INSTITUTO FEDERAL DE EDUCAÇÃO, CIÊNCIA E TECNOLOGIA

DO TRIÂNGULO MINEIRO – *CAMPUS* PARACATU

BACHARELADO EM ENGENHARIA ELÉTRICA

ITALO BRITO GONCALVES

Integração WhatsApp com banco dados MySQL com objetivo de ler dados de inversores fotovoltaicos

PARACATU - MG

2023

ITALO BRITO GONCALVES

Integração WhatsApp com banco dados MySQL com objetivo de ler dados de inversores fotovoltaicos

Trabalho de conclusão de curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro, Campus Paracatu, como exigência parcial para obtenção do diploma de Bacharel em Engenharia Elétrica, sob a orientação do Prof. Msc. Prof. Silas Martins Sousa.

PARACATU - MG

2023

Ficha Catalográfica elaborada pelo Setor de Referência do IFTM –

Campus Paracatu

G635i Italo brito Gonçalves-
Integração WhatsApp com banco dados MySQL com objetivo de ler dados de inversores fotovoltaicos./ Italo brito Gonçalves - 2023.
43f. : il.

Orientador: Silas Martins Sousa.
Trabalho de conclusão de curso (graduação) - Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro, Curso de bacharelado em Engenharia Elétrica, Paracatu, 2023.

1. Energia fotovoltaica. 2. Inversores fotovoltaicos. 3. Banco de dados. 4. Whatsapp. 5. Chatbot. I. Silas Martins Sousa. II. Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro - Campus Paracatu. III. Título.

CDD 621.31

Bibliotecária: Nathália de Morais Torres CRB6-3097

TERMO DE APROVAÇÃO

ITALO BRITO GONCALVES

Integração WhatsApp com banco dados MySQL com objetivo de ler dados de inversores fotovoltaicos.

Trabalho de conclusão de curso apresentado ao Instituto Federal de Educação, Ciência e Tecnologia do Triângulo Mineiro, Campus Paracatu, como exigência parcial para obtenção do diploma de Bacharel em Engenharia Elétrica, sob a orientação do Prof. Msc. Prof. Silas Martins Sousa.

Aprovado em 21 de junho de 2023.

Prof. Msc. Silas Martins Sousa.

Prof. Msc. Rafael Mendes Faria.

Prof. Dr. Claiton Luiz Soares.

Denis Alves Nobre (Membro Externo).

PARACATU - MG

2023

DEDICATÓRIA

Dedico este trabalho aos meus pais, amigos e colegas que estiveram sempre ao meu lado durante toda a minha jornada acadêmica. Seus apoios constantes e encorajamento foram fundamentais para meu crescimento e sucesso na graduação. Agradeço imensamente a todos por fazerem parte dessa conquista e por compartilharem comigo momentos inesquecíveis ao longo dessa jornada.

AGRADECIMENTOS

Gostaria de expressar minha gratidão primeiramente a Deus, que me concedeu força e sabedoria para concluir minha graduação. Além disso, sou grato aos meus pais, que generosamente forneceram os recursos necessários para que eu pudesse prosseguir com meus estudos.

EPÍGRAFE

Porque para Deus nada é impossível. (Lucas 1:37).

RESUMO

A criação de uma aplicação para o mercado atual de energia fotovoltaica é uma iniciativa que visa aproveitar os benefícios dessa fonte de energia limpa e sustentável (REZENDE, 2019). A aplicação em questão consiste em uma integração do WhatsApp com um banco de dados MySQL, por meio da linguagem JavaScript, a fim de fornecer informações valiosas para um número conectado. Essa aplicação tem a capacidade de ler os dados armazenados no banco de dados, possibilitando a entrega dessas informações ao número solicitante. Para alimentar os dados no banco, um código em Python é utilizado para buscar informações nos sites dos inversores e realizar o upload automático no MySQL, eliminando a necessidade de intervenção humana. Essa funcionalidade da aplicação é especialmente útil, uma vez que permite a integração de diversos inversores de marcas diferentes em um único canal de comunicação. Dessa forma, os usuários podem acessar facilmente as informações relevantes dos seus inversores, independentemente da marca que utilizam. Isso simplifica o monitoramento e controle dos sistemas fotovoltaicos, contribuindo para o desenvolvimento sustentável no setor de energia.

Palavras chaves: Inversores. Energia. Whatsapp. Tecnologia. Chatbot.

ABSTRACT

The creation of an application for the current photovoltaic energy market is an initiative aimed at harnessing the benefits of this clean and sustainable energy source (REZENDE, 2019). The application in question involves integrating WhatsApp with a MySQL database using the JavaScript language, in order to provide valuable information to a connected number. This application has the ability to read the data stored in the database, enabling the delivery of this information to the requesting number. To populate the database with data, a Python code is used to fetch information from inverter websites and automatically upload it to MySQL, eliminating the need for human intervention. This functionality of the application is particularly useful as it allows the integration of various inverters from different brands into a single communication channel. In this way, users can easily access relevant information about their inverters, regardless of the brand they use. This simplifies the monitoring and control of photovoltaic systems, contributing to sustainable development in the energy sector.

Keywords: Inverters. Energy. WhatsApp. Technology. chatbot.

LISTA DE FIGURAS

LISTA DE QUADROS

LISTA DE ABREVIATURAS E SIGLA

API *Application Programming Interface*

VPS *Virtual Private Server*

SQL *Structured Query Language*

QRCODE *Quick Response Code*

SUMÁRIO

1 INTRODUÇÃO

A evolução da tecnologia tem sido especialmente notável nas últimas décadas, com a criação de novas ferramentas que revolucionaram a maneira como as pessoas se comunicam e trabalham (FERNANDES, 2016).

Com a popularização do WhatsApp, uma das ferramentas de comunicação mais utilizadas em todo o mundo, surgiu a oportunidade de explorar novas formas de utilização dessa plataforma. Uma possibilidade, é utilizar o WhatsApp como uma ferramenta para fornecer dados de inversores fotovoltaicos.

O sistema fotovoltaico é uma descoberta revolucionária, permitindo que a engenharia e a tecnologia produzam energia de forma sustentável e simples. Por meio das placas solares, é possível captar energia em corrente contínua (CC) e convertê-la em corrente alternada por meio do inversor (BRAGA, 2008). Dessa forma, toda essa energia limpa pode ser utilizada diretamente na rede elétrica ou armazenada em baterias para uso posterior. Além disso, os sistemas fotovoltaicos representam uma forma limpa e sustentável de geração de energia, pois não produzem emissões de gases de efeito estufa durante o processo (REZENDE, 2019).

Devido aos avanços tecnológicos e aos esforços contínuos para melhorar a eficiência dos equipamentos de captação de energia fotovoltaica e aprimorar os inversores, temos observado um aumento significativo na utilização e instalação de novos pontos de coleta (OLIVEIRA, 2017). Sabendo que esse crescimento é impulsionado pelo reconhecimento dos benefícios ambientais e econômicos proporcionados pela utilização desses sistemas, dedicou-se esforços para realizar o desenvolvimento contínuo dessa tecnologia de automação, e assim, torná-la mais suscetível ao aumento expressivo na capacidade de gerar resultados positivos para os usuários.

1.1 TEMA

O desenvolvimento da API do WhatsApp estabelece conexão com um banco de dados MySQL para automatizar a leitura dos dados armazenados pelos inversores de energia fotovoltaica. Esses dados são inseridos no banco por meio de um programa em Python. Como resultado, o mercado de energia sustentável pode experimentar um crescimento significativo, pois essa ferramenta facilita a obtenção de informações do inversor, que são de difícil acesso quando se pesquisa no site oficial da marca do inversor. Portanto, a principal motivação do projeto é impulsionar o mercado de energia fotovoltaica ao oferecer uma solução que favorece os usuários ao disponibilizar os dados dos inversores.

1.2 PROBLEMA

Os inversores fotovoltaicos são dispositivos usados em sistemas de energia solar para converter a energia gerada pelos painéis solares em energia elétrica que pode ser usada em residências e empresas (DO NASCIMENTO, 2004). A obtenção de dados desses dispositivos é essencial para garantir que o sistema de energia solar esteja funcionando corretamente e para identificar problemas em caso de falhas.

Infelizmente, muitos usuários negligenciam a verificação regular desses dados, o que pode resultar em prejuízos financeiros significativos. Além disso, os usuários podem enfrentar dificuldades para pesquisar e *logar* em sites específicos das marcas de inversores, e como consequência disso, podem demorar dias ou até meses para que percebam as falhas no sistema. Esse cenário se torna mais ainda desafiador considerando a falta de tempo e a quantidade de tarefas diárias que eles já possuem em sua rotina.

Portanto, é essencial buscar soluções mais simples e práticas para a verificação dos dados dos inversores fotovoltaicos. Ao facilitar o acesso e a compreensão dessas informações, podemos analisar melhor as falhas e os rendimentos do sistema, permitindo a tomada de decisões mais assertivas. Uma abordagem mais amigável e acessível para monitorar os dados dos inversores pode ajudar a identificar problemas mais rapidamente, reduzir o tempo de inatividade do sistema e otimizar o desempenho geral da instalação de energia solar.

1.3 OBJETIVOS

1.3.1 Objetivo geral

Desenvolver uma aplicação que estabeleça conexão com o aplicativo oficial do WhatsApp e esteja integrada a um banco de dados MySQL.

1.3.2 Objetivos específicos

- Verificar se a aplicação conectada ao WhatsApp torna possível a leitura dos dados do inversor.

- Desenvolver um código em Python capaz de baixar arquivo dentro do site do inversor.

- Elaborar uma aplicação capaz de lê arquivo Excel e excluir de forma automática.

1.4 HIPÓTESE

H1- A aplicação de API (*Application Programming Interface)* facilitará a consulta de informações do dado gerado pela energia solar aos usuários.

H2- Os usuários negligenciam acesso ao site web dos inversores.

H3- A simulação de uma conversa no WhatsApp através de codificação cria um relacionamento positivo entre o usuário e a API.

1.5 JUSTIFICATIVA

Com o mercado inovando e a tecnologia avançando, surgiu a ideia de desenvolver uma ferramenta capaz de alinhar melhor os dados da produção de energia fotovoltaica. Essa iniciativa representa uma possibilidade para impulsionar o mercado, uma vez que a ferramenta traz alguns benefícios muito importantes, como a otimização do tempo dos clientes e a facilidade de acesso.

O sistema também permite a integração de diferentes marcas de inversores, o que torna o processo de monitoramento muito mais eficiente. Com a união de dados em um único banco de dados, o usuário pode comparar informações e otimizar o uso de seus inversores para obter a melhor produção possível.

Além disso, o sistema permite a análise dos dados de produção ao longo do tempo, o que é útil para identificar tendências e fazer projeções futuras. Com essas informações em mãos, os usuários podem tomar decisões, embasadas nos dados, a respeito do uso e manutenção de seus sistemas de energia solar. Em resumo, a utilização da API para acessar os dados dos inversores é uma solução eficiente e prática que pode ajudar os usuários a monitorar e otimizar a produção de energia solar. Com essa ferramenta, o processo de coleta de dados é simplificado e os usuários podem obter informações relevantes de forma rápida e fácil.

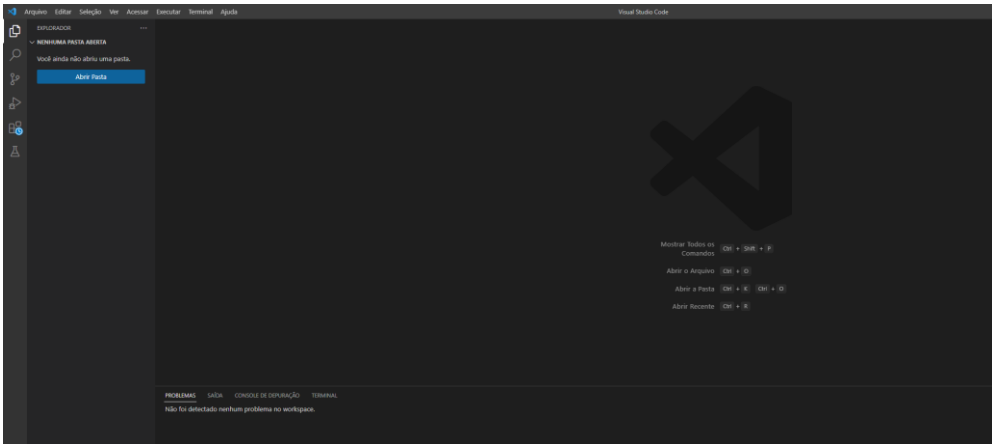
2 REFERENCIAL TEÓRICO

Para o desenvolvimento de uma API, é necessário utilizar algumas ferramentas que facilitam o trabalho e melhoram a funcionalidade da aplicação. A seguir, listamos alguns programas e emuladores essenciais:

2.1 IDE (*Integrated Development Environment*)

É um ambiente de desenvolvimento integrado que facilita a escrita de código. Algumas IDEs populares para desenvolvimento de API incluem o Visual Studio Code mostrado na Figura 1 , Eclipse e NetBeans.

Figura 1 -Vista programa Visual Studio code aberto



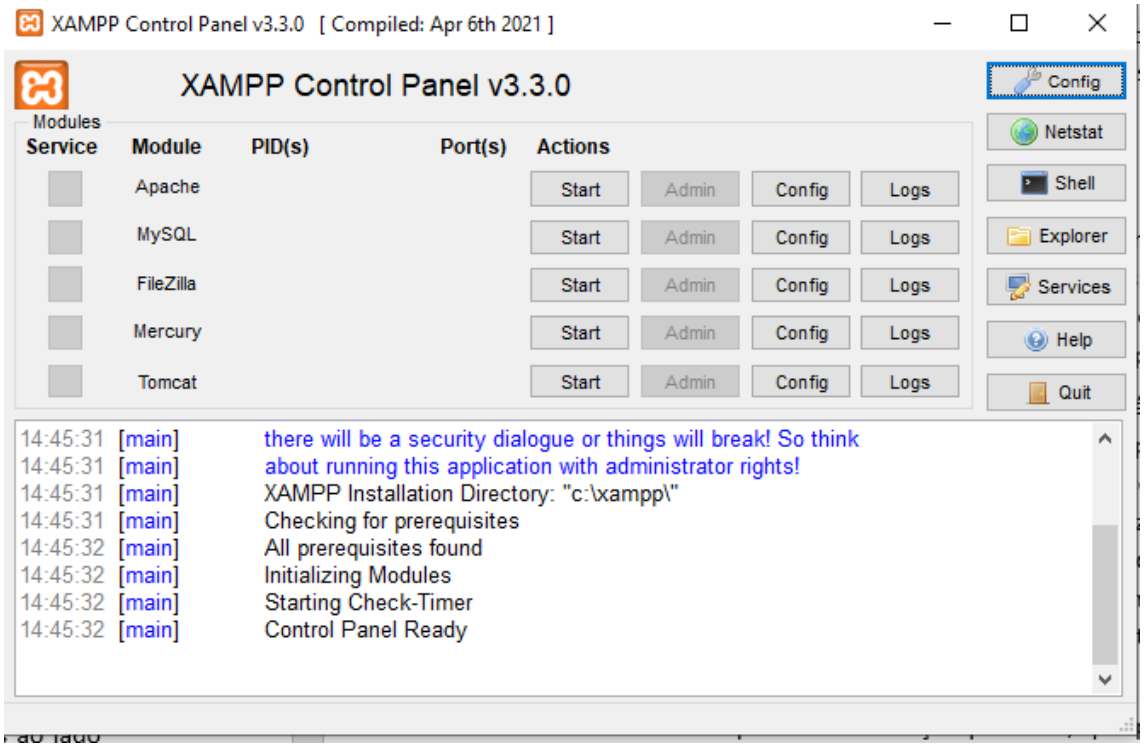
Fonte: MICROSOFT, 2023.

Para conectar uma API com um banco de dados MySQL, é necessário utilizar um servidor como o XAMPP, que permite criar e testar aplicativos web localmente no computador, sem precisar de uma conexão com a internet ou de um servidor web externo.

O XAMPP é uma escolha popular entre desenvolvedores, já que oferece uma maneira fácil e rápida de configurar um ambiente de desenvolvimento completo em um computador local sem precisar configurar manualmente cada componente. Ele é compatível com os principais sistemas no mercado, incluindo Windows, Linux e MacOS, que auxiliam no processo de tornar o desenvolvimento mais fácil, eficiente e colaborativo. No entanto, ressalta-se que o XAMPP não deve ser utilizado com o objetivo de hospedar aplicativos web em produção, tendo

em vista que para isso seria necessário um servidor web dedicado e configurado conforme determinados termos de segurança e desempenho. Na figura 2, observa-se o XAMPP, servidor que possui as funções Apache e MySQL que iniciam o banco de dados localmente.

Figura 2– Tela inicial do XAMPP



Fonte: próprio autor (2023).

2.2 Desenvolvimento com Node.js

O Node.js é uma plataforma que oferece muitos recursos e bibliotecas para ajudar no desenvolvimento de APIs. Por exemplo, o Express é um dos frameworks mais populares para criar APIs em Node.js, fornecendo uma estrutura flexível e escalável para desenvolver rotas, controladores e *middleware* (ferramenta responsável pela comunicação e integração entre esses sistemas). Somado a isso, o Express.js oferece uma camada de abstração que simplifica a criação de servidores HTTP de forma instantânea.

De acordo com a documentação oficial do Node.js (Node.js Foundation, 2023), a plataforma é descrita como "um ambiente de tempo de execução construído com base no motor

JavaScript V8 do Chrome". O Node.js permite que os desenvolvedores usem JavaScript tanto no '*front-end '* (aplicação com a qual os usuários interagem) quanto no '*back-end '* (não visível aos usuários), o que torna a plataforma uma opção atraente para desenvolvimento web.

O Express, por sua vez, é um framework construído em cima do Node.js e pode ser instalado através do gerenciador de pacotes npm (*Node Package Manager*) que é amplamente utilizado na comunidade de desenvolvimento de software baseado em JavaScript pois permite que os programadores instalem, compartilhem, atualizem e gerenciem as dependências de um projeto de forma eficiente.

Quando se utiliza o npm, se torna possível acessar um vasto repositório de pacotes de código aberto, que são módulos de software prontos para uso e desenvolvidos por terceiros. Esses pacotes fornecem funcionalidades adicionais, bibliotecas, frameworks e ferramentas que podem ser incorporados aos projetos para agilizar o processor. Logo, o npm oferece uma série de recursos para a criação de APIs, como roteamento, gerenciamento de *middleware* e controle de erros

Outro recurso importante do Node.js é a capacidade de usar o JavaScritp assíncrono, pois ele permite que as tarefas sejam executadas sem bloquear a thread principal do navegador ou do ambiente de execução. Isso, é extremamente útil para lidar com solicitações de entrada e saída (I/O) e outros processos que podem ser lentos ou não bloqueantes como leitura e gravação de arquivos, acesso a bancos de dados e chamadas de rede. Nas aplicações web tradicionais, essas operações de I/O podem ser bloqueantes, o que significa que o processo fica parado, aguardando a conclusão de uma determinada tarefa antes de prosseguir para a próxima. Neste contexto, o Node,js pode ajudar a melhorar a eficiência e a capacidade de resposta da API, permitindo que ela suporte mais solicitações ao mesmo tempo.

Além disso, o Node.js é um sistema multiplataforma que pode ser executada em vários sistemas operacionais, incluindo Windows, Linux e MacOS. Isso significa que o usuário pode desenvolver em qualquer sistema operacional que preferir e implantá-la em um servidor que suporte Node.js. Existem muitos provedores de hospedagem na nuvem que suportam Node.js, como a Amazon Web Services (AWS), Microsoft Azure, Google Cloud Platform, Heroku, DigitalOcean. Todos esses provedores fornecem recursos para implantar e gerenciar aplicativos Node.js na nuvem. Os arquivos desta pesquisa foram hospedados uma VPS (*Virtual Private Server*).

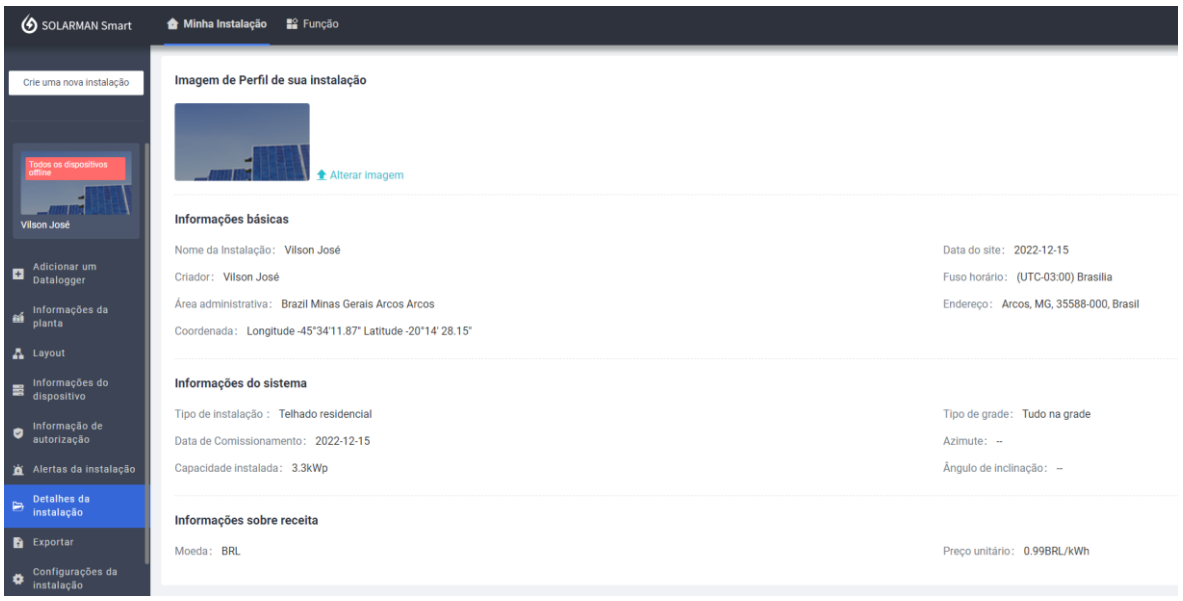
Por fim, o Node.js é uma plataforma de código aberto com uma comunidade de

desenvolvedores ativa e crescente. Isso significa que pode-se encontrar muitos recursos, documentação e suporte online para ajudar no desenvolvimento do projeto.

2.3 Objeto de estudo para desenvolver o projeto

Para fins de estudo, foi utilizado o site Solarman Smart como objeto de análise. A partir dos dados de produção dessa instalação, foram coletadas as informações necessárias. Além disso, todo o código e programa foram testados por meio do próprio site, realizando pesquisas baseadas nesse sistema, mostrado na Figura 3.

Figura 3- Imagem do painel usado para estudos



Fonte: SOLARMAN SMART, 2023

O projeto foi fundamentado em uma instalação real, utilizando dados reais. Para esse propósito, um inversor da marca BelEnergy foi empregado a fim de realizar testes realistas e executar o projeto. O adaptador Wi-Fi foi devidamente instalado para permitir que o sistema do inversor envie os dados para o banco de dados central da marca. O que viabiliza o acesso às informações por meio do site disponibilizado pela empresa fabricante do inversor, mostrado na Figura 4.

Figura 4- Imagem do inversor

Fonte: BelEnergy (2023)

3 **Materiais e Métodos**

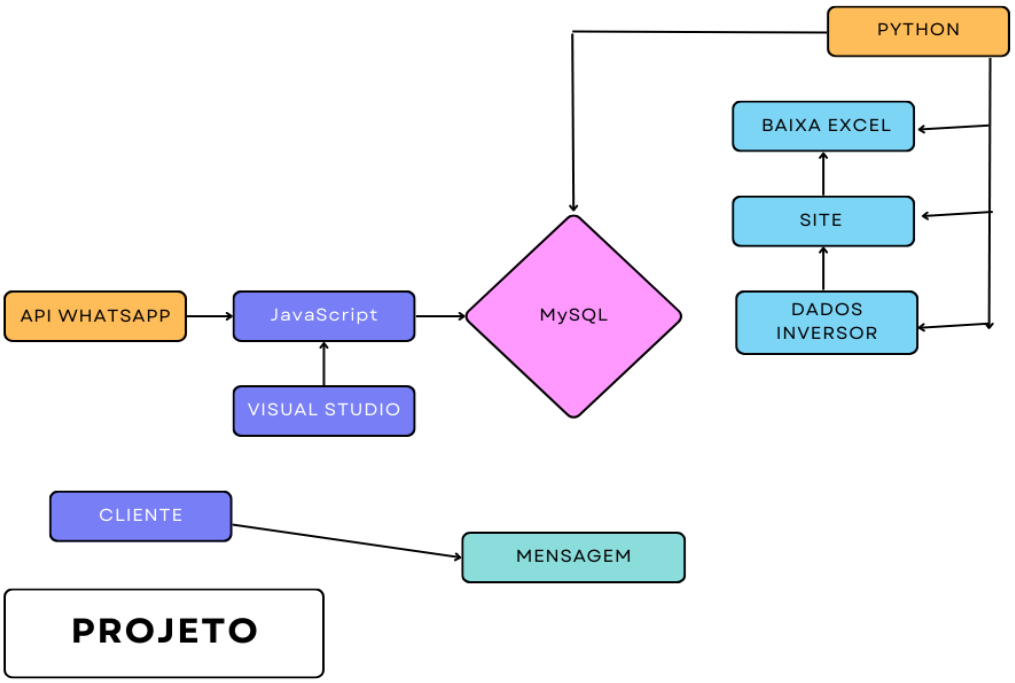
A metodologia utilizada para iniciar o projeto consistiu no uso do programa XAMPP, responsável por criar um banco de dados MySQL localmente, juntamente com o Visual Studio Code, onde os códigos foram escritos para execução.

A integração da API (*Application Programming Interface*) com o WhatsApp foi implementada em JavaScript, utilizando a biblioteca whatsappweb.js, que foi desenvolvida especificamente para realizar a comunicação entre essas duas plataformas mencionadas. Após a conexão, é necessário estabelecer a ligação entre o programa e o banco de dados, o que é possível através da função *'mysql.createConnection*()', responsável por estabelecer uma conexão com o MySQL em JavaScript. Com todo o processo configurado, foi desenvolvido um código para buscar informações no banco de dados, utilizando bibliotecas específicas. Além disso, a interface do usuário *'front-end'* foi projetada para ser utilizada de duas maneiras: uma lista de opções para seleção e uma opção com números correspondentes.

Além disso, após a conexão e o pleno funcionamento da API, é necessário fornecer os dados corretos ao banco de dados. Para isso, foi criada uma automação em Python capaz de acessar o sistema dos inversores, simulando a ação de um humano. Essa automação realiza o download das informações disponíveis no próprio site através de um arquivo Excel e, em seguida, estabelece uma conexão com o MySQL para carregar o arquivo no banco de dados. Após esse processo, uma função é executada para excluir o arquivo, possibilitando o download com o mesmo nome no dia seguinte. E todo esse procedimento ocorre de forma automatizada e não é visível para o usuário do sistema.

Em resumo, a metodologia utilizada para a execução do projeto envolveu pesquisa, busca de códigos e bibliotecas, conexão de sistemas e utilização de tecnologias avançadas para aprimorar a interação com o usuário e o monitoramento de equipamentos. O sistema resultante é um exemplo de como a tecnologia pode ser utilizada para aprimorar o desempenho de equipamentos e simplificar a interação do usuário com as ferramentas de monitoramento, o diagrama do projeto é mostrado na Figura 5.

Figura 5- Diagrama do projeto



Fonte: próprio autor (2023).

3.1 Recursos

Para fazer o sistema funcionar perfeitamente, é necessário deixar o programa criado em execução 24 horas por dia. Para isso, é preciso de um servidor capaz de suportá-lo e mantê-lo. Os dados podem ser armazenados neste servidor para que possam ser acessados a partir de qualquer local com acesso à internet, valores do orçamento mostrado no Tabela 1.

Descrição	Unidade	Quantidade	Valor Unitário	Valor Total
VPS (*virtual private server*)	01 MENSAL	01	R$ 140	R$ 140
TOTAL				**R$ 140**

Tabela 1- Recurso necessário rodar o projeto fora do servidor local. Fonte: próprio pelo autor.

3.2 Comando para instalar a conexão ‘QR code’ do WhatsApp oficial

Para a instalação da API 'QR code’, foi utilizada uma biblioteca pertencente ao site WWEBIS(2023), obtendo o módulo npm através do comando: ‘npm i whatsapp-web.js’ NPM (*Node Package Manager*) é o gerenciador de pacotes padrão do Node.js, usados para instalar e gerenciar bibliotecas e módulos de terceiros que podem ser usados em um projeto Node.js. Com o NPM, é possível instalar facilmente as bibliotecas necessárias para um projeto com um simples comando em um terminal como na Figura 5

Este módulo JS possibilita a conexão do WhatsApp Web com a API, utilizando-o para iniciar as funções de conexão com o MySQL e fazer a requisição de mensagens e alimentar o '*chatbot* 'conforme o usuário final desejar.

'Chatbot ' é um programa de computador projetado para interagir com seres humanos por meio de conversas em linguagem natural.

Figura 6– Tela do terminal aberta para Instalação do NPM

Fonte: próprio autor (2023).

3.3 Iniciando a API de conexão ‘QR code’

Ao iniciar a API, a biblioteca executa e gera o ‘QR code’, permitindo a conexão do número central para interação do usuário com os dados no MySQL. Essa conexão é estabelecida

de maneira semelhante à conexão com o WhatsApp Web. Na Figura 7, é apresentado o processo em que o código foi escrito para a biblioteca do ‘whatsApp-web.js’, garantindo sua execução correta.

Figura 7- Parte do código de conexão QR code

```
const { Client, LocalAuth, MessageMedia, List, Location } = require('whatsapp-web.js');
const express = require('express');
const { body, validationResult } = require('express-validator');
const qrcode = require('qrcode');
const port = process.env.PORT || 8000;
const app = express();
const server = http.createServer(app);
const io = socketIO(server);
```

Fonte: próprio autor (2023).

Lembrando que é possível utilizar apenas localmente. É importante ressaltar que essa conexão é estabelecida através do uso de bibliotecas específicas, como a ‘whatsapp-web.js’, ‘express-validator’, ‘socket.io’, ‘qrcode’, entre outras.

3.4 Abertura da página para conectar WhatsApp central

Para estabelecer uma conexão entre uma API e o WhatsApp usando o ‘QR Code’, é necessário abrir um navegador no localhost, na porta 8000 como mostrado na Figura 8.

Figura 8- “Qr code” para leitura do WhatsApp central

Fonte: próprio autor (2023).

O ‘QR code’ é uma forma de comunicação visual que permite que os dispositivos se conectem facilmente sem a necessidade de digitar informações complexas. Quando você escaneia o código ‘QR’, ele contém informações sobre a conexão, como um *token* de acesso, permitindo que o dispositivo se conecte ao WhatsApp Web.

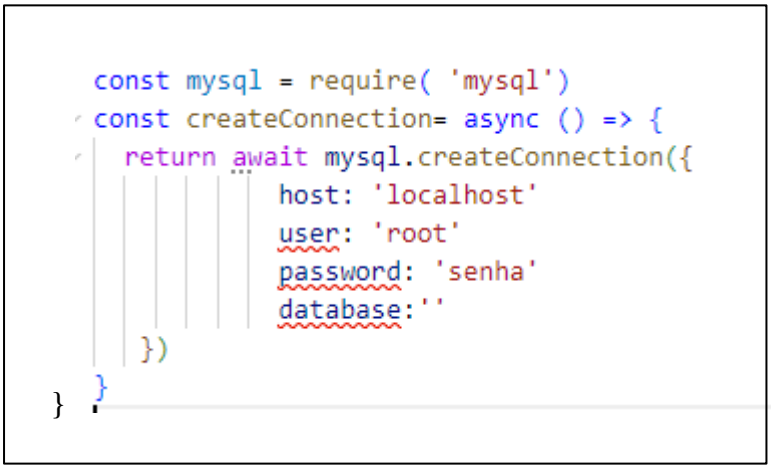
3.5 Comunicação do '*chatbot* ' com banco de dados.

Para estabelecer a comunicação com um banco de dados, utiliza-se uma biblioteca que viabiliza a interação entre JavaScript e MySQL. Com ela, garante-se uma conexão segura, possibilitando consultas aos dados e obtenção das informações solicitadas pelos usuários.

No desenvolvimento desse trabalho, explorou-se diversas funções, entre elas a '*createConnection*', que permitiu obter uma conexão com o banco de dados MySQL. Essa

função utiliza o método *'mysql.createConnection()*' para criar uma nova conexão, fornecendo os detalhes de hospedagem, nome de usuário, senha e o nome do banco de dados mostrado na Figura 9.

Figura 9- Código para conexão do '*bot*' com banco de dados



```
const mysql = require( 'mysql')
const createConnection= async () => {
  return await mysql.createConnection({
            host: 'localhost'
            user: 'root'
            password: 'senha'
            database:''
  })
}
}
```

Fonte: próprio autor (2023).

3.6 Código para buscar no banco de dados as repostas

Neste trecho de código, há uma funcionalidade crucial que envolve a conexão com um banco de dados para acessar informações sobre os inversores. Essa conexão é estabelecida utilizando a função '*createConnection*()', que permite estabelecer uma ligação confiável e segura com o banco de dados MySQL.

A conexão bem-sucedida com o banco de dados é um aspecto fundamental para o funcionamento adequado do projeto. Garantir a integridade e a disponibilidade dos dados é essencial para obter resultados precisos e confiáveis.

O código foi cuidadosamente desenvolvido para buscar as respostas necessárias dos clientes por meio de consultas ao banco de dados. Ao executar a consulta específica na tabela de inversores, obtém-se informações valiosas sobre os mesmos, como títulos e descrições.

Além disso, é importante destacar que foram implementados atrasos estratégicos por meio da função '*delay*()' para aguardar determinados períodos de tempo. Esses atrasos podem ser úteis para sincronizar o processo de encerramento da conexão com o banco de dados e garantir uma finalização adequada.

No final do código, realiza-se a finalização da conexão por meio do método *'end*()' e, em seguida, destruímos completamente o objeto de conexão com o banco de dados usando o método '*destroy*()'. Essas etapas são cruciais para liberar recursos e garantir a integridade do sistema. Após todas essas etapas, o código realiza uma verificação condicional para determinar se a consulta retornou algum resultado.

Caso tenha sido encontrado algum registro na tabela de inversores, esses registros são retornados como resultado. Caso contrário, o valor *false* é retornado, indicando a ausência de dados. Em suma, esse trecho de código representa uma funcionalidade robusta para acessar informações dos inversores por meio de uma conexão segura com o banco de dados. Ele foi cuidadosamente desenvolvido para garantir a integridade dos dados e fornecer respostas precisas aos clientes como mostrado na Figura 10.

Figura 10– Código buscar respostas no MySQL

```
const getDadosClientes = async () => {
  const connection = await createConnection();
  const [rows] = await connection.execute('SELECT title, description FROM DadosClientes');
  delay(1000).then(async function() {
    await connection.end();
    delay(500).then(async function() {
      connection.destroy();
    });
  });
  if (rows.length > 0) return rows;
  return false;
```

Fonte: próprio autor (2023).

3.7 Desenvolvimento do '*front end* '

Nesse projeto, optou-se ao desenvolvimento do código de '*front-end* ', que permite uma interação direta e amigável com os usuários. Ao enviar uma lista de opções para seleção, buscamos oferecer uma experiência intuitiva e prática. Um dos principais recursos é a possibilidade de selecionar o dispositivo desejado para obter informações específicas.

Os nomes Amanda e Kleber representam exemplos fictícios de usuários que podem realizar essa escolha. Após a seleção da instalação desejada, o programa entra em ação, exibindo uma lista de opções personalizadas. Por exemplo, é possível verificar a geração ou o status do inversor. Essa abordagem flexível permite que os usuários acessem dados relevantes de forma simples e eficiente.

Além da funcionalidade. Busca-se sobretudo tornar a comunicação com o '*chatbo*t ' um processo interativo e agradável. Com base nessa visão, todo o código foi projetado para ser hospedado em um servidor VPS garantindo uma experiência contínua e de qualidade aos usuários. É importante mencionar que o exemplo fornecido refere-se a uma condição específica para a exibição de lista no WhatsApp. Contudo, a comunicação com o *'bot*' pode ser adaptada para diferentes plataformas e métodos, como a interação por números de escolhas mostrado na Figura 11.

Figura 11– Código '*Front end* ' interação do usuário final

```
const pedido = [{title:'amanda', description: 'Escolha as
opções'},{title:'kleber', description: 'Escolha as opções'},{title:'Geraldo
rosa', description: 'Escolha as opções'},];
  const chat = await msg.getChat();
  chat.sendStateTyping();

  if (msg.body !== "" && msg.type !== 'list_response' && msg.type !==
'location' && msg.body !== 'amanda' && msg.body !== 'kleber' && msg.body !==
'Geraldo rosa' ) {
    let sections = [{title:'bot',rows:pedido}];
    let list = new List('Olá ' + nomeContato + ', tudo bem? Escolha os itens
do seu pedido selecionando uma das opções do menu','consultar
dados',sections,'STATUS');
    client.sendMessage(msg.from, list);
  }
  if (msg.body === 'amanda\nEscolha as opções' && msg.type !== 'location') {
    let sections = [{title:'BOT',rows:DadosClientes}];
    let list = new List(nomeContato + ', escolha um dos itens ','consultar
dados',sections,'SATUS');
    client.sendMessage(msg.from, list);

    if (msg.body === 'Kleber\nEscolha as opções' && msg.type !== 'location') {
      let sections = [{title:'BOT',rows:DadosClientes}];
      let list = new List(' nomeContato + ', escolha um dos itens ','consultar
dados',sections,'SATUS');
      client.sendMessage(msg.from, list);

      if (msg.body === 'Geraldo rosa\nEscolha as opções' && msg.type !==
'location') {
        let sections = [{title:'BOT',rows:DadosClientes}];
        let list = new List( nomeContato + ', escolha um dos itens
','consultar dados',sections,'SATUS');
        client.sendMessage(msg.from, list);
```

Fonte: próprio autor (2023).

3.8 Resposta do cliente e busca no banco de dados

Para selecionar o mês com precisão e obter os rendimentos de cada período, foi desenvolvida uma lógica um pouco diferente da mostrada anteriormente. Agora, trata-se de uma lista enviada em que o usuário seleciona apenas o número correspondente à opção mostrado na Figura 12

Figura 12- Programação gera as opções para o usuário

```
const meses = {
'1': 'Janeiro',
'2': 'Fevereiro',
'3': 'Março',
'4': 'Abril',
'5': 'Maio',
'6': 'Junho',
'7': 'Julho',
'8': 'Agosto',
'9': 'Setembro',
'10': 'Outubro',
'11': 'Novembro',
'12': 'Dezembro',
};
client.on('message', async (message) => {
    if (message.body.toLowerCase() === 'meses') {
      let options = 'Escolha um número para selecionar o mês:\n';
      for (const key in meses) {
        options += `${key}. ${meses[key]}\n`;
      }
      await message.reply(options);
    } else if (meses[message.body]) {
      const mesSelecionado = meses[message.body];
      const query = `SELECT * FROM respostas WHERE mes =
'${mesSelecionado}'`;

      connection.query(query, (err, results) => {
        if (err) {
          console.error('Erro ao consultar o banco de dados:', err);
          return;
        }
```

```
        if (results.length > 0) {
          const resposta = results[0].resposta;
          await message.reply(`A resposta do mês de ${mesSelecionado} é:
${resposta}`);
        } else {
          await message.reply('Não foi encontrada uma resposta para o mês
selecionado.');
        }
      });
    }
  });
```

Fonte: próprio autor (2023).

3.9 Iniciando a automação em Python

Python é uma das linguagens mais populares e amplamente utilizadas devido à sua facilidade de compreensão e vasta comunidade de desenvolvedores. Foi escolhida como a linguagem para a automação devido a essas vantagens. Para utilizá-la, foi necessário instalar o pacote Selenium, que é usado para gerenciar pacotes no Python, por meio do pip, que é uma ferramenta para gerenciamento de pacotes de software em Python.

Além disso, o ChromeDriver desempenha um papel essencial na execução de comandos automáticos na VPS (*Virtual Private Server*). A automação em Python depende desse driver para executar o navegador Google Chrome e realizar tarefas automatizadas dentro dele. O ChromeDriver atua como uma ponte entre o script de automação em Python e o navegador Chrome, permitindo a interação programática com o navegador e a realização de tarefas específicas dentro dele. Essa dependência do ChromeDriver garante que a automação seja realizada de maneira eficiente e precisa.

3.10 Executando o código da automação.

Durante o desenvolvimento do projeto, foi implementado um código que automatiza a abertura do navegador da web de forma invisível. Esse código permitiu acessar os dados necessários através da interação com os botões identificados por seus IDs. A automação foi executada até que fosse possível baixar o arquivo Excel fornecido pelo próprio site do inversor, instalação das bibliotecas mostrada na Figura 13.

Figura 13– Código Python para baixar arquivo web

```
from selenium import webdriver
from selenium.webdriver.common.keys import Keys
from selenium.webdriver.chrome.options import Options
import time


chrome_options = Options()
chrome_options.add_argument("--headless") )
nevegador = webdriver.Chrome()
navegador.get('https://exemplo.com')
```

Fonte: próprio autor (2023).

Todos os dados de acesso ao sistema estão integrados no próprio código como mostrado na Figura 14. Essa abordagem permite simular a interação humana, pois os dados de *login* são inseridos automaticamente e a plataforma é acessada de forma programática.

Figura 14– Código Python autenticar acesso ao sistema

```
email_element = nevegador.find_element_by_id('id do campo email')
email_element.send_keys('seu email')

senha_element = nevegador.find_element_by_id('id da senha')
senha_element.send_keys('sua_senha')

senha_element.send_keys(Keys.ENTER)

botao_element = navegador.find_element_by_id('id_do_botao').click()

time.sleep(5)

navegador.quit()
```

Fonte: próprio autor (2023).

Dessa forma, o ‘*download* ‘do arquivo é feito utilizando a capacidade do programa para realizar a tarefa de maneira similar a um ser humano. É relevante destacar que todo esse

processo ocorre de forma discreta, sem exibir nenhuma alteração visível no navegador utilizado no servidor.

3.11 Inserindo os dados do Excel no banco de dados MySQL

Para inserir os dados do Excel no sistema de banco de dados, será utilizado um código Python com três bibliotecas importantes: ‘*pandas*’, ‘*mysql.connector*’ e ‘*schedule*’. Cada uma dessas bibliotecas desempenha uma função específica para garantir o funcionamento adequado do sistema.

O objetivo é automatizar o processo, executando todas as funções no servidor principal e alimentando o banco de dados MySQL para que os dados possam ser consultados posteriormente. Dessa forma, facilita-se a gestão e análise dos dados obtidos.

Uma conexão bem-sucedida com o banco de dados MySQL é de extrema importância para o projeto. Para garantir essa conexão, foi desenvolvido um código específico que contém todas as informações necessárias para estabelecer a ligação com o sistema. Essa conexão é essencial para que se possa realizar a inserção dos dados provenientes do arquivo Excel no banco de dados. Além disso, o código também inclui a funcionalidade de leitura do arquivo Excel através do ‘*pandas’*. Por meio dessa leitura, pode-se obter as informações contidas no arquivo e transferi-las de forma eficientes para o ambiente web.

Ao realizar essas etapas, garante-se a sincronização adequada entre o banco de dados e os dados contidos no arquivo Excel, código mostrado na Figura 15. Dessa forma, pode-se utilizar as informações armazenadas no banco de dados MySQL para consultas, análises e outras operações necessárias ao projeto.

Figura 15– Código Python para colocar dados no MySQL

```
def inserir_dados():
    config = {
        'user': 'usuario',
        'password': 'senha',
        'host': 'localhost',
        'database': 'banco_de_dados',
```

```
        'raise_on_warnings': True
    }
    dados_excel = pd.read_excel('caminho/para/o/arquivo.xlsx')
    try:
        conn = mysql.connector.connect(**config)
        cursor = conn.cursor()

        for _, row in dados_excel.iterrows():


          MES_instalacao = row['mes da Instalação']


            nome_instalacao = row['Nome da Instalação']
            producao_kwh = row['Produção(kWh)']

            query = "INSERT INTO tabela(mes da Instala,nome_instalacao,
producao_kwh) VALUES (%s, %s, %s)"
            values = (MES_instalacao,nome_instalacao, producao_kwh)
            cursor.execute(query, values)

        conn.commit()
        print("Dados inseridos com sucesso!")

    except mysql.connector.Error as err:
        print(f"Erro ao conectar ao banco de dados: {err}")

    finally:
        if conn.is_connected():
            cursor.close()
            conn.close()
            print("Conexão encerrada.")
```

Fonte: próprio autor (2023).

3.12 Funcionamento do código e agendamento de tarefas

A biblioteca ‘*schedule*’ é uma ferramenta essencial para agendar tarefas no funcionamento de um código em Python. Com ela, pode-se programar a execução de tarefas de forma pontual e organizada.

Por exemplo, utilizando a biblioteca '*schedule*', pode-se agendar a função de *download* de arquivos para executar todos os dias às 00:00. Em seguida, pode-se utilizar o agendamento para

estabelecer a conexão com o banco de dados e enviar o arquivo para o MySQL às 00:30. Posteriormente, às 05:00, pode-se agendar a exclusão do arquivo baixado, permitindo que o próximo arquivo tenha o mesmo nome e o código possa reconhecê-lo para leitura.

Dessa maneira, o uso do agendamento proporcionado pela biblioteca '*schedule*' permite automatizar essas tarefas, garantindo a execução correta e no momento adequado, facilitando a organização e o fluxo de trabalho em nosso código Python, mostrado na Figura 16.

Figura 16– Código Python agendamento de execução

```
                    import schedule
import time

def baixarArquivo():
    print("Executando BAIXAR ARQUIVO...")

# Agendando a tarefa para ser executada 00:00
schedule.every().day.at("00:00").do(baixarArquivo)

while True:
    schedule.run_pending()
time.sleep(1)
def Su  birArquivo():
    print("Executando subir ARQUIVO...")

# Agendando a tarefa para ser executada 00:30
schedule.every().day.at("00:30").do(SubirArquivo)

while True:
    schedule.run_pending()
    time.sleep(1)
```

Fonte: próprio autor (2023).

3.13 Exclusão do arquivo automaticamente

A sincronização e a ordem de execução dos processos são de extrema importância no projeto. Após baixar o arquivo contendo os dados do dia, o nome desse arquivo é especificado dentro do código responsável por enviar os dados para o banco de dados. Uma vez que as informações do arquivo são inseridas no MySQL, pode-se removê-lo para que o próximo

arquivo seja baixado no próximo dia. A função para isso, escrita em Python, mostrada na Figura 17, é encarregada de executar e finalizar todas as etapas do projeto. Todas essas tarefas são integradas em um único programa, que é responsável por executar o projeto de forma completa e organizada.

Figura 17– Código Python para excluir o arquivo baixado

```
import schedule
import time
import os

def excluir_arquivoBaixado():
    arquivo = "arquivoBaixado.xlsx"
    if os.path.exists(arquivobBaixado):
        os.remove(arquivoBaixado)
        print(f"Arquivo {arquivoBaixado} excluído com sucesso.")
    else:
        print(f"Arquivo {arquivoBaixado} não encontrado.")
schedule.every().day.at("05:00").do(excluir_arquivo)

while True:
    schedule.run_pending()
    time.sleep(1)
```

Fonte: próprio autor (2023).

4 RESULTADO E DISCUSSÃO

Este projeto tem como objetivo integrar o WhatsApp com um banco de dados para viabilizar a obtenção de informações através de uma simples conversa. Os dados serão coletados diretamente dos inversores, baixadas na VPS e alimentarão o banco de dados MySQL automaticamente, proporcionando aos usuários a possibilidade de consultar informações 24 horas por dia, de maneira precisa e sem complicações. Com essa solução, espera-se trazer mais praticidade e eficiência para o acesso às informações dos inversores.

4.1 Resultado da interação do '*chatbot* ' com usuário

O resultado alcançado com o '*Front-end* ' demonstra uma opção mais interativa, que consiste em uma listagem para seleção, permitindo que o usuário faça suas escolhas de forma simples e prática, conforme ilustrado na figura 18. Essa abordagem oferece uma solução satisfatória para o projeto, fornecendo uma maneira conveniente de consultar os dados do inversor armazenados no MySQL. Além disso, a figura 19 apresenta a opção de lista, na qual são exibidos os nomes das instalações disponíveis no sistema e que podem ser consultados.

Figura 18- Imagem do '*Front end* ' no formato lista

Fonte: próprio autor (2023).

Figura 19- Opções das instalações instaladas

Fonte: próprio autor (2023).

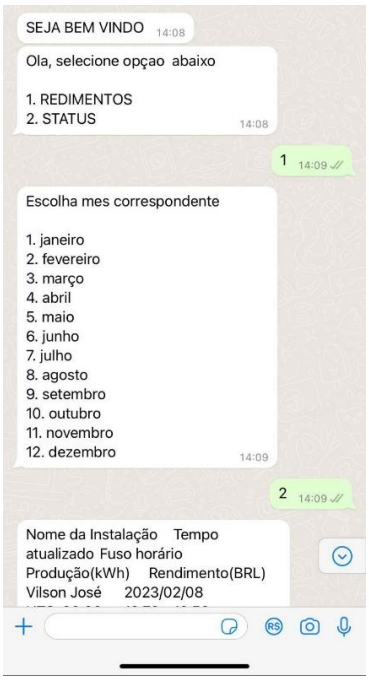
Após o usuário selecionar a instalação que deseja verificar, são apresentadas duas opções: ‘status’ e ‘consultar dados’. Ao escolher ‘consultar dados’, o usuário tem acesso à quantidade de energia gerada durante o dia. Já ao selecionar "status", ele pode verificar se o dispositivo, inversor está *offline.* Essas opções fornecem ao usuário informações importantes sobre o funcionamento da instalação fotovoltaica. Ao consultar os dados, ele pode acompanhar o desempenho da geração de energia ao longo do dia, o que permite uma análise do aproveitamento.

4.2 Obtendo resultados esperados da comunicação com '*chatbot* '

O processo de automação com Python foi altamente bem-sucedido. Através da integração de código, foi possível desenvolver um programa que opera de forma automática dentro da VPS. A lógica implementada permite coletar dados dos websites dos inversores e armazená-los automaticamente no banco de dados MySQL. Esses dados são processados e utilizados para fornecer informações valiosas aos usuários do sistema. Com a coleta automática dos dados dos inversores, elimina-se a necessidade de intervenção manual e agiliza-se a disponibilização de informações atualizadas

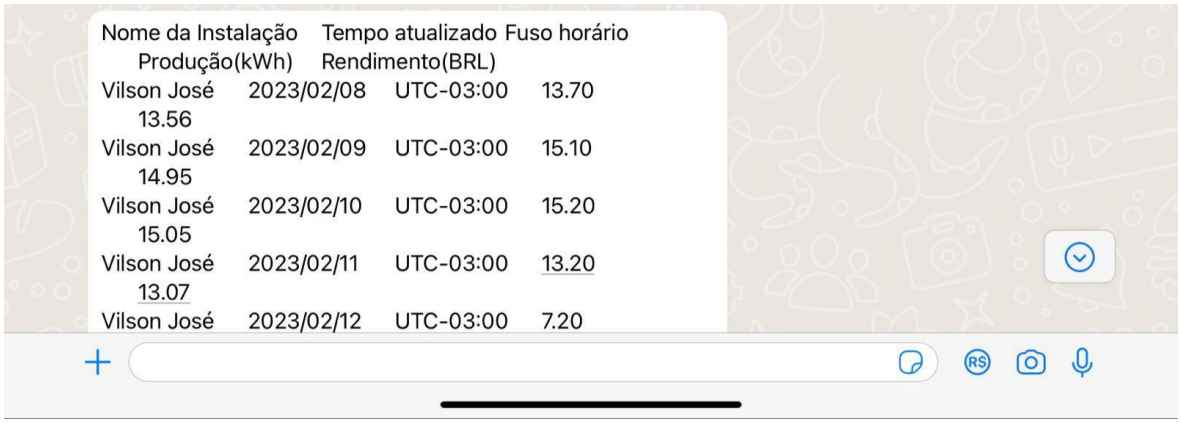
Além disso, os resultados obtidos ao selecionar o mês desejado para obter os dados de rendimento foram positivos, como mostrado na Figura 20. No entanto, é possível aprimorar a apresentação dos resultados, fornecendo informações mais organizadas. Na Figura 21, pode-se observar um exemplo dos dados do mês de fevereiro.

Figura 20- Interação com opção números no '*Front end* '



Fonte: próprio autor (2023).

Figura 21- Resultado do rendimento do mês de fevereiro



Fonte: próprio autor (2023).

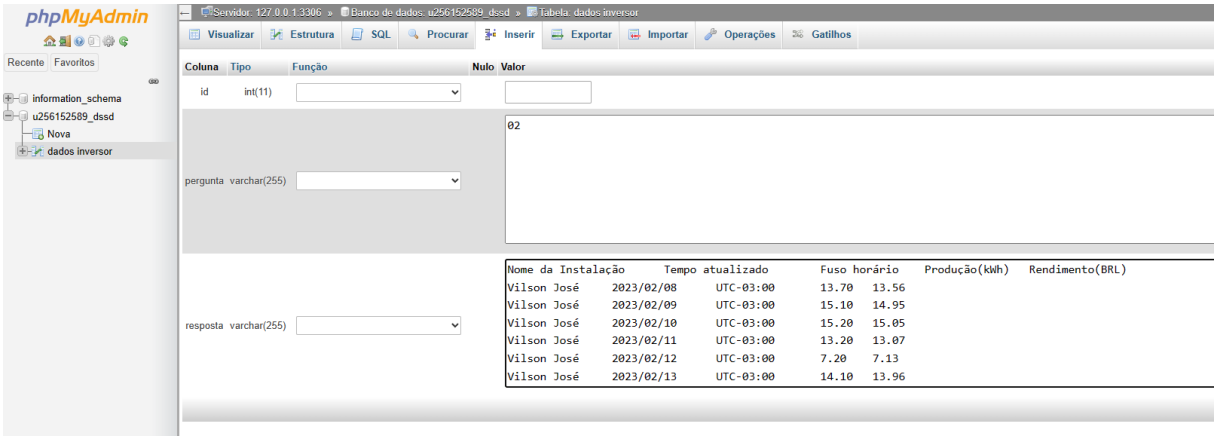
4.3 Resultado busca de dados no MySQL

Os dados dentro do banco de dados MySQL possuem uma configuração simples que permite que o programa interprete e obtenha as informações corretas para o usuário. Essa configuração

é baseada em perguntas e respostas, onde o sistema busca a resposta e, ao encontrá-la, recupera as informações relevantes e as envia ao usuário solicitante.

Como ilustrado na Figura 22, todo o sistema gira em torno dos resultados, pois os dados armazenados são o fator-chave e as informações obtidas são provenientes do website do inversor.

Figura 22- Pergunta e reposta dentro MySQL



Fonte: próprio autor (2023).

5 CONCLUSÃO

A integração de uma ferramenta de fácil acesso ao sistema dos inversores proporcionou melhorias significativas na análise e na tomada de decisões. Agora, os usuários podem acessar todos os dados dos inversores por meio do WhatsApp, agilizando o processo de análise sem a necessidade de abrir o site oficial para obter informações. Com apenas uma mensagem, é possível obter todas as informações necessárias.

A utilização dessas soluções tecnológicas torna o monitoramento e a otimização da produção de energia solar mais viáveis e eficientes. Os dados coletados permitem uma análise precisa do desempenho dos sistemas fotovoltaicos, facilitando a identificação de falhas, a implementação de melhorias e a maximização dos benefícios para os usuários.

Essa integração entre tecnologia e energia solar traz benefícios substanciais para o setor, uma vez que simplifica o acesso aos dados e proporciona uma visão detalhada do funcionamento dos sistemas. Os usuários podem tomar decisões mais informadas, identificar possíveis problemas de forma rápida e implementar medidas corretivas com agilidade.

Sugestões de trabalhos futuros: Para continuar avançando na tecnologia, como trabalho futuro, é recomendável realizar uma integração mais avançada com o processo em Python, focando na criação de funções mais rápidas e eficientes. Além disso, é fundamental aprimorar a comunicação do '*chatbot '* com o usuário final, garantindo uma interação mais fluida e natural. Para isso, é necessário refinar os códigos, eliminando bugs e otimizando o desempenho do sistema. É totalmente viável aprimorar o trabalho e torná-lo ainda melhor e mais tecnológico para atender às demandas do mercado. Isso pode incluir a implementação de algoritmos mais eficientes, a adoção de boas práticas de programação e a utilização de bibliotecas e ferramentas avançadas.

REFERÊNCIAS

AMAZON WEB SERVICES. Disponível em: https://aws.amazon.com/ Acesso em: 03 de junho de 2023.

BRAGA, R. P. **Energia Solar Fotovoltaica: Fundamentos e Aplicações**. UFRJ, 2008.

CANALTECH. **WhatsApp anuncia novas ferramentas para empresas em sua plataforma de mensagens.** Canaltech, São Paulo, 10 jun. 2022. Disponível em:https://canaltech.com.br/empresa/whatsapp/. Acesso em: 14 abr. 2023.

CEPEL; CRESESB; **Manual de Engenharia para Sistemas Fotovoltaicos**. Edição Especial PRC-PRODEEM. Rio de Janeiro, 2004. Disponível em: . Acesso em: 12 Abr. 2023.

DIGITALOCEAN. Disponível em: https://www.digitalocean.com/Acesso em: 03 de junho de 2023.

DO NASCIMENTO, **Cássio Araújo. Princípio de funcionamento da célula fotovoltaica**. Diss. Universidade Federal de Lavras, 2004.

EXPRESS.JS - **Fast, unopinionated, minimalist web framework for Node.js**. Disponível em: https://expressjs.com/. Acesso em: 03 de junho de 2023.

FERNANDES, Euclécio Alves et al. **A evolução da comunicação impactada pela tecnologia**. Ideias e Inovação-Lato Sensu, v. 3, n. 2, p. 93-102, 2016.

GOOGLE CLOUD PLATFORM.Disponível em: https://cloud.google.com/Acesso em: 03 de junho de 2023.

HEROKU. Disponível em: https://www.heroku.com/Acesso em: 03 de junho de 2023.

META. WhatsApp. Disponível em: https://about.meta.com/br/technologies/whatsapp/. Acesso em: 2 de junho de 2023.

MICROSOFT AZURE.Disponível em: https://azure.microsoft.com/Acesso em: 03 de junho de 2023.

NODE.JS. Local: Node.js Foundation, 2023. Disponível em: https://nodejs.org/en/. Acesso em: 01 de junho de 2023.

NODEJS. Site oficial do Node.js. Disponível em: https://nodejs.org/en. Acesso em: 05 abr. 2023.

NPM. Site oficial do NPM. Disponível em: https://www.npmjs.com. Acesso em: 08 abr. 2023.

NPMJS.COM. 2023. Disponível em: https://www.npmjs.com Acesso em: 01 de junho de 2023.

OLIVEIRA, Othon Garcia; OLIVEIRA, Rafael Henrique; GOMES, Renato Oliveira. **Energia solar: um passo para o crescimento.** REGRAD-Revista Eletrônica de Graduação do

UNIVEM-ISSN 1984-7866, v. 10, n. 01, p. 377-389, 2017.

PARANHOS. **Processo de Evolução Tecnológica**. 20. Out. 2013. Disponível em: https://www.oficinadanet.com.br/post/11764-processo-de-evolucao-tecnologica .Acesso em: 2. Jun. 2023

PYTHON. Disponível em https://www.python.org/download/Acesso em: 03 de junho de 2023.

REZENDE, Jaqueline Oliveira. **A importância da Energia Solar para o Desenvolvimento Sustentável**. São Paulo: Atena, 2019.

WEBJS. **Guia prático de desenvolvimento web**. Disponível em: http://wwebjs.dev/guide. Acesso em: 12 abr. 2023.