do IP em `host` e o nome do banco de dados utilizado (`database`).

```
mydb = mysql.connector.connect(
    user="root",
    password="23022021",
    host="127.0.0.1",
    port=3306,
    database='database')
```

Listagem 3: Conexão com banco de dados.

Uma vez que a conexão entre a interface e o banco de dados estão estabelecidas, a próxima etapa é a consulta dos dados. Na Listagem 4, tem-se a consulta (`query_x`) das variáveis de entrada da BCS. Vale destacar que as demais variáveis serão consultadas de mesma forma.

```
query_x = "SELECT * FROM entradas"
df_x = pd.read_sql(query_x, con=mydb)
df_x.columns = ["Frequência/Hz", "Abertura da Válvula Choke",
                "Pressão no Reservatório/Bar",
                "Pressão no manifold/Bar", "Datatime"]
```

Listagem 4: Consulta ao banco de dados.

- CONSTRUÇÃO DOS GRÁFICOS

Compondo uma das partes essenciais do trabalho, a elaboraão dos gráficos se dão através do uso de duas das pricnipais bibliotecas: `plotly.express` e `plotly.graph.objects`. Como pode ser visto na Listagem 5, onde tem-se a criação dos gráficos de desempenho da BCS através do uso do `plotly.graph.objects` (PARMER, 2023).

Cada gráfico é gerado de forma independente para depois serem mostrados em conjunto no mesmo plano. A Listagem 5, mostra a definição dos parâmetros, como varável independete e dependente (`x` e `y`), o nome do gráfico (`name`) e o tipo de plotagem gráfica (`line`) que é passada como um dicionário em python.

```
# Gerando os gráficos desempenho
fig3=go.Figure()
fig3.add_trace(go.Scatter(x=df_3["Datatime"],
                          y=df_3["Head/m"],
                          name="Head/m",
                          line = dict(color='green', width=3, dash='dash')))
fig3.add_trace(go.Scatter(x=df_3["Datatime"],
                          y=df_3["Potência/W"],
                          name="Potência/W",
                          mode = "lines",
                          line = dict(color='cyan', width=2, dash='dashdot')))
fig3.add_trace(go.Scatter(x=df_3["Datatime"],
                          y=df_3["Eficiência(%)"],
                          name='Eficiência(%)',
                          line = dict(color='red', width=3, dash='solid')))
```

Listagem 5: Construção gráfica do desempenho da BCS.

Já em relação a construção de gráficos através da biblioteca `plotly.express`, seu uso está associado a biblioteca Dash Core Components (DCC), pois possibilita a interação entre o usuário e a ferramenta (KRUCHTEN, 2023). No caso mostrado na Listagem 6, tem-se a consulta ao banco de dados para obtenção de hitórico de operação e a construção do gráfico de saídas da BCS dentro de uma função chamada `update_figure`. Além dos aspectos comuns aos gráficos, como título e legenda, ao se tratar de consulta a histórido de dados, foi necessário incluir entradas na interface (*inputs*) que recebem datas relacionadas a periodos desejados para consulta dos dado solicitados, neste caso, `print(input1)` e `print(input2)`. As variáveis relcionadas a desempenho e entradas da BCS seguem também modelo análogo de construção para consulta ao histórico de dados.

```
def update_figure(n_clicks, input1, input2):
    print(input1)
    print(input2)
    query_001 = "SELECT * FROM saidas_x_tempo"
    df_001 = pd.read_sql(query_001, con=mydb)
    df_001.columns = ["Nível Do Anular/m", "Pressão Na Choke/Bar", "Vazão
Média/(m³/h)",
                      "Vazão Na Válvula Choke/(m³/h)",
                      "Vazão No Reser. Est. Através Da P. De Fundo Reser. E
IP/(m³/h)", "Pressão No Fundo Do Poço/Bar",
                      "Pressão De Intake/Bar", "Dif. De Pres. Ins. P. B./Bar",
"Datatime", "id"]
    fig001 = px.line(df_001, x='Datatime',
                     y=["Nível Do Anular/m", "Pressão Na Choke/Bar", "Vazão
Média/(m³/h)",
                        "Dif. De Pres. Ins. P. B./Bar"],
                     range_x=[str(input1)+' 00:00:00', str(input2)+' 23:59:59'])
    fig001.update_yaxes(showline=True, gridcolor='#161616')
    fig001.update_xaxes(rangeslider_visible=True, showline=True,
gridcolor='#161616')
    fig001.update_layout(
        plot_bgcolor=colors['background'],
        paper_bgcolor=colors['background'],
        font_color=colors['text'],
        legend=dict(orientation="h",yanchor="bottom",y=-
0.90,xanchor="right",x=0.90
        ),
        # height=350,
        xaxis_title="Tempo/h",
        yaxis_title="Variável",
    )
    return fig001
```

Listagem 6: Gráfico para consulta do histórico de operação.

Ainda relacionado com a busca do histórico de dados e a inserção de informações associadas a essa busca, tem-se a confirguração das janelas de entradas das datas para a consulta do histórico. A Listagem 7 e a figura 20 mostram a construção e a exibição dessas informações, respectivamente.

```
# Janela de entrada da data
number_input = html.Div(
    [
        html.P("Data de Operação Buscada: yy/mm/dd",style={'color':'cyan'}),
        dcc.Input(id='input', type='text'),
        dcc.Input(id='input-2-state', type='text'),
        html.Button(id='submit-button-state', n_clicks=0, children='Submit'),
        html.Br(),
        html.P(id="styled-numeric-input"),
    ],
),
```

Listagem 7: Construção das janelas de entradas das datas.

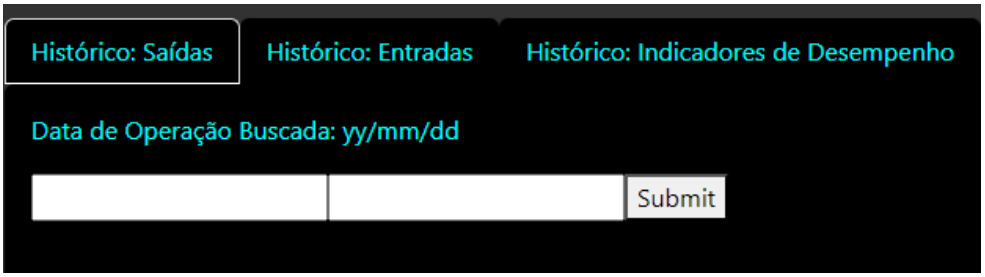

Figura 20: Janelas de entradas para as datas.

Em sintese, pode-se observar de forma esquemática na Figura 21, a maneira como se dá o tratamento dos dados inseridos no layout da interface: 1° Insere-se as datas, 2° A interface consulta os dados referentes ao periodo digitado, 3° É gerada as curvas correspondentes ao período digitado, 4° Por fim, a figura gerada vai para o layout da interface.

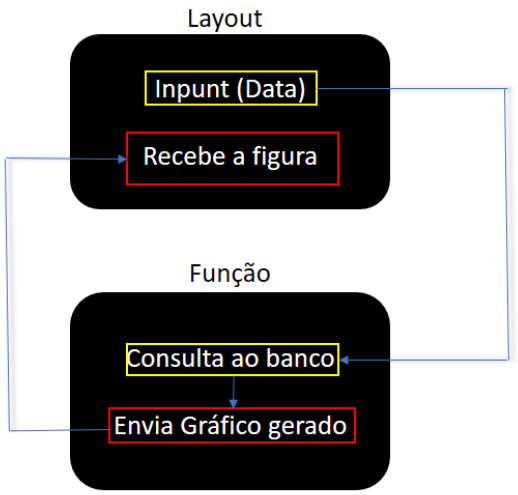

Figura 21: Algorítimo de consulta aos histórico de dados.

- CONCSULTA AO BANCO DE DADOS EM TEMPO REAL

A interface possibilita a consulta dos dados em tempo real através do módulo `dcc.Interval`, mostrado na Listagem 8. Tendo como principais parâmetros os `id` que identifica o temporizador usado para uso em tempo real, bem como o intervalo (`interval`) de tempo utilizados nas buscas que a interface faz no banco de dados, neste caso, 1000 (mil) ms ou 1 (um) s.

```
dcc.Interval(
    id='interval_extendablegraph_update',
    interval=1000,
    n_intervals=0,
    disabled=True,
    max_intervals=-1),
```

Listagem 8: Definição do intervalo de tempo para busca no banco de dados.

Com o intervalo de busca definido, a função `update_extendData` faz uso deste intervalo através do parâmetro `n_intervals`. Com isso é possível fazer as buscas (query) a cada 1 (um) s das variáveis desejadas como mostra a Listagem 9.

```
# Exibe os dados em realtime conservando pontos anteriores numa janela de 10
dados/s para cada variável
def update_extendData(n_intervals, existing):
    df_0 = get_value(n_intervals)
    x_new  = float(df_0["id"])
    y_new1 = float(df_0["Nível Do Anular/m"])
    y_new2 = float(df_0["Pressão Na Choke/Bar"])
    y_new3 = float(df_0["Vazão Média Ao Longo Do Sistema BCS/(m³/h)"])
    y_new4 = float(df_0["Dif. De Pressao Inserido Pela Bomba/Bar"])
    return [[dict(x=[x_new], y=[y_new1]),dict(x=[x_new], y=[y_new2]),
             dict(x=[x_new], y=[y_new3]),dict(x=[x_new], y=[y_new4])],
            [0,1,2,3,4],10], n_intervals
```

Listagem 9: Consulta automática ao banco de dados

Uma parte importante para ser destacada juntamente com o acesso em tempo real ao banco de dados é o uso de uma biblioteca chamada `dash_extendable_graph` (LIANG, 2023). Ela permite exibir os dados gráficos na interface em uma janela de quantidade de dados, ou seja, na Listagem 9, em `return`, observa-se o número 10 dentro do dicionário, isso quer dizer que a cada 10 pontos exibidos no gráfico, é atulizada para uma nova janela de outros 10 novos dados e assim sucessivamente, isso é útil para evitar o condensamento de pontos exibidos no gráfico. Pode ser observado na Listagem 10, a construção do gráfico de saídas da BCS em tempo real com o uso do módulo `deg.ExtendableGraph` associado a biblioteca anteriormente citada.

```
deg.ExtendableGraph(# https://plotly.com/python/figure-introspection/#accessing-
javascriptcomputed-defaults
      id='extendablegraph_example',
      figure={'layout': {'title': 'Saídas', 'barmode': 'overlay','xaxis':
{'anchor': 'y', 'title': {'text': 'Tempo/h'}},
                         'yaxis': {'anchor': 'x', 'title': {'text': 'Variável'}},
                         'legend': dict(orientation="h", yanchor="bottom", y=-
0.43,xanchor="right", x=0.90),
                         'plot_bgcolor': colors['background'],'paper_bgcolor':
colors['background'],
                         'font': {'color': colors['text']},
          },
              'data': [{'x': [], 'y': [], 'mode': 'lines',
                        'line': dict(color='cyan', width=2,
dash='dash'),'showlegend': True, 'name': 'Nível Do Anular/m'},
                       {'x': [], 'y': [], 'mode': 'lines','line':
dict(color='red', width=2, dash='dot'),'showlegend': True, 'name': "Pressão Na
Choke/Bar"},
                       {'x': [], 'y': [], 'mode': 'lines','line':
dict(color='#c7ff00', width=2, dash='dashdot'),'showlegend': True, 'name': "Vazão
Média/(m³/h)"},
                       {'x': [], 'y': [], 'mode': 'lines','line':
dict(color='#39ff14', width=2, dash='solid'),'showlegend': True, 'name': "Dif. De
Pres. Ins. P. B./Bar"}
                ],
      },
),)
```

Listagem 10: Construção do gráfico para uso em tempo real.

O plotly oferece a possibilidade de inserir um botão com o objetivo de usá-lo para ativar e desativar o modo de busca em tempo real. Na Listagem 11, tem-se a utilização de mais um dos módulos da biblioteca Dash, neste caso, o `html.Button`, onde pode-se configurar a parte estética (frontend) do botão que aparecerá na interface. Já a função `toggle_interval` apresentada na listagem 12, atua no reconhecimento da ativação ou desativação do botão, desencadeando as alterações nos modos de busca dos dados, onde, dependendo do valor de `n` (cliques no botão), a função retorna a ativação (`not disabled`) ou desativação (`disabled`).

```
html.Button("LIVE", id="button",n_clicks=0,
            style={'background-color': '#127cf4', 'border': '2px #127cf4 solid',
                   'borderRadius': 8,
                   'font-family': 'Impact', 'display': 'block', 'margin': 'auto
auto 10px auto',
                   'border-color':'cyan','width': '45px'}),
```

Listagem 11: Configuração do botão.

```
def toggle_interval(n, disabled):
    if n:
        return not disabled
    else:
        return disabled
    return disabled
```

Listagem 12: Reconhecimento da ativação ou desativação do botão.

A exibição em tempo real das informações de desempenho (Head, eficiência e potência) e entradas da BCS se dão de uma forma diferente em relação as saídas. Ao inés do uso de gráficos, optou-se pela a escolha de recursos (*features*), possibilitados pela biblioteca `dash_daq` (JOHNSON, 2023). Esta bilbiloteca possui alguns recursos visuais, os quais foram utilizados na interface, como: Gauge (manômetro), GraduatedBar (Barra graduada), Knob, Tank (tanque) e entre outros. A Listagem 13, mostra a construção do recurso, que indica abertura da válvula *choke* em tempo real, nele é possível notar os parâmetros de configuração, entre eles o `id`, valor máximo e mínimo para exibição (`max`, `min`), o `label`, para formatação do título entre outros.

```
daq.Knob(
        id='knob-1',
        size=140,
        #value=3,
        max=100,
        min=0,
        color='cyan',
        disabled=True,
        label={
            'label': 'Abertura da válvula Choke (%)',
            'style': {
                'color': 'cyan',
                'fontSize': 14
            }
        },

    ), md=2, className='mt-4',
),
```

Listagem 13: Construção do indicador da entrada Abertura da Válvula *Choke*.

De modo geral, a construção dos recursos do `Dash_daq` seguem o mesmo padrão, isso se aplica as demais variáveis de entradas que são exibidas na interface, bem como aos indicadores de desempenho, como se vê na Listagem 14, onde tem-se o exemplo da construção do indicador de desempenho *head*.

```
daq.Tank(
            id='mtanjok',
            label={
                'label': 'Head',
                'style': {
                    'color': 'cyan',
                    'fontSize': 14
                }
            },
            min=0,
            max=190,
            value=0,
            units='m',
            showCurrentValue=True,
            style={'margin-left': '550px', 'margin-top': '-540px'},
```

Listagem 14: Construção do indicador do desempenho *head*.

- TESTE NO SERVIDOR DO LEA-CTAI

Para dar prosseguimento aos testes e ensaios com a interface, fez-se necessário estabelecer a arquitetura utilizada para o fluxo de dados do processo. Para isto foi preciso maior aprofundamento no estudo e aplicação do padrão de comunicação OPC, mais especificamente, o OPC-DA (Data Access), pois a BCS-LEA utiliza este tipo de comunicação. Assim, a obtenção dos dados da planta passa por um OPC Server que encaminha as informações para um OPC Client, em que estas informações serão obtidas e usadas pela interface. Como a interface é construída em Python, através do framework Plotly, é preciso estabelecer previamente a comunicação entre o Python e o OPC-DA, para isto, o Python dispõe de uma biblioteca chamada OpenOPC, que realiza esta comunicação (KWIATKOWSKI, 2023). Uma vez conectados, é possível construir o código que escreve os dados obtidos diretamente da planta no banco de dados como mostra a Listagem 15.

```
Import time
from opcua import Client, Node
from datetime import datetime, date
import mysql.connector
import pandas as pd
import mariadb
import openOPC

mydb=mariadb.connect(user="root", password="23022021",host="127.0.0.1",port=3306,
database='database')
class SubscriptionHandler:
    def datachange_notification(self, node: Node, myvar, data):
        mycursor = mydb.cursor()
        d=datetime.now().strftime("%Y%m%d%H%M%S")
        #d=int(d)/3600
        t=datetime.now().strftime("%S")
        mycursor.execute("INSERT INTO teste_escrita (y, data) VALUES (" +
str(myvar) + "," + str(d) + ")")
        mydb.commit()
        print(str(datetime.now().strftime("%Y-%m-%d %H:%M:%S %p")) + " : " +
str(node) + " : " + str(myvar))

client = Client("opc.tcp://DESKTOP-96QLF5V:49380")
client.connect()

# myvar = client.get_node("ns=2;s=Endress.Bancada.SWG70.Adaptador_02.TT-
003.Device.Dynamic_Variables.PV.Value")
# myvar2 = client.get_node("ns=2;s=Endress.Bancada.SWG70.Adaptador_04.PIT-
008.Device.Dynamic_Variables.PV.Value")
myvar = client.get_node("ns=2;s=Channel1._Statistics._FailedReads")


handler = SubscriptionHandler()
sub = client.create_subscription(1000, handler)
handle = sub.subscribe_data_change(myvar)

time.sleep(3600)

sub.unsubscribe(handle)
client.disconnect()
```

Listagem 15: Código para a coleta dos dados da planta para o banco de dados via OPC.

Utilizando a estrutura do LEA - CTAI, foi possível conduzir um teste experimental da conexão de cliente OPC-DA construído em Python com o servidor OPC-DA do próprio LEA, através de um experimento em bancada, usando uma BCS instalada horizontalmente. Com isso, foi possível confirmar a funcionalidade da conexão, bem como do registro dos dados obtidos da planta diretamente no banco de dados de forma automática. As Figuras 22 à 24 mostram parte da realização deste teste.

Figura 22: Parte do hardware utilizado.

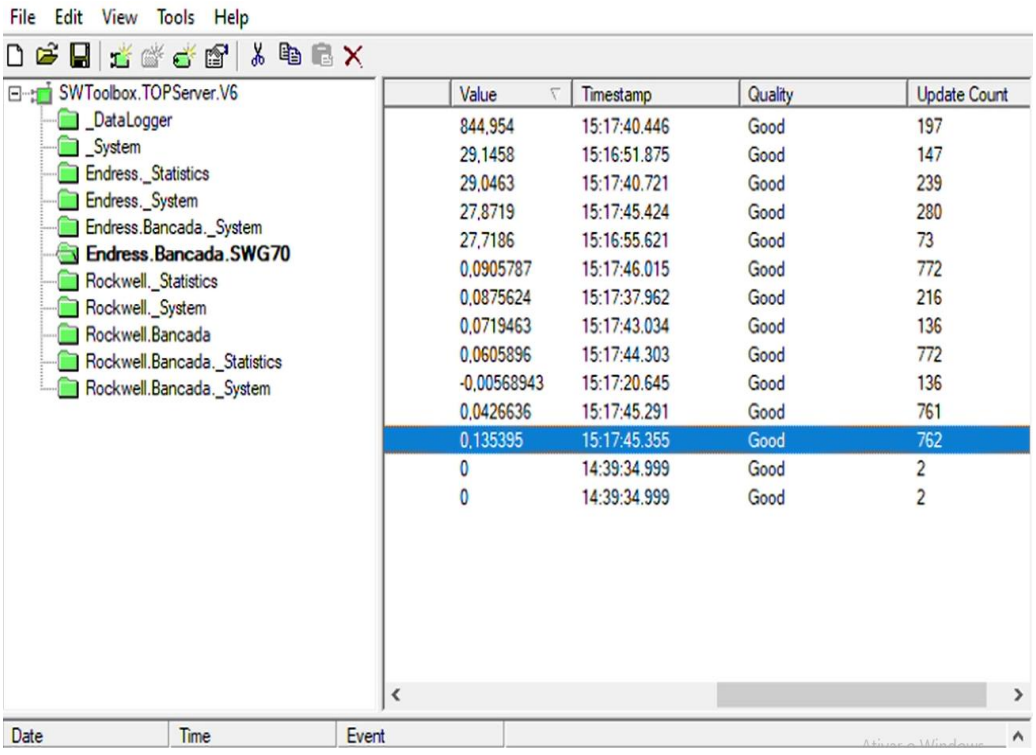

Figura 23: Uso do Top Server durante o teste.

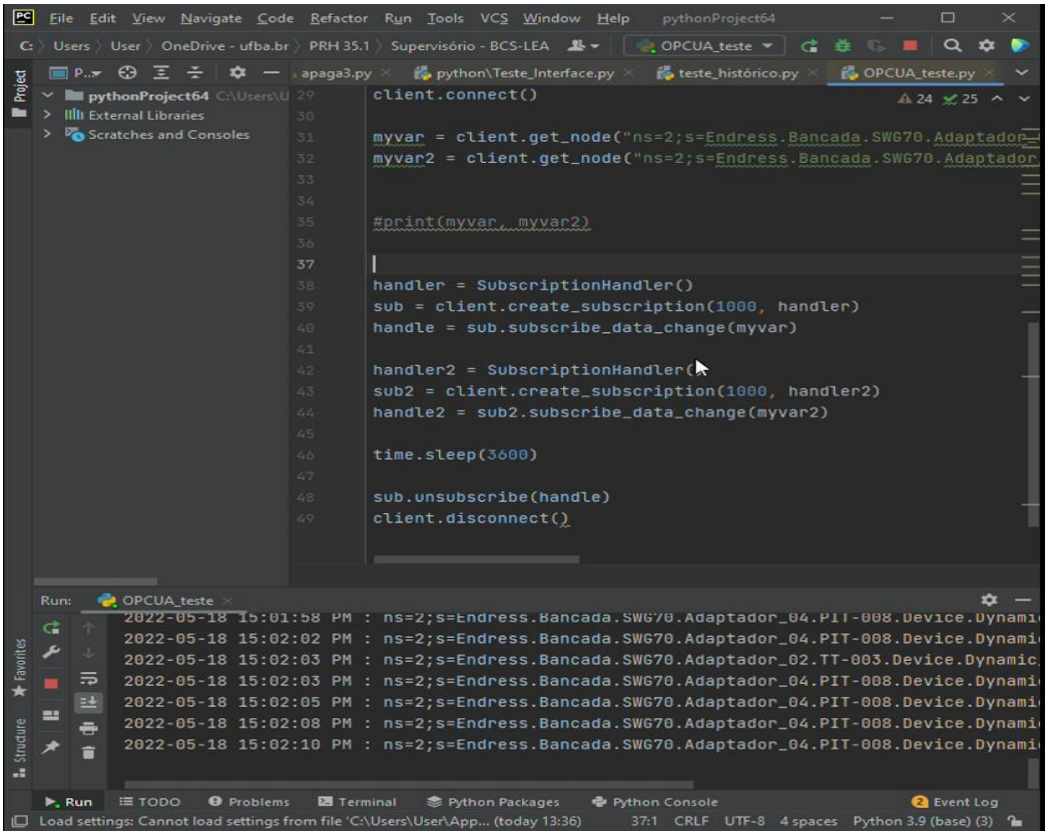

Figura 24: Uso da conecção entre o python e oprotocolo OPC durante o teste.

4.4 LIMITAÇÕES COMPUTACIONAIS

- Ainda não é possível compilar o código em outras ide's (Integrated Development Environment) com garantias de pleno funcionamento;
- A renderização dos gráficos pode sofrer atrasos devido à execução do código ou à capacidade de processamento da máquina;
- Múltiplas dependencias entre sistemas para funciomaneto,especialmente dos navegadores web;
- A impossibilidade atual de gerar um executável que facilitaria a usabilidade da ferramenta;
- Pode apresentar lentidão na execução dependendo do volume de dados.

5 GUIA DE USO

- REQUISITOS

  - A interface necessita que o usuário tenha instalado o Pycharm, o banco de dados MariaDB, o protocolo OPC-DA e o Python instalados no computador.

- ACESSO A FERRAMENTA:

  - Projeto_PRH_ANP_35.1

- EXTRAÇÃO DOS ARQUIVOS

  - A Figura 25 mostra os arquivosque devem ser extraídos para uma mesma pasta no computador do usuário.

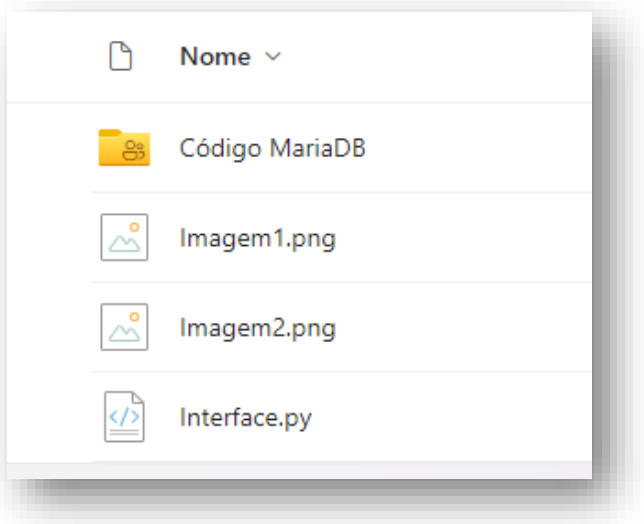

Figura 25: Página com os arquivos

  - Transfira os códigos na pasta “Código MariaDB” para o banco dados dentro do Heidisql, como se vêna Figura 26.

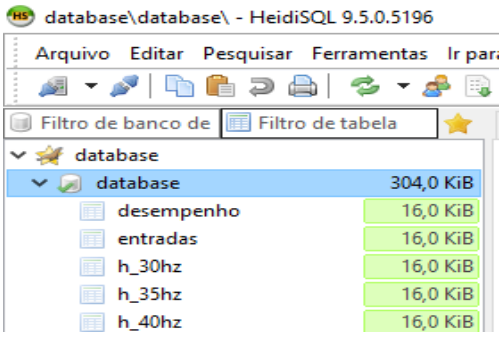

Figura 26: Bano de dados no Heidisql

- COMPILAÇÃO DO CÓDIGO DA INTERFACE

- Com os arquivos devidamente extraido e alocado na mesma pasta de preferência do usuário. Deve-se abrir o código “Interface.py” no Pycharm.
- Em seguida, com o pycharm aberto, instale (caso não os tenha) os pacotes essenciais para a interface através do comando `pip install "nome da biblioteca"`, no terminal do computador. As bibliotecas necessárias são mostradas na Listagem 16.

```
pip install dash
pip install pandas
pip install plotly.graph_objects
pip install base64
pip install dash_bootstrap_components
pip install mysql.connector
pip install dash_extendable_graph
pip install dash_daq
pip install plotly.express
```

Listagem 16: Bibliotecas para instalação.

- Após a instalação dos pacotes, realize a compilação do código clicando no botão “Run”, na área superior a direita da tela do Pycharm, como mostrado na Figura 27.

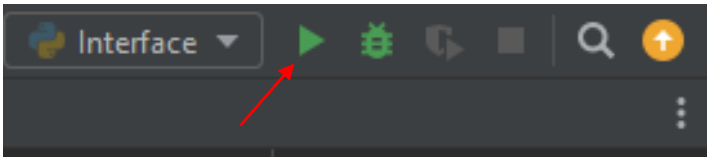

Figura 27: Compilação do código.

- Uma vez compilado, na parte inferior do Pycharm, será exibido o resultado da compilação em “Dash is runnin on”, como mostrado na Figura 28.

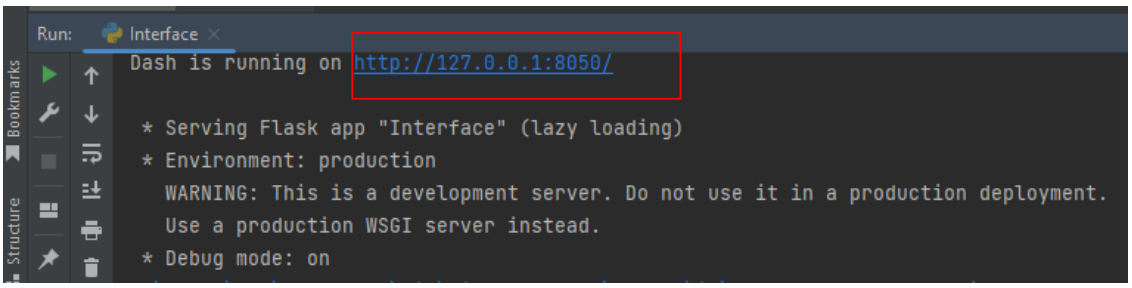

Figura 28: Resultado do código compilado

- Após clicar no link indicado na Figura 28, o usuário será encaminhado para o seu navegador padrão de internet, onde a interface será exibida.

➢ FUNCIONALIDADE DOS GRÁFICOS

- O usuário tem a possibilidade de escolher uma variável específica que ele deseja observar apenas clicando duas vezes no nome da variável na legenda do gráfico como mostrado na Figura 29.

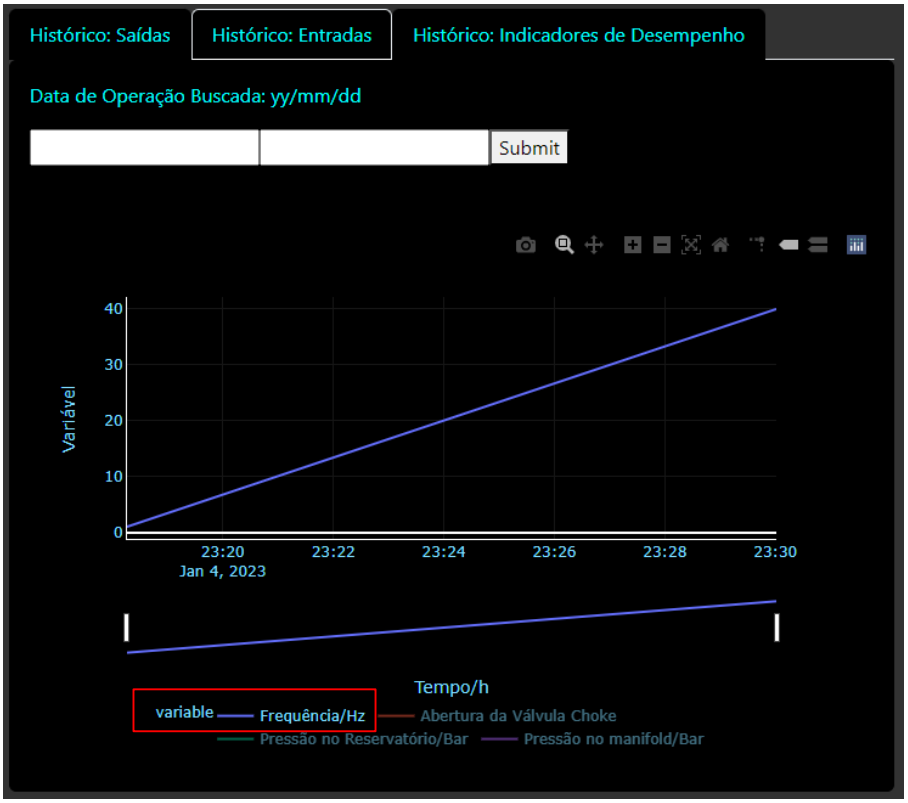

Figura 29: Exibição de uma única curva.

- É possível também visualizar algumas informações sobre as variáveis (valor, unidade de medida e nome), apontando o indicador do mause do computador para um ponto desejado no gráfico como ilustra a Figura 30.

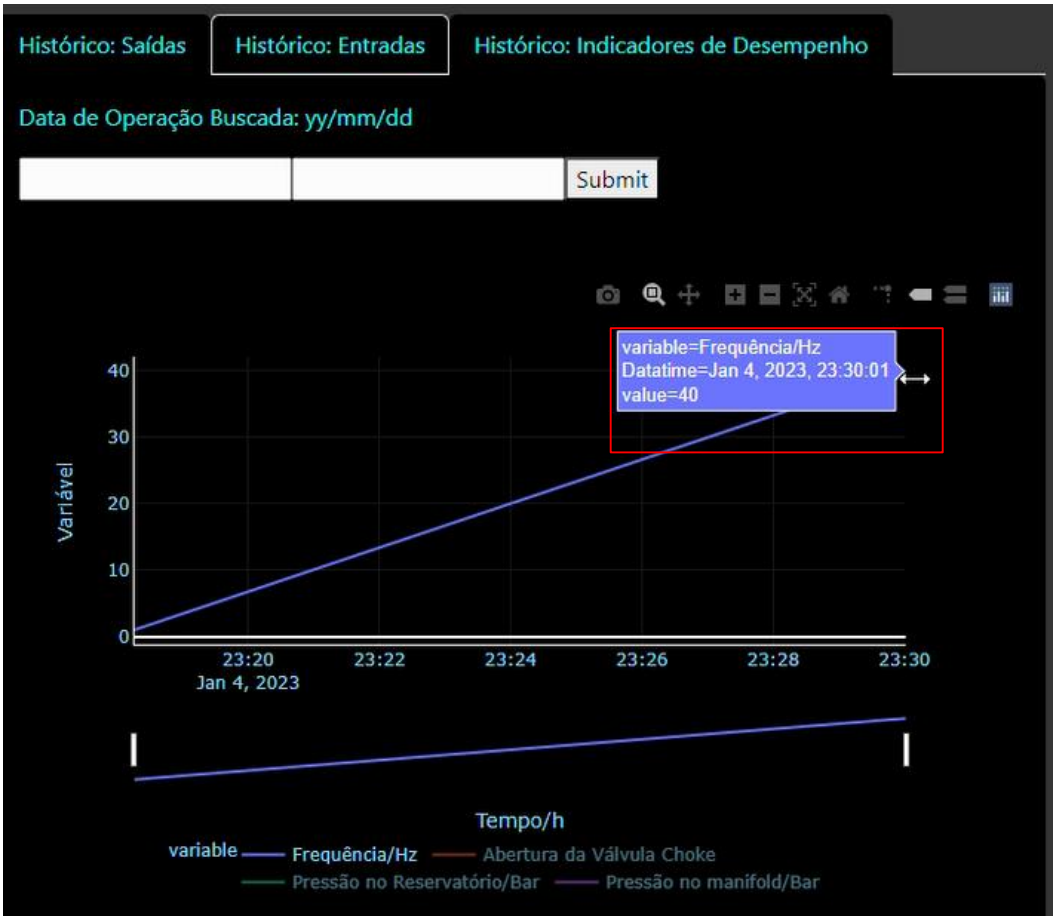

Figura 30: Informações das variáveis.

- Pode-se acessar o histórico de dados das entradas, saídas e indicadores de desempenho da BCS atrvés da alternância de suas respectivas abas e incluindo o período desejado de busca no modelo yy (ano), mm (mês), dd (dia), como pode ser visto na Figura 31.

Figura 31: Consulta ao histórico de dados.

- Outro aspecto muito importante no uso da interface são os recursos de visualização gráfica em destaque na Figura 32 onde, da esquerda para a direita, apresenta as seguintes funcionalidade: Download plot as an png, Zoom, Zoom in, Zoom out, Autoscale, Reset axes, Toggle spike lines, Show closest data on hover e Compare data on hover.

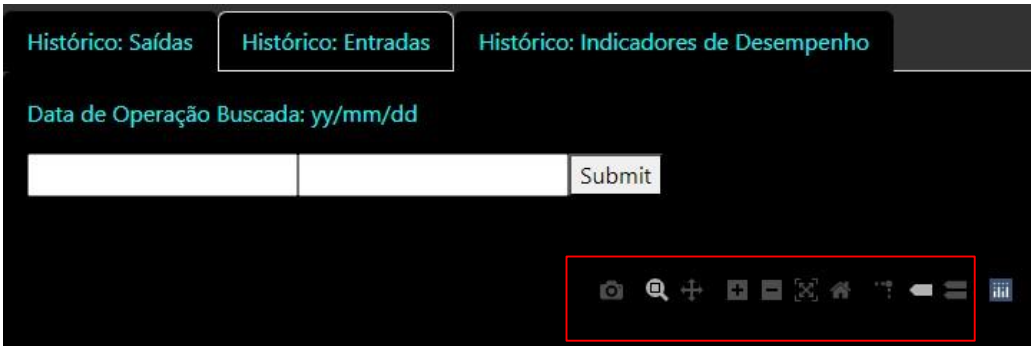

Figura 32: Recursos gráficos.

- ATIVAÇÃO DA BUSCA EM TEMPO REAL

  - Para a ativação e desativação do modo de busca automático de dados, basta apenas clicar no botão “live” na parte superior central da interface, com isso o cronômetro (Tempo/s) começará a fazer a contagem do tempo de funcionamento doprograma, como se vê na Figura 33.

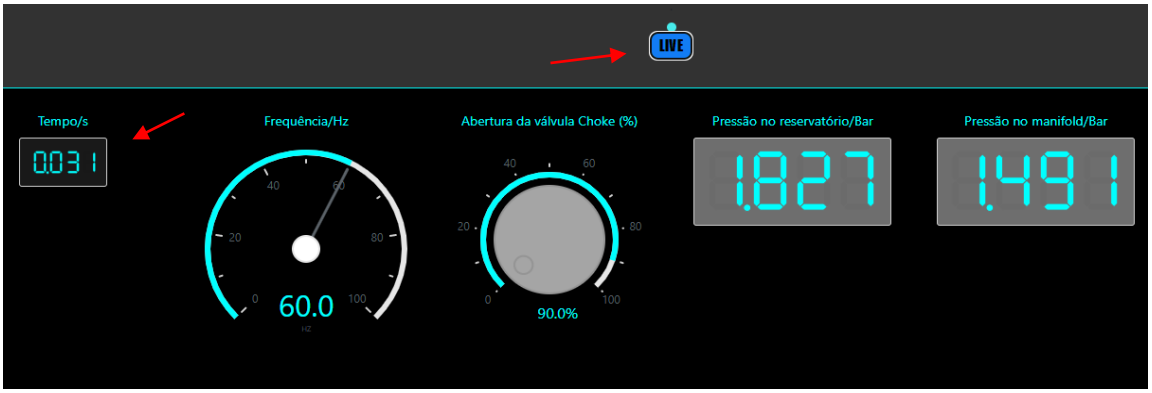

Figura 33: Ativação do modo de busca automático.

6 CONCLUSÃO

A interface final apresentada neste trabalho consegue abarcar parte significativa de todas as variáveis necessárias ao monitoramento da BCS-LEA. Ela também apresenta o funcionamento automático da consulta ao banco de dados para a utilização das informações de variáveis de entradas, saídas e indicadores de desempenho na interface, bem como a possibilidade de conexão entre o protocolo OPC-DA e o banco de dados através de um cliente OPC construído em Python, mostrando-se uma ferramenta de aplicabilidade dinâmica e funcional para as demandas de monitoramento do sistema BCS.

Em relação ao formato projetado da arquitetura do fluxo de dados, tem-se os sensores instalados na planta BCS-LEA que envia as informações medidas para os controladores lógicos programáveis (CLP's), que por sua vez enviam os dados para o OPC Server, este servidor será consultado pelo OPC Client (através da conexão python-OPC) que fará a obtenção das *tags* das variáveis e seus respectivos valores e as escreverão no banco de dados MariaDB, o banco é gerenciado pelo software HeidiSQL que facilita sua escrita e estruturação. O banco de dados será consultado pelo *software* do sistema supervisório automaticamente, e assim, a interface fará a exibição dos dados de forma gráfica e amigável ao usuário. Este processo foi validado via simulações e teste realizado em laboratório, destacando aqui a atualização em tempo real da interface ao consultar o banco de dados e exibir as informações na interface de forma automática, esta parte já está em funcionamento.

Em síntese, a interface consegue acessar o banco de dados que contem as informações operacionais, de desempenho (*head*, eficiência e potência) e o envelope de operação da BCS, tendo a possibilidade de apresentar os dados do processo em tempo real, acrescentando maiores possibilidades técnicas ao uso da planta BCS-LEA.

REFERÊNCIAS

BORGES, Alexandre Magno Coutinho. **Estudo prático comparativo para validação de curva de desempenho de bombas centrífugas submersas operando com fluidos viscosos**. 2018. 95 f. Dissertação (Mestrado) - Curso de Engenharia Química, Universidade Federal da Bahia, Salvador, 2018.

CARVALHO, Lívia Chaguri e. Bombeamento de Fluidos. Lorena: Departamento de Engenharia Química, 2019.

CASTELLANOS, Mauricio Barrios. Monitoramento de falhas operacionais em bombas centrífugas multiestágios usando Árvores de Decisão. 2019. 108 f. Dissertação (Mestrado) - Curso de Engenharia Mecânica, Universidade Estadual de Campinas, Campinas, 2019.

ARAðJO, Alana Raniele Nascimento. DETERMINAÇÃO DA CURVA EQUIVALENTE PARA UMA ASSOCIAÇÃO DE BOMBAS BCS EM SÉRIE. 2015. 62 f. TCC (Graduação) - Curso de Engenharia de Petróleo, Universidade Federal do Rio Grande do Norte, Natal, 2015.

Pavlov, A., Krishnamoorthy, D., Fjalestad, K., Aske, E., Fredriksen, M., 2014. Modelling and Model Predictive Control of oil wells with Electric Submersible Pumps, in: 2014 IEEE CONFERENCE ON CONTROL APPLICATIONS (CCA), IEEE International Conference on Control Applications. IEEE, 345 E 47TH ST, NEW YORK, NY 10017 USA, pp. 586–592.

LIMA, Antonio Cezar de Castro; COSTA, Erbet Almeida; SANTOS, Flávio Jesus dos; FONTANA, Márcio; ABREU, Odilon Santana Luiz de; SILVA., Tiago de Oliveira. APÊNDICE B – RELATÓRIO TÉCNICO 03 POÇO DE BCS INTELIGENTE. Salvador: Centro de Capacitação Tecnológica em Automação Industrial - Ufba, 2020.

BECKER, Ansgar. **What's this?** Disponível em: https://www.heidisql.com/. Acesso em: 30 nov. 2023. SOFTWARE TOOLBOX. TOP Server. Disponível em: https://software.com.br/p/top-server. Acesso em: 30 nov. 2023.

SILVA, Matheus Ferreira da. **Desenvolvimento de uma Interface de Comunicação entre Servidores OPC DA e Clientes OPC UA**. 2019. 51 f. TCC (Doutorado) - Curso de Engenharia Elétrica, Universidade Federal de Campina Grande, Campina Grande, 2019.

SOUZA, Deyse da Mata Oliveira. **Sistemas de medição de desempenho para projetos de PD&I no setor de petróleo e gás natural**. 2014. 296 f. Tese (Doutorado) - Curso de Ciência e Engenharia de Petróleo, Universidade Federal do Rio Grande do Norte, Natal, 2014.

FORMIGONI, Philipe de Araújo Fernandes. **PYTHON NA ANÁLISE DE DADOS: ESTUDO DE CASO COM DADOS DE ACIDENTES AÉREOS NO BRASIL**. 2021. 168 f. TCC (Graduação) - Curso de Engenharia de Produção, Universidade Federal Fluminense, Niterói, 2021.

SOARES, Lennedy Campos. **Sistema supervisório para Poços de Petróleo Baseados no Método de Elevação Artificial Plunger Lift**. 2011. 65 f. Dissertação (Mestrado) - Curso de Ciência e Engenharia de Petróleo, Universidade Federal do Rio Grande do Norte, Natal - Rn, 2011.

SOUZA, Rodrigo Barbosa de. Uma Arquitetura para Sistema Supervisórios Industrias e suas Aplicações em Processos de Elevação Artificial de Petróleo. 2005. 71 f. Dissertação (Mestrado) - Curso

de Engenharia Elétrica, Universidade Federal do Rio Grande do Norte, Natal, 2005.

McKinney, W., & others. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in ScienceConference (Vol. 445, pp. 51–56).

SOUZA, Douglas. **Python para Análise de Dados**. Disponível em: https://www.dio.me/articles/python-para-analise-de-dados. Acesso em: 24 out. 2023.

VASCONCELLOS, Paulo. **Como criar gráficos interativos utilizando Plotly e Python**. Disponível em: https://paulovasconcellos.com.br/como-criar-gr%C3%A1ficos-interativos-utilizando-plotly-e-python-3eb6eda57a2b. Acesso em: 15 mar. 2021.

Sievert C (2020). *Interactive Web-Based Data Visualization with R, plotly, and shiny*. Chapman and Hall/CRC. ISBN 9781138331457, https://plotly-r.com.

QUISPE, Newton Roy Pampa. **Técnicas e ferramentas para a extração inteligente e automática de conhecimento em banco de dados**. 2003. 104 f. Dissertação (Doutorado) - Curso de Engenharia Elétrica, Universidade Estadual de Campinas, Campinas, 2003.

JOHNSON, Alex. **Dash-daq 0.5.0**. Disponível em: https://pypi.org/project/dash-daq/. Acesso em: 01 dez. 2023.

SOUZA, Ivan de. **Banco de dados: saiba o que é, os tipos e a importância para o site da sua empresa**. Disponível em: https://rockcontent.com/br/blog/banco-de-dados/. Acesso em: 25 fev. 2020.

FRANCA, Gilberto Horacio. AUTOMAÇÃO DE POÇOS DE PETRÓLEO QUE UTILIZAM O BOMBEIO CENTRÍFUGO SUBMERSO (BCS) COMO MÉTODO DE ELEVAÇÃO. In: XXXI ENCONTRO NACIONAL DE ENGENHARIA DE PRODUCAO, 31., 2011, Belo Horizonte. **Inovação Tecnológica e Propriedade Intelectual: Desafios da Engenharia de Produção na Consolidação do Brasil no Cenário Econômico Mundial.** Belo Horizonte: Nda, 2011. p. 1-11.

SILVA, Eduardo. **10 Melhores IDEs e Editores de Código em Python para 2022**. Disponível em: https://blog.geekhunter.com.br/ides-e-editores-de-codigo-em-python-para-2021/. Acesso em: 30 nov. 2023.

SOUZA, Ivan de. **MariaDB ou MySQL: saiba qual tecnologia de banco de dados escolher**. Disponível em: https://rockcontent.com/br/blog/mariadb/. Acesso em: 30 nov. 2020.

NOKERI, Tshepo Chris. **Dash Bootstrap Components**. Disponível em: https://link.springer.com/chapter/10.1007/978-1-4842-7783-6_6. Acesso em: 30 nov. 2023.
BAWEJA, Chaitanya. **Python and MySQL Database: A Practical Introduction**. Disponível em: https://realpython.com/python-mysql/. Acesso em: 30 nov. 2023.

PARMER, Chris. **Plotly 5.18.0**. Disponível em: https://pypi.org/project/plotly/. Acesso em: 01 dez. 2023.

SOARES, Lennedy Campos. **Sistema supervisório para Poços de Petróleo Baseados no Método de Elevação Artificial Plunger Lift**. 2010. 65 f. Dissertação (Mestrado) - Curso de Ciências e Engenharia de Petróleo, Universidade Federal do Rio Grande do Norte, Natal - Rn, 2010.

KRUCHTEN, Nicolas. **Plotly-express 0.4.1**. Disponível em: https://pypi.org/project/plotly-express/. Acesso em: 01 dez. 2023.

LIANG, Brad. **Dash-extendable-graph 1.3.0**. Disponível em: https://pypi.org/project/dash-extendable-graph/. Acesso em: 01 dez. 2023.

KWIATKOWSKI, Michal. **OpenOPC-Python3x 1.3.1**. Disponível em: https://pypi.org/project/OpenOPC-Python3x/. Acesso em: 1 dez. 2023.

SANTOS, Vanessa Maria dos. MODELAGEM DE UM SISTEMA SUPERVISÓRIO DA CABEÇA DE PRODUÇÃO DE UM POÇO COM BCP UTILIZANDO O INDUSOFT. 2019. 54 f. Monografia (Especialização) - Curso de Engenharia de Petróleo, Centro Universitário Tiradentes, Maceió, 2019.

ALVES, Gustavo Furtado de Oliveira. O que é um SGBD? Disponível em: https://dicasdeprogramacao.com.br/o-que-e-um-sgbd/. Acesso em: 17 dez. 2023.

HUGHES, Adam. Open Database Connectivity (ODBC). Disponível em: https://www.techtarget.com/searchoracle/definition/Open-Database-Connectivity. Acesso em: 17 dez. 2023.