Faculdade de Informática e Administração Paulista

FIAP Paulista

Henrique Martins Oliveira – RM563620
Henrique Texeira Cesar – RM563088

Symbio – Plataforma de Requalificação por IA

Global Solution – Java

São Paulo
2025

SUMÁRIO

OBJETIVO E ESCOPO DO PROJETO **2**

BREVE DESCRIÇÃO DAS FUNCIONALIDADES **3**

TABELAS ENDPOINTS **3**

PROTÓTIPO – PRINTS DAS TELAS **4**

MODELO DE ENTIDADE-RELACIONAMENTO **8**

DIAGRAMA DE CLASSSES **8**

OBJETIVO E ESCOPO DO PROJETO

Objetivo: A SYMBIO é uma plataforma B2B de inteligência corporativa focada em Mobilidade Interna e Requalificação. O objetivo deste componente de backend, desenvolvido em Java com Quarkus, é servir como o núcleo de processamento lógico, orquestrando a comunicação entre o banco de dados Oracle, o frontend e a API de Inteligência Artificial.

Escopo: O escopo desta API é fornecer endpoints RESTful seguros e performáticos para:

1. Gerenciar o catálogo de Cargos, Vagas, Skills e Colaboradores.
2. Integrar-se a um modelo de IA para analisar e salvar o risco de automação de cada cargo.
3. Executar a lógica de "Match" (Análise de Lacunas), calculando a compatibilidade percentual entre um colaborador e uma vaga interna.

BREVE DESCRIÇÃO DAS FUNCIONALIDADES

A solução Java foi totalmente implementada e hospedada na nuvem (Render), com as seguintes funcionalidades principais em operação:

- **Cadastro Inteligente de Cargos:** O endpoint POST /cargos não apenas salva um novo cargo, mas primeiramente consulta a API de IA em Python (https://symbio-api-ia.onrender.com/prever/risco) para calcular o risco de automação (Baixo, Médio, Alto). Esse risco é então persistido no banco de dados Oracle, enriquecendo o dado.
- **Algoritmo de Match (A Alma do Projeto):** O endpoint GET /match/{idColab}/{idVaga} executa a lógica de negócio principal. Ele compara as skills do colaborador (com seus níveis de proficiência 1-5) contra os requisitos da vaga (com seus pesos de importância 1-10), aplicando um cálculo ponderado.
- **API RESTful Completa:** Fornece todos os endpoints CRUD (Create, Read, Update, Delete) necessários para o frontend gerenciar Colaboradores, Vagas e Skills, seguindo as melhores práticas REST.
- **Conectividade com Oracle:** A camada DAO utiliza JDBC puro para comunicação direta com o banco de dados Oracle da FIAP, garantindo total controle sobre as transações.

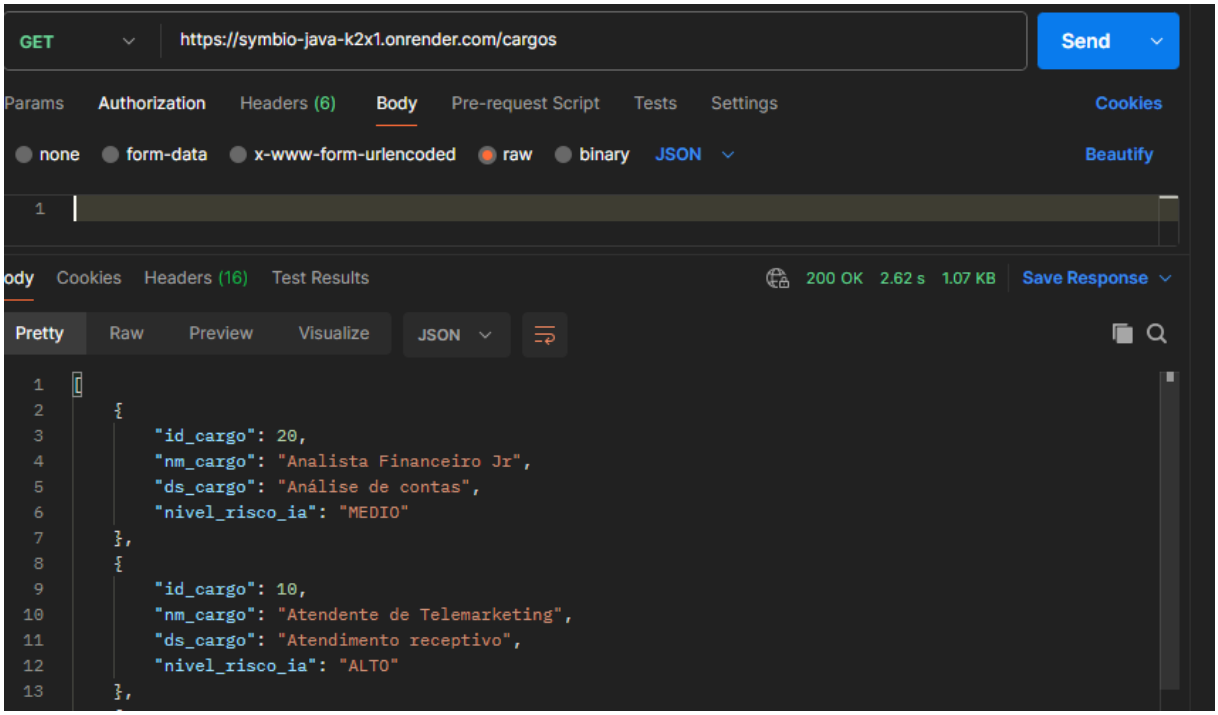
TABELAS ENDPOINTS

Verbo	URI (Caminho)	Descrição	Status (Sucesso)
GET	/hello	Endpoint de *Health Check* (teste).	200 OK
POST	/cargos	Cadastra novo cargo, consultando a IA.	201 Created
GET	/cargos	Lista todos os cargos.	200 OK
GET	/match/{idColab}/{idVaga}	Calcula o Match.	200 OK
GET	/colaboradores	Lista todos os colaboradores.	200 OK
GET	/colaboradores/{id}	Busca um colaborador por ID.	200 OK
POST	/colaboradores	Cadastra um novo colaborador.	201 Created

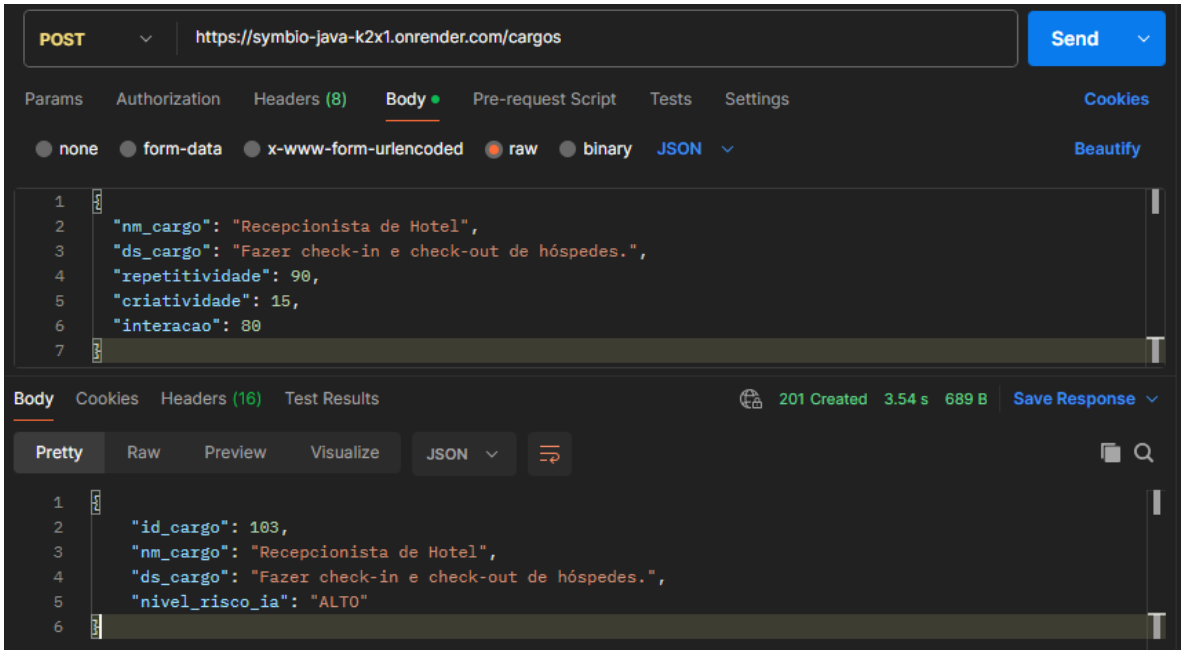
PUT	/colaboradores/{id}	Atualiza um colaborador.	200 OK
DELETE	/colaboradores/{id}	Deleta um colaborador.	204 No Content
GET	/vagas	Lista todas as vagas.	200 OK
POST	/vagas	Cadastra uma nova vaga.	201 Created
GET	/skills	Lista todas as skills.	200 OK
POST	/skills	Cadastra uma nova skill.	201 Created

PROTÓTIPO – PRINTS DAS TELAS

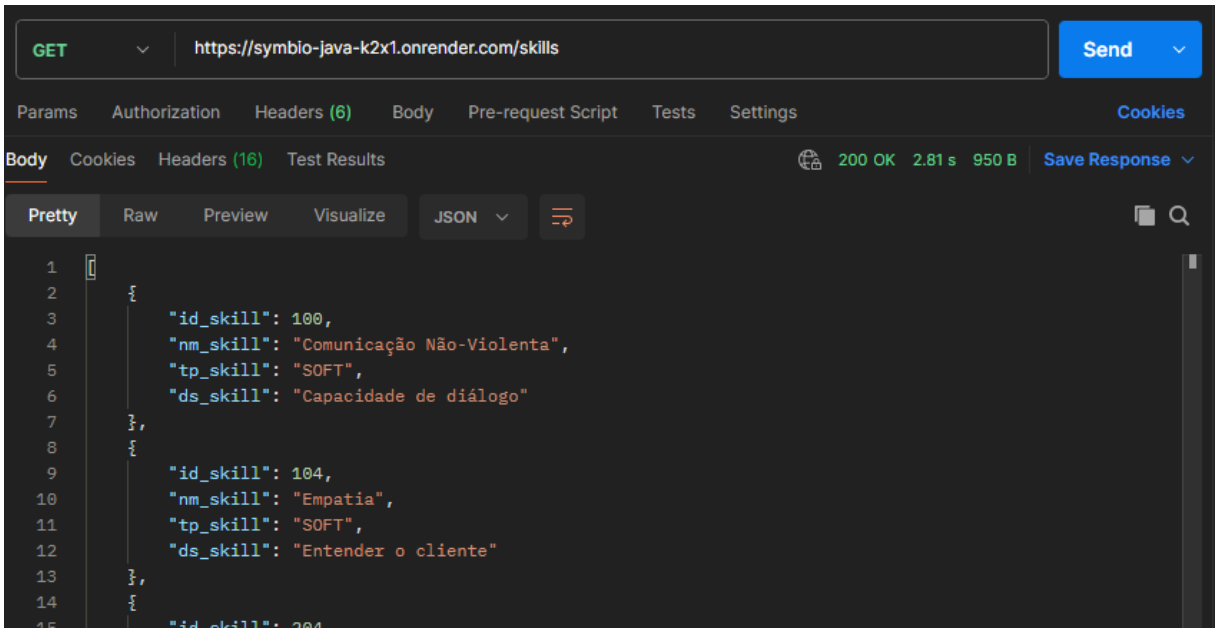
Listar Cargos - GET



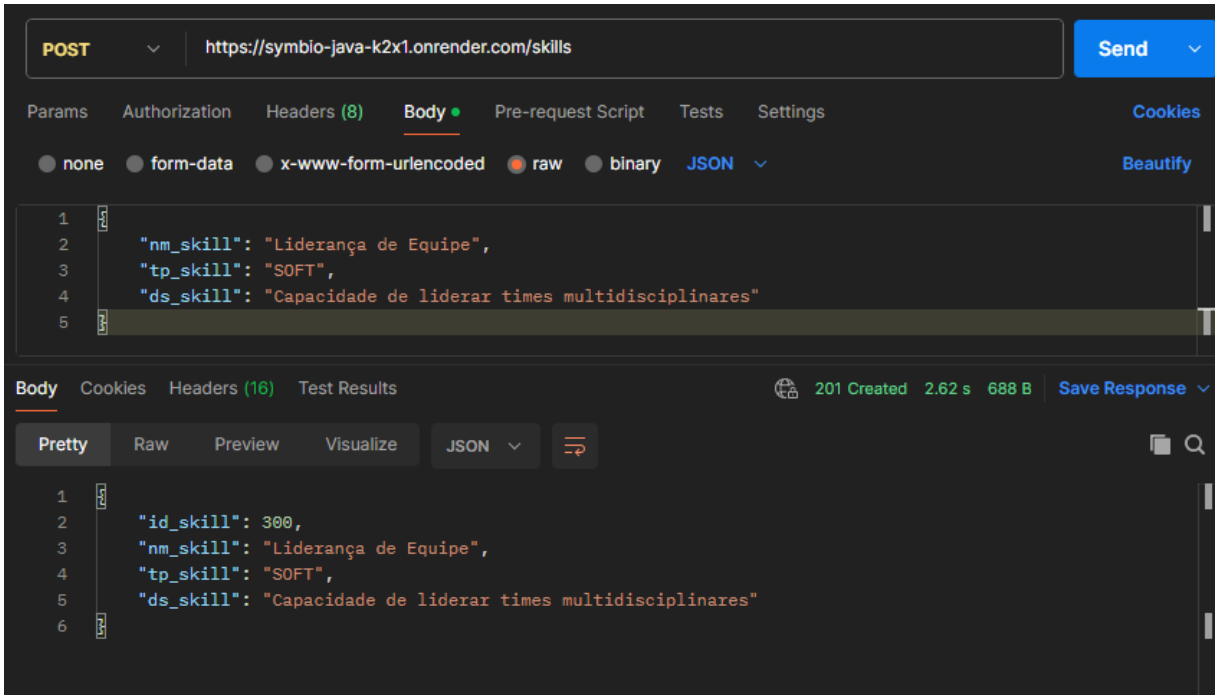
Cadastrar Cargo - POST



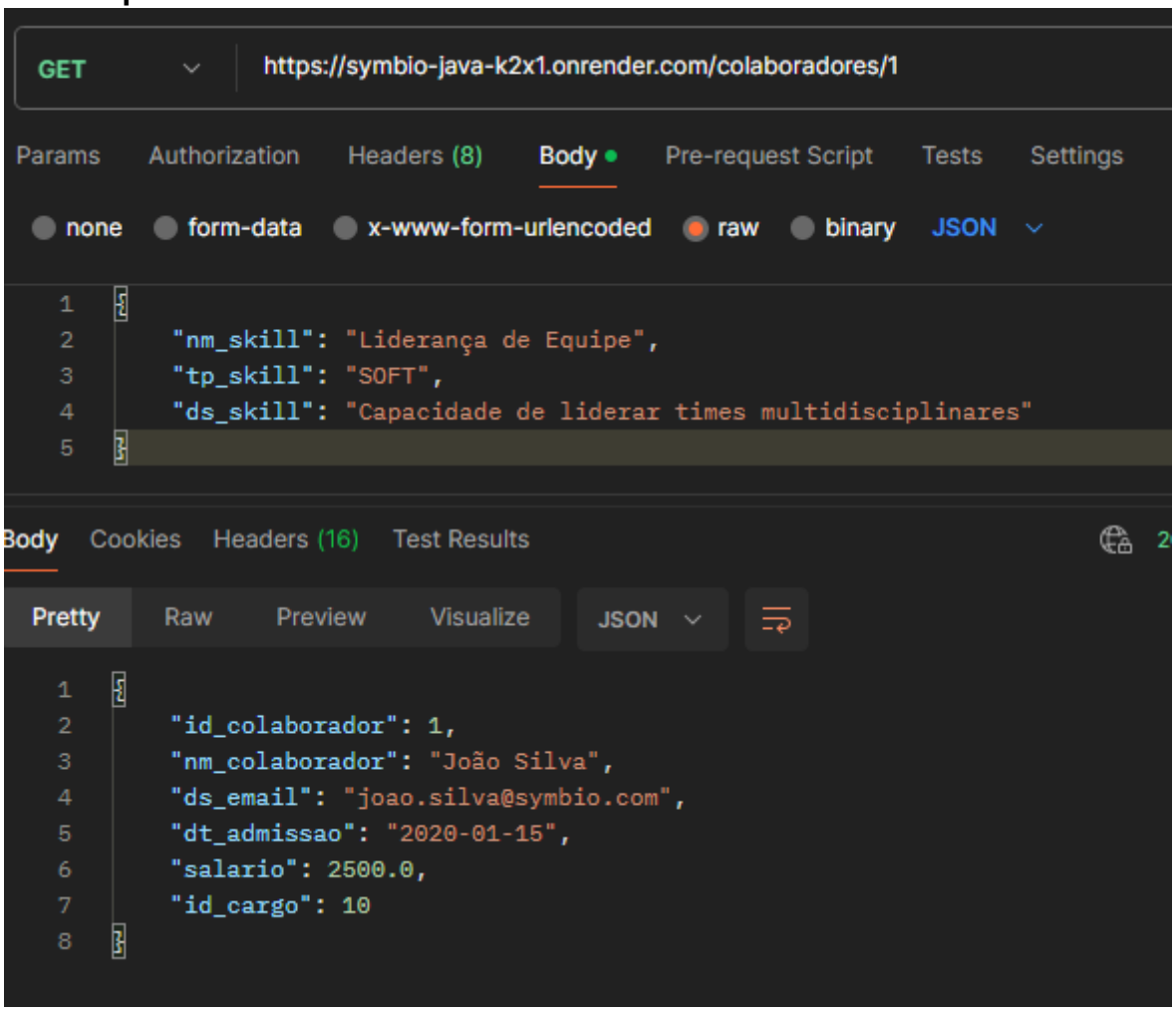
Listar Skills - GET



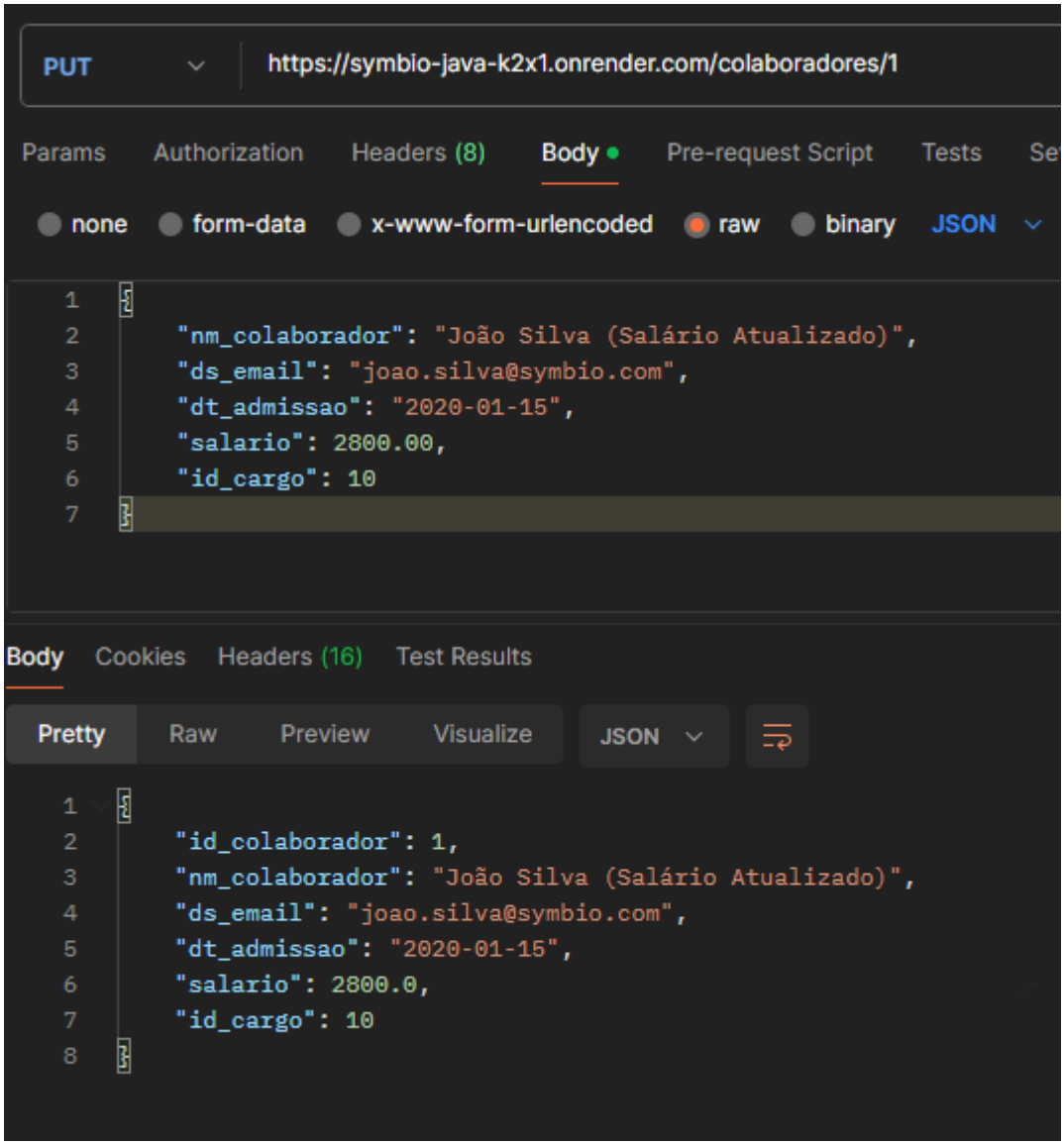
Cadastrar Skill – POST



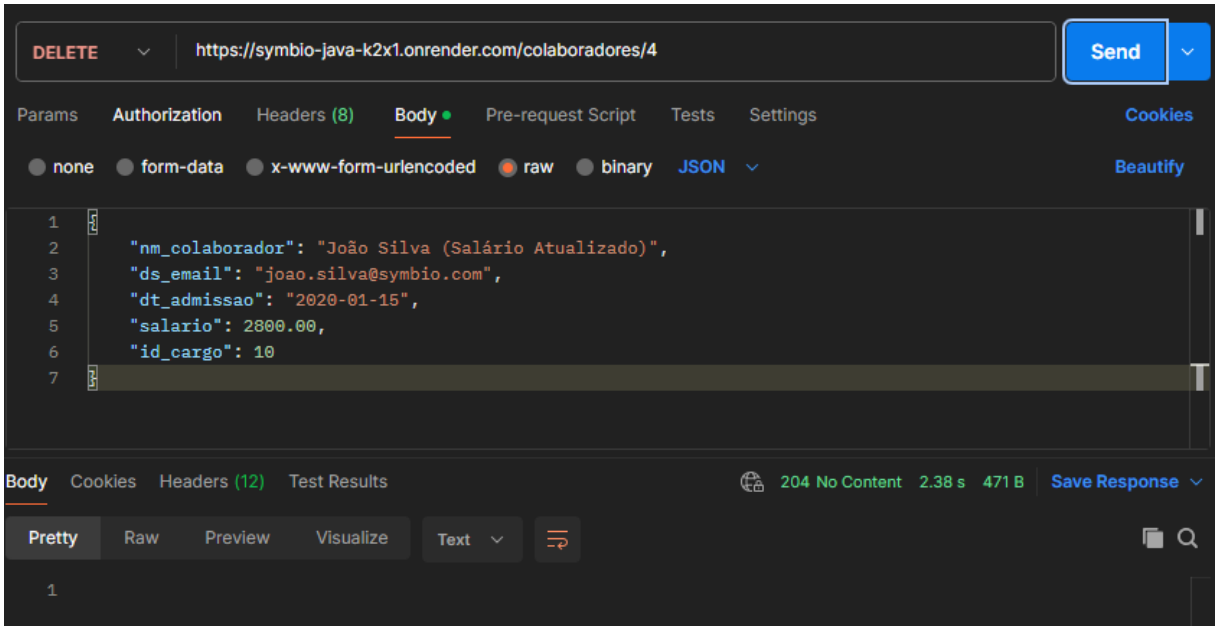
Buscar por ID



Atualizar Colaboradores – PUT



DELETE



MODELO DE ENTIDADE-RELACIONAMENTO

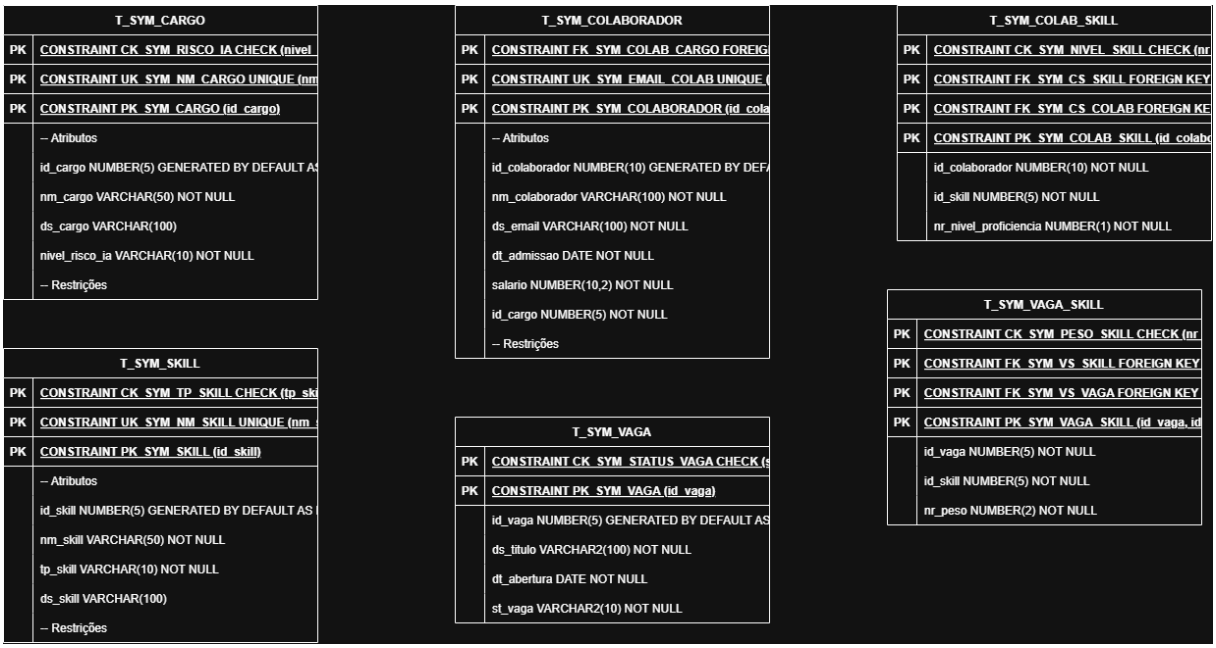
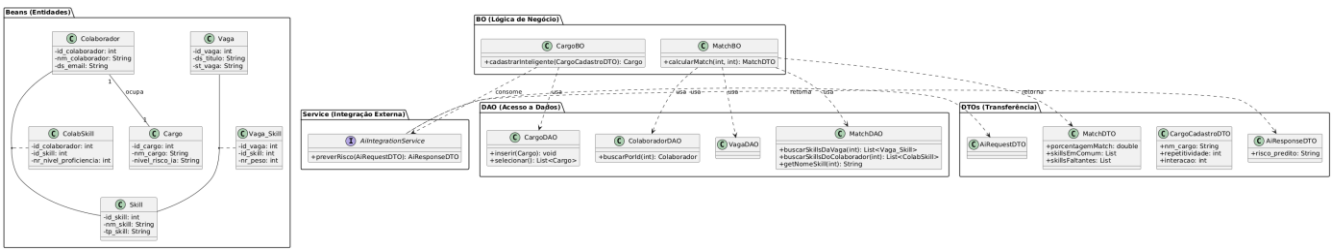


DIAGRAMA DE CLASSSES



COMO UTILIZAR

API Java (Quarkus): https://symbio-java-k2x1.onrender.com

API de IA (Python): https://symbio-api-ia.onrender.com

Vídeo de Demonstração: https://www.youtube.com/watch?v=dFaqvPYaIJc

Vídeo Pitch: https://youtu.be/mT2-jy_XQQ0

GITHUB: https://github.com/Symbio-Global-Solution/symbio-java.git

COLD START

Ambas as APIs "dormem" após 15 minutos sem uso.

A API Java (Passo 2) precisa chamar a API de IA (Passo 1). Se a API de IA estiver "dormindo", a API Java (que acorda mais rápido) não conseguirá obter o risco de IA a tempo e irá salvar o valor de fallback "ANALISE_PENDENTE" no banco de dados.

Para demonstrar o fluxo completo (onde a IA retorna "ALTO", "MEDIO" ou "BAIXO"), **você DEVE "aquecer" a API de IA (Python) PRIMEIRO.**

PASSO A PASSO PARA TESTAR

Use o Postman ou o cliente HTTP do IntelliJ.

Passo 1: "Acordar" a API de IA (Python)

Primeiro, envie uma requisição direta para a API de IA para forçá-la a "acordar".

Método: POST

URL: https://symbio-api-ia.onrender.com/prever/risco

Body (raw, JSON):

{ "features": [90, 10, 30] }

Ação: Clique em "Send" e **aguarde**. Esta primeira chamada pode demorar de **30 a 60 segundos**. Você saberá que funcionou quando receber a resposta {"risco_predito": "ALTO"}.

Passo 2: "Acordar" a API Java (Quarkus)

Agora que a IA está acordada, acorde o Java.

Método: GET

URL: https://symbio-java-k2x1.onrender.com/hello

Ação: Clique em "Send". Esta chamada também pode demorar alguns segundos.
Resultado: Você receberá o JSON {"status": "online", ...}.

Pronto! Ambas as APIs estão "quentes" e prontas para se comunicar.