**SÃO PAULO TECH SCHOOL**

**MONITORAMENTO DE HARDWARE DE COMPUTADOR EM MANUTENÇÕES DE METRÔ**

**NOME:** Eduardo Castrillo, Felipe Magalhães,

Matheus Rabello, Renan Alves, Samarah Costa.

**São Paulo**

**2024**

Sumário

[Contexto 2](#_Toc169296704)

[Justificativa 5](#_Toc169296705)

[Objetivo 6](#_Toc169296706)

[Escopo 7](#_Toc169296707)

[Riscos e Restrições 9](#_Toc169296708)

[Premissas 9](#_Toc169296709)

[Especificação Tecnica 10](#_Toc169296710)

[Especificação Funcional 12](#_Toc169296711)

[StoryBoard 15](#_Toc169296712)

[Proto Persona 15](#_Toc169296714)

[Visita 16](#_Toc169296716)

[Lean UX Canvas 16](#_Toc169296718)

[Planner 17](#_Toc169296719)

[Product Backlog 18](#_Toc169296721)

[MER 21](#_Toc169296722)

[Diagrama de classes 22](#_Toc169296723)

[Diagrama de sequência 22](#_Toc169296724)

[BPMN 23](#_Toc169296725)

[Diagrama de solução 24](#_Toc169296726)

[Agenda de controle 24](#_Toc169296727)

[Wireframe 25](#_Toc169296728)

[Especificação log 26](#_Toc169296729)

[Especificação inovação 29](#_Toc169296730)

[Especificação Slack 30](#_Toc169296731)

[Script de instalação 32](#_Toc169296732)

[GitHub 37](#_Toc169296733)

[PPTX 37](#_Toc169296734)

# Contexto

O metrô é um dos sistemas de transportes mais utilizado no Brasil e no mundo e possui um lugar de destaque em nossa sociedade e economia. De acordo com a União Internacional de Ferrovias (UIF) o Brasil ocupa o **23º lugar no ranking mundial de extensão de malha ferroviária.** Em 2022, a rede metroviária atingiu a marca de 2,3 bilhões de passageiros transportados no ano, tendo uma média de 7,8 milhões de passageiros por dia útil. Também chegou na faixa de 38,2 mil funcionários e 629 estações, sendo 361 dessas estações de metrô.

Em 2017, a rede metroviária atingiu a marca de 1,3 bilhão de passageiros transportados, sendo que o Metrô de São Paulo foi responsável pelo transporte de 1,1 bilhão desses passageiros, destacando-se mundialmente pelos resultados obtidos na produção e na qualidade do serviço prestado no transporte público de passageiros sobre trilhos.

* 91 São Paulo
* 52 Rio de Janeiro
* 37 Recife
* 22 Porto Alegre
* 27 Brasília
* 34 Salvador
* 19 Belo Horizonte
* 20 Fortaleza
* 24 Natal
* 11 Teresina

Ainda em 2022, o Metrô de São Paulo com aproximadamente **4,5 bilhões de reais**. O resultado representa um aumento de 43,4% em relação a 2021 (3,4 bilhões naquele ano), e se deve ao aumento da demanda por transporte público, à eficiência na gestão da empresa e aos investimentos em infraestrutura.

De acordo com a Companhia do Metropolitano de São Paulo (Metrô)

“Em 2017, a rede metroviária atingiu a marca de 1,3 bilhão de passageiros transportados, sendo que o Metrô de São Paulo foi responsável pelo transporte de 1,1 bilhão desses passageiros, destacando-se mundialmente pelos resultados obtidos na produção e na qualidade do serviço prestado no transporte público de passageiros sobre trilhos.”

Entretanto no Brasil, os serviços metroferroviarios tiveram muitas falhas, impactando muitas pessoas e a própria empresa, além disso **54% das falhas** são originadas de **problemas técnicos, 22% de falhas humanas 24% de fatores externos.**

Como por exemplo, no dia 17 de Fevereiro de 2020, a linha-3 vermelha do metro partiu com as **portas abertas** **com os passageiros embarcados** em horário de pico, segundo ao Diário dos Trilhos “Um trem do Metrô circulou com porta aberta entre as estações Artur Alvim e Corinthians-Itaquera. Uma passageira filmou parte do trajeto e o vídeo foi divulgado no Facebook (<https://www.facebook.com/saopaulodepre/videos/vb.109715172520016/215000302882968/?type=2&theater>). Por meio de nota o Metrô informou que vai apurar as causas que levaram o trem a circular desta forma e se foi falha mecânica ou humana.” Até o momento não encontramos uma segunda nota do Metrô revelando o ocorrido.

Outro caso ainda na linha-03 vermelha, no dia 11 de Outubro de 2022 o Metrô **parou** para **desembarcar os passageiros** de maneira **desalinhada com a plataforma** como na foto:



Segundo ao Diário dos Trilhos, “É possível ver que a posição em que o trem parou e abriu as portas, impede que as pessoas entrem ou saiam deste trem. Em seguida o sinal de fechamento das portas é ativado, a porta fecha.”

No dia 24 de Junho de 2023, a linha-05 lilás teve **um descarrilhamento** entre as estações Capão Redondo á Santo Amaro. Segundo ao Diário do Transporte, “Um trem da linha 5-Lilás de metrô, operada pela ViaMobilidade, descarrilhou por volta das 20h40 deste sábado, 24 de junho de 2023, entre as estações Santo Amaro e Capão Redondo, na zona Sul de São Paulo. [...] as operações começaram normalmente, mas a noite foi marcada por problemas para os passageiros [...]”

Com esse tipo de falhas, anualmente cerca de **42,44% de lucro** são **comprometidos** como por exemplo, em 2022 aproximadamente **2 bilhões** de reais foram **perdidos** devido aos incidentes e multas geradas pelos problemas apresentados,porém com o **sistema de monitoramento do metrô** é possível **mitigar** e até mesmo **eliminar** tais **incidentes**, e ainda sim, em casos de incidentes é possível ter uma **solução de contorno** rápida e eficiente.

Com isso, o projeto abordado propõe uma **monitoria de hardware** que **garante** o funcionamento das máquinas durante o tempo de serviço, reduzindo problemas e falhas técnicas.

# Justificativa

Mitigar incidentes e aumentar a produtividade de rotação dos metrôs em 20%, recuperar lucro em até 40%, através do nosso sistema NOC para monitoramento.

# Objetivo

A ViaTech tem o foco em implementar um sistema de monitoramento NOC para garantir a segurança, eficiência e confiabilidade das operações ferroviárias em busca de melhorar a eficiente da rotação dos metrôs em 20%.

# Escopo

**Descrição Resumida:**

Desenvolver um website institucional responsivo, e criar um sistema de monitoramento de hardware do computador central de metrôs, para garantir o funcionamento da máquina durante o tempo de serviço.

**Equipe do projeto:**

* Eduardo Castrillo
* Felipe Magalhães
* Matheus Rabello
* Renan Alves
* Samarah Costa

**Partes Interessadas:**

* Engenheiros de Ti
* Técnicos de Manutenção
* Autoridades Reguladoras
* Empresas de Metroferroviário
* Equipe de manutenção
* Equipe operacional do metrô
* Governo brasileiro

**Recursos Necessário:**

* 2 Desenvolvedor Backend
* 1 Especialistas em Banco de Dados
* 1 Frontend
* 1 Gestor de projetos
* Serviço em Cloud

**Resultados Esperados:**

* Reduzir em até X% a quantidade de incidentes, e recuperar até 40% do lucro.

**Sistema de monitoramento funcional e eficiente:**

Monitoramento dos componentes de hardware em tempo real.

Detecção de anomalias e falhas.

Geração de alarmes e notificações.

**Aumento da confiabilidade:**

Detecção precoce de falhas.

Prevenção de falhas graves.

**Melhoria da segurança:**

Identificação de riscos potenciais.

Prevenção de acidentes.

**Macro Cronograma – 80 Dias**

* Levantamento de requisitos: 10 Dias;
* Elaboração do Plano do Projeto: 15 Dias;
* Implementação do banco de dados para armazenamento dos dados: 15 Dias;
* Desenvolvimento da interface do usuário; - 30 Dias
* Integração dos componentes do sistema; - 20 Dias

**Limites e Exclusões**

* Não abrangemos Sensores (Temperatura, posição e etc...);
* Não será monitorado a Infraestrutura do sistema metroviário (trilhos, estações, sinalização);
* O sistema não terá a capacidade de prevenir falhas no hardware, apenas alertar sobre possíveis problemas com base nos dados coletados.
* O sistema não será responsável por reparos ou substituições de componentes de hardware danificados.

# Riscos e Restrições

* Servidor em nuvem ficar fora do ar;
* Desviar o projeto do escopo original;
* Prazos muito curtos;
* Ferramenta nova;
* Software novo;
* Falhas no software;
* Mudanças nos requisitos devido a solicitação do cliente;
* Dificuldades com a comunicação;
* Falta de habilidade necessária para determinadas tarefas;
* Requisitos inadequados ou incompletos;
* Sobrecarga de trabalho;
* Responsividade não se adequar;
* O sistema será desenvolvido apenas para monitorar a CPU dos computadores que monitoram o metrô.

# Premissas

* O computador será compatível com o nosso software;
* O projeto iniciará com data prevista;
* A equipe estará disponível e terá habilidades necessárias para concluir as tarefas.

# Especificação Tecnica

Plataforma:

O projeto ViaTech foi desenvolvido para sistemas operacionais Windows e Linux, buscando atender as principais plataformas usadas em ambientes de trabalho.

Linguagem de Programação:

A linguagem de programação usada para a criação do projeto foi o Java, buscando compatibilidade com os sistemas operacionais alvos.

Monitoramento de Hardware:

Com o auxílio da API “Looca”, as informações de hardware, CPU, memória RAM, disco Rígido e entradas USB de cada máquina são capturadas e armazenadas no banco de dados MYSQL em intervalos regulares, e por fim, exibidas em uma dashboard.

Inovação:

Captura da temperatura de todos os componentes, visto que a alta temperatura afeta diretamente seu desempenho na máquina.

Interface de Usuário:

Uma interface intuitiva e amigável para nossos clientes.

As informações sobre os hardwares e o desempenho da máquina desejada são exibidas de forma clara e acessível.

Permitir que a empresa cadastre um funcionário e o nível de acesso que ele possui.

Notificações e Alertas:

O projeto conta com um sistema de notificação que alerta o funcionário responsável sobre problemas de desempenho, como a alta utilização da CPU, disco ou RAM, assim como a alta temperatura de algum desses componentes.

Segurança:

Garantir que todas as informações coletadas e processadas sejam tratadas com segurança e que a privacidade dos nossos clientes seja protegida.

Implementar medidas de segurança como permissões de acesso restritas, permissões essas que o próprio cliente pode definir o nível de acesso e controle.

# Especificação Funcional

**Wireframe:**

Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamente

Interface gráfica do usuário

Descrição gerada automaticamente

**Descrição:**

Uma das principais telas do sistema, nela é possível visualizar o desempenho dos hardwares monitorados em tempo real, possui uma interface intuitiva e de fácil entendimento do usuário.

**Como é feito:**

Uma tela estruturada em html css e chartjs, onde o cliente visualiza as principais KPI’S da máquina desejada com o auxílio de determinados gráficos para ajudar a compreensão do usuário.

**API Looca:**

Com o auxílio da API “Looca” é feita a captura dos dados de hardware dos componentes: CPU, memória RAM, disco Rígido e entradas USB.

**Banco de Dados MYSQL:**

Os dados coletados pela API Looca são armazenados no banco de dados ViaTech, onde posteriormente será utilizado para ser exibido na página dashboard.

**Funções:**

*Função*: “exibirEstacoes()”

*Objetivo*: Usuário selecionar a linha que deseja visualizar as estações.

*Função*: “clicarMenu()”

*Objetivo*: Usuário escrever o nome da determinada estação que deseja buscar e visualizar.

**KPI’S exibidas:**

N° de máquinas na linha;

N° de problemas na semana;

Porcentagem das estações que estão em alerta;

Estação com mais problemas;

Uso da CPU em %;

Uso de Disco em %;

Uso de RAM em %;

Temperatura;

Utilização de Disco em %;

Utilização de RAM em %;

Utilização da CPU em %;

USB’s conectados.

# StoryBoard

# 

# Proto Persona

# Tela de computador com texto preto sobre fundo azul Descrição gerada automaticamente

# Visita

Nossa visita foi constituída para tratarmos da solução do desenvolvimento da nossa aplicação web do nosso sistema ViaTech, então realizamos diversas perguntas em torno do desenvolvimento do projeto, onde conseguimos produzir User Story e proto persona de acordo com a solução que estávamos propondo para nosso cliente

# Pessoas posando para foto fazendo careta Descrição gerada automaticamente

# Lean UX Canvas

O Lean UX Canvas é uma ferramenta visual que faz parte da abordagem Lean UX, que visa integrar práticas ágeis com design centrado no usuário. Esse canvas é utilizado para mapear de forma rápida e colaborativa os principais elementos de um projeto, como personas, jornadas do usuário, problemas a serem resolvidos, soluções propostas e métricas de sucesso, facilitando a tomada de decisões e a comunicação entre as equipes.

Gráfico, Gráfico de mapa de árvore

Descrição gerada automaticamente

# Planner

Este é nosso planner onde foi feito todo nosso planejamento de tarefas e distribuídos entre todos os membros do nosso grupo para manter a organização do projeto:

https://tasks.office.com/sptech.school/pt-BR/Home/Planner/#/plantaskboard?groupId=b6bc7f9a-f925-40c2-b678-d2abc18849e4&planId=UtCZRuUtuku-V1OsVs97TGQAAkJV

# Aplicativo Descrição gerada automaticamente com confiança média

# Product Backlog

**Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente  
  
  
Interface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamenteInterface gráfica do usuário, Texto, Aplicativo, chat ou mensagem de texto

Descrição gerada automaticamente**

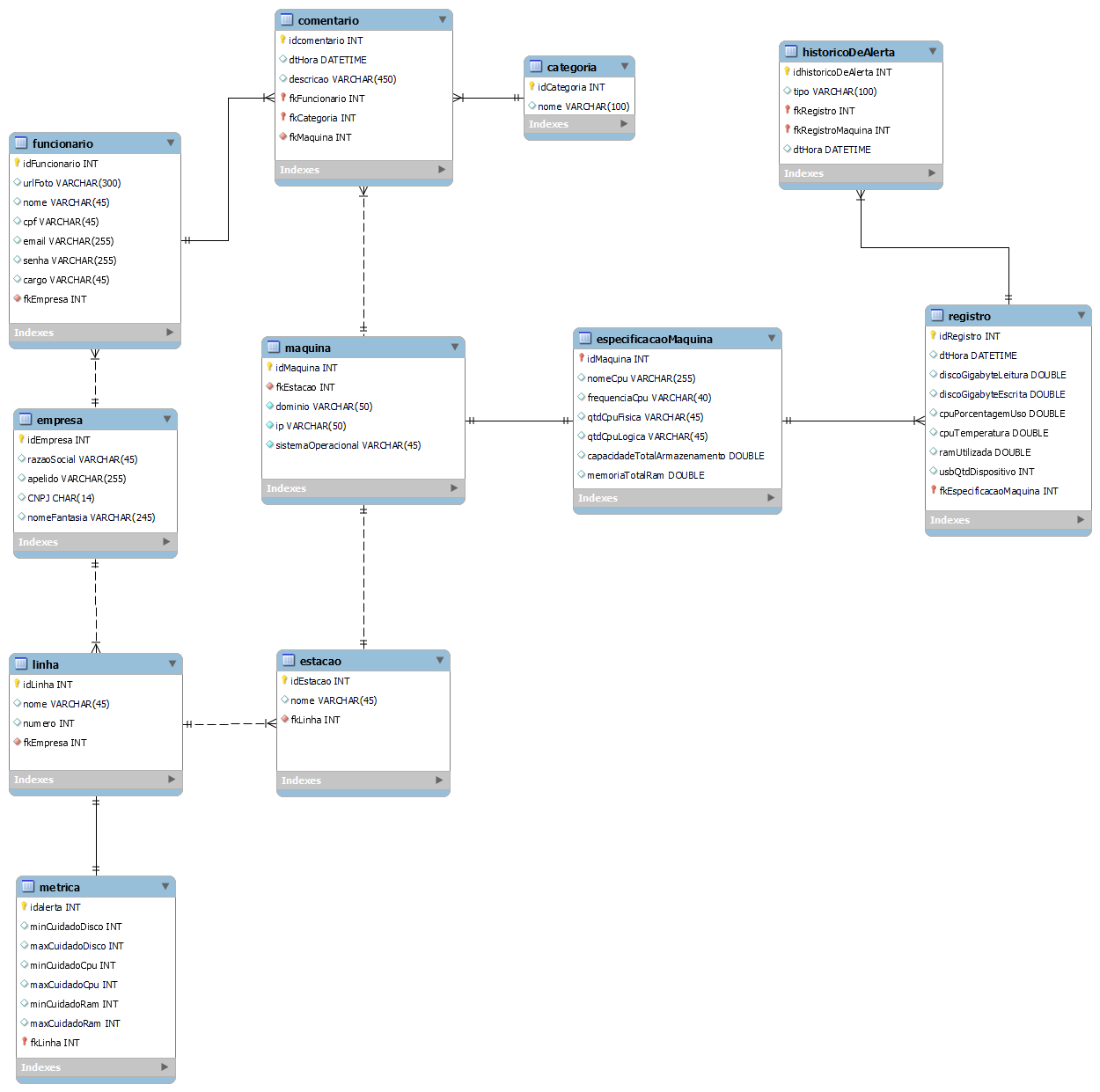
**Interface gráfica do usuário, Aplicativo

Descrição gerada automaticamenteInterface gráfica do usuário, Aplicativo

Descrição gerada automaticamente**

# MER

Nosso MER consiste em 11 tabelas, onde aqui está toda nossa estrutura do nosso sistema de desenvolvimento. Nosso Banco de Dados, onde aqui realizamos nossa integração com nosso sistema web (site) e com a aplicação do sistema java, que realizamos o monitoramento de todas as máquinas do nosso sistema



# Diagrama de classes

Diagrama

Descrição gerada automaticamente

Diagrama de sequência

Calendário

Descrição gerada automaticamente

# BPMN

Diagrama, Esquemático

Descrição gerada automaticamente

Diagrama

Descrição gerada automaticamente

# Diagrama de solução

Diagrama

Descrição gerada automaticamente

# Agenda de controle

Tabela

Descrição gerada automaticamente

# Wireframe

Interface gráfica do usuário

Descrição gerada automaticamente

# Especificação log

A classe Log é uma representação de um registro de log de um sistema. Ela encapsula detalhes como o sistema operacional, arquitetura, hostname, data, ID da máquina, mensagem de log, nível de log e código de status. Esta classe é utilizada para registrar e analisar eventos e erros ocorridos no sistema durante a etapa de login.

Atributos privados

**sistemaOperacional**: String

**arquitetura**: Integer

Descrição: A arquitetura do sistema operacional (32 ou 64 bits).

**hostname**: String

Descrição: O nome do host (nome da máquina) onde o log foi gerado.

**data**: String

Descrição: A data e hora em que o log foi gerado.

**idMaquina**: String

Descrição: Um identificador único da máquina onde o log foi gerado.

**mensagem**: String

Descrição: A mensagem de log que contém informações detalhadas sobre o evento ou erro.

**logLevel**: String

Descrição: O nível do log (INFO, ERROR).

**statusCode**: Integer

Descrição: O código de status associado ao log (200 se funcionar, 400 em caso de erro)

**Todos esses atributos possuem métodos Getters e Setters**

**Construtor:  
Texto

Descrição gerada automaticamente**

**Metodo ToString:**

Esse método retorna uma representação em formato de string dos dados do log, formatada de maneira legível.  
Texto

Descrição gerada automaticamente

Exemplos de saída:

Texto

Descrição gerada automaticamente

**Metodo salvarLog:**

O método salvarLog é responsável por salvar uma instância da classe Log em um arquivo de log. Ele formata a data atual para ser usada no nome do arquivo e grava a representação da instância de Log no arquivo.

Texto

Descrição gerada automaticamente

-A data atual é formatada no padrão yyyy-MM-dd para ser usada no nome do arquivo de log;

-O nome do arquivo de log é gerado usando a data formatada, seguindo o padrão ./logs/log-YYYY-MM-DD.txt.;

-Usa FileWriter com o modo de adição (append) para escrever a representação da instância de Log no arquivo;

-Captura ‘IOException’ para lidar com erros que podem ocorrer durante a escrita no arquivo, e caso aconteça o erro, imprime uma mensagem de erro no console.

**Exemplo de saida:**

**Interface gráfica do usuário, Texto, Aplicativo

Descrição gerada automaticamente**

# Especificação inovação

**Objetivo**

Visando uma melhor organização e comunicação entre os funcionários dos nossos clientes, disponibilizamos uma funcionalidade de comentários/anotações.

**Funcionalidades**

· Acesso: Qualquer funcionário com acesso ao nosso sistema poderá ler e fazer comentários/anotações personalizados em cada máquina.

· Organização: Para facilitar a visualização e o gerenciamento das anotações, os comentários serão organizados em categorias pré-definidas.

· Categorias: As categorias são projetadas para cobrir os principais aspectos de manutenção das máquinas, por exemplo:

1. Manutenção

2. Troca de peças

3. Observações

4. Incidentes externos

Melhor Comunicação: Facilitar a troca de informações entre funcionários, melhorando a colaboração.

Maior Organização: Anotações categorizadas permitem um acesso rápido e organizado às informações.

Rastreabilidade: Histórico de comentários que ajuda a rastrear problemas e soluções aplicadas.

Segurança: Comentários estão salvos no servidor o que garante a segurança das informações sem risco de perdê-las

# Especificação Slack

**Objetivo**

Utilizar o Slack como ferramenta para o envio de notificações quando os parâmetros dos componentes de Hardware estiverem fora das métricas estipuladas.

**Implementação**

Seguindo a documentação do Slack presente no site: https://api.slack.com/, foi criada uma classe contendo as configurações necessárias para integração entre o back-end da nossa aplicação com a API do Slack:

Texto

Descrição gerada automaticamente

É necessário um token criado no Slack para envio do método HTTP, para isso, foi criado uma consulta no banco de dados, mantendo o token em segurança.

Tela de computador com texto preto sobre fundo branco

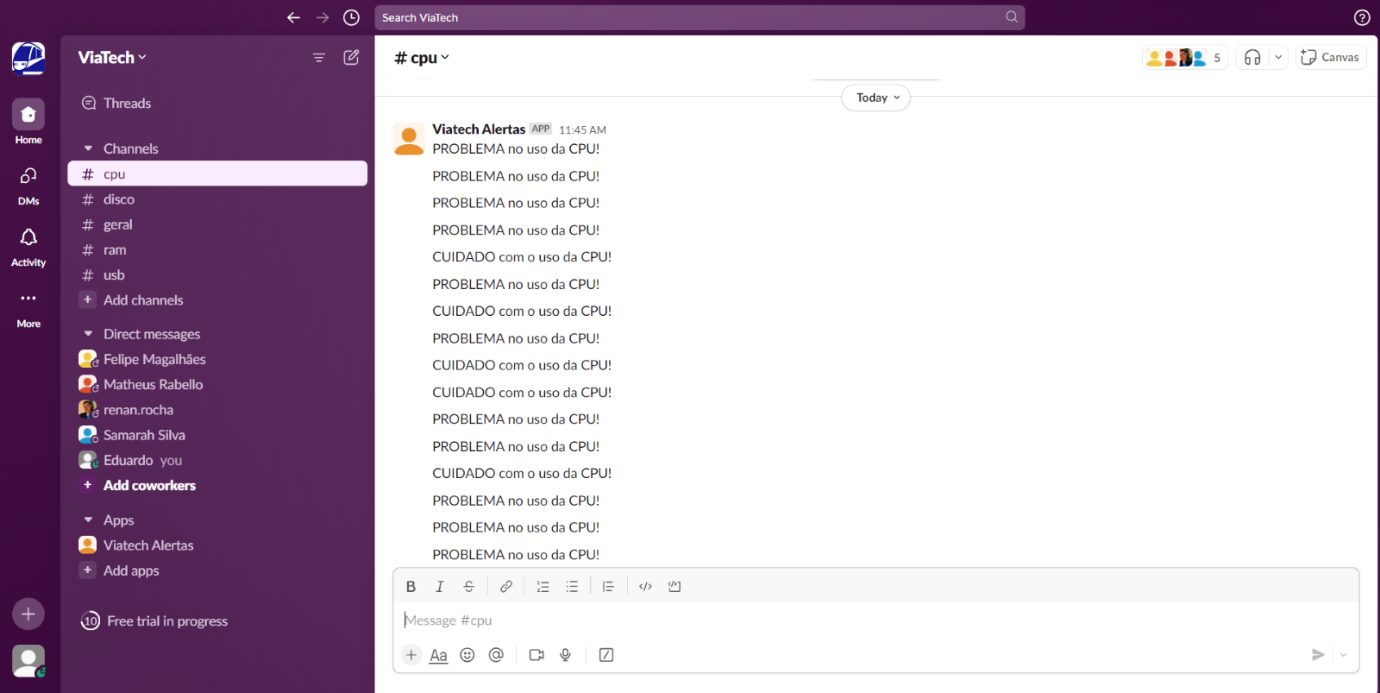
Descrição gerada automaticamente

Após isso, basta implementar a classe e chamar o método *sendMessage* passando como parâmetro a mensagem e o canal de destino.

Texto

Descrição gerada automaticamente

**Resultado da integração:**



# Script de instalação

Este script de instalação automatiza o processo de preparação do ambiente para o Sistema ViaTech:

#!/bin/bash

# Define colors

PURPLE='\033[0;35m'

NC='\033[0m'

GREEN='\033[0;32m'

RED='\033[0;31m'

YELLOW='\033[0;33m'

CYAN='\033[0;36m'

VERSAO=17

# Helper function for displaying messages

bot\_message() {

    echo -e "${CYAN}[Bot assistant]${NC}: $1"

}

bot\_success() {

    echo -e "${GREEN}[Bot assistant]${NC}: $1"

}

bot\_warning() {

    echo -e "${YELLOW}[Bot assistant]${NC}: $1"

}

bot\_error() {

    echo -e "${RED}[Bot assistant]${NC}: $1"

}

# Display a progress bar

progress\_bar() {

    echo -n "["

    for i in $(seq 1 $1); do

        echo -n "#"

        sleep 0.1

    done

    for i in $(seq 1 $((5 - $1))); do

        echo -n " "

    done

    echo -n "]"

    echo

}

clear

bot\_message "Olá Cliente, serei seu assistente para instalação do Java!"

bot\_message "Verificando aqui se você possui o Java instalado..."

sleep 2

java -version

if [ $? -eq 0 ]; then

    bot\_success "Você já tem o Java instalado!!!"

else

    bot\_warning "Opa! Não identifiquei nenhuma versão do Java instalado, mas sem problemas, irei resolver isso agora!"

    bot\_message "Confirme para mim se realmente deseja instalar o Java (S/N)?"

    read inst

    if [ "$inst" == "S" ]; then

        bot\_message "Ok! Você escolheu instalar o Java ;D"

        bot\_message "Adicionando o repositório!"

        progress\_bar 5

        sudo add-apt-repository ppa:linuxuprising/java -y

        clear

        if [ $VERSAO -eq 17 ]; then

            bot\_message "Preparando para instalar a versão 17 do Java. Confirme a instalação quando solicitado ;D"

            sudo apt install -y openjdk-17-jdk

            clear

            bot\_success "Java instalado com sucesso!"

        fi

    else

        bot\_warning "Você optou por não instalar o Java por enquanto, até a próxima então!"

        exit 0

    fi

fi

# Verificar se o Docker está instalado

if ! [ -x "$(command -v docker)" ]; then

    bot\_warning "Docker não está instalado. Instalando Docker..."

    progress\_bar 3

    sudo apt-get update

    sudo apt-get install -y apt-transport-https ca-certificates curl software-properties-common

    curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -

    sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

    sudo apt-get update

    sudo apt-get install -y docker-ce

fi

# Verificar se o Docker Compose está instalado

if ! [ -x "$(command -v docker-compose)" ]; then

    bot\_warning "Docker Compose não está instalado. Instalando Docker Compose..."

    progress\_bar 3

    sudo curl -L "https://github.com/docker/compose/releases/download/v2.23.0/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose

    sudo chmod +x /usr/local/bin/docker-compose

    clear

fi

# Verificar se o Maven está instalado

if ! [ -x "$(command -v mvn)" ]; then

    bot\_warning "Maven não está instalado. Instalando Maven..."

    progress\_bar 3

    sudo apt-get install -y maven

    clear

fi

# Criar diretório para yml

mkdir -p dockerCompose

# Ir para o diretório

cd dockerCompose

# Clone GitHub repository

bot\_message "Clonando o repositório do GitHub..."

progress\_bar 5

git clone "https://github.com/ViaTechSP/java-viatech.git"

clear

# Construir o projeto usando Maven

bot\_message "Construindo o projeto Maven..."

progress\_bar 5

cd java-viatech

mvn clean install

clear

# Verificar se o JAR foi criado corretamente

if [ ! -f "target/java-viatech-1.0-SNAPSHOT-jar-with-dependencies.jar" ]; then

    bot\_error "Erro: O arquivo JAR não foi encontrado após a construção do Maven."

    exit 1

fi

# Voltar para o diretório dockerCompose

cd ..

# Create docker-compose.yml file

bot\_message "Criando o arquivo docker-compose.yml..."

progress\_bar 3

cat <<EOL > docker-compose.yml

version: '3.3'

services:

  mysql:

    container\_name: containerViatech

    image: mysql:5.7

    restart: always

    environment:

      MYSQL\_ROOT\_PASSWORD: urubu100

      MYSQL\_DATABASE: viatech

    volumes:

      - ./init-scripts:/docker-entrypoint-initdb.d

    ports:

      - "3306:3306"

EOL

# Create init-scripts directory e mover para init.sql file

bot\_message "Criando o diretório init-scripts e movendo o arquivo init.sql..."

progress\_bar 3

mkdir -p init-scripts

cp java-viatech/out/artifacts/java\_viatech\_jar/init.sql init-scripts/

# Start Docker

bot\_message "Iniciando o Docker ..."

sudo systemctl start docker

# Start Docker Compose

bot\_message "Iniciando o Docker Compose..."

progress\_bar 5

sudo docker-compose up -d

# Adicionar um sleep para dar tempo ao Docker Compose de iniciar os containers

bot\_message "Aguardando os containers iniciarem..."

progress\_bar 5

sleep 5

# Verificar se os containers estão em execução

if [ "$(sudo docker inspect -f '{{.State.Running}}' containerViatech)" == "true" ]; then

    bot\_success "Todos os containers estão em execução."

else

    bot\_error "O container MySQL não está em execução. Verifique os logs para mais detalhes."

    sudo docker-compose logs

    exit 1

fi

# Adicionar alias ao arquivo .bashrc

bot\_message "Adicionando alias para facilitar a execução do JAR..."

echo "alias executar='cd ~/dockerCompose/java-viatech/target && java -jar java-viatech-1.0-SNAPSHOT-jar-with-dependencies.jar'" >> ~/.bashrc

source ~/.bashrc

# Perguntar ao usuário se ele deseja executar o JAR

bot\_message "Deseja executar o arquivo JAR agora (S/N)?"

read exec\_jar

if [ "$exec\_jar" == "S" ]; then

    bot\_message "Executando o arquivo JAR..."

    cd java-viatech/target

    java -jar java-viatech-1.0-SNAPSHOT-jar-with-dependencies.jar

else

    bot\_warning "Você optou por não executar o JAR agora. Até a próxima!"

fi

bot\_success "Instalação concluída!"

# 

# GitHub

Todos os repositórios usados durante a execução do projeto:

https://github.com/ViaTechSP

# PPTX

Apresentação final PPTX:

[**https://www.canva.com/design/DAGD7cjHGqw/8cD7fRR\_17WXxUXJG\_5EbQ/edit**](https://www.canva.com/design/DAGD7cjHGqw/8cD7fRR_17WXxUXJG_5EbQ/edit)

Logotipo, Ícone

Descrição gerada automaticamente