**Passos para Construir e Executar com Docker**

1. **Navegue para o Diretório do Projeto:** Abra o terminal (prompt de comando) e vá até o diretório onde está localizado o seu Dockerfile. Execute o seguinte comando:

bash

 cd "C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1"

 **Verifique se o Dockerfile Está Presente:** Depois de entrar no diretório, verifique se o Dockerfile está lá. Execute:

bash

 dir

Você deve ver o Dockerfile listado na saída.

 **Construir a Imagem Docker:** Agora que você está no diretório correto, construa sua imagem Docker usando:

bash

 docker build -t sicode-sicode\_v1-web .

Certifique-se de que não há erros durante a construção.

 **Executar o Contêiner:** Após a imagem ter sido construída com sucesso, você pode executar o contêiner usando:

bash

1. docker run -it --rm -v "C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1:/app" sicode-sicode\_v1-web
2. Isso montará seu diretório local dentro do contêiner, permitindo que você veja as alterações em tempo real.

**Usando Docker Compose (se aplicável)**

Caso você também esteja utilizando o Docker Compose e tenha um arquivo docker-compose.yml, siga estes passos:

1. **Verifique sua configuração do Compose:** Certifique-se de que o arquivo docker-compose.yml está no diretório C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1.
2. **Executar Docker Compose:** Execute o seguinte comando para construir e iniciar os serviços definidos no arquivo docker-compose.yml:

bash

docker-compose up --build

C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1>docker build -t sicode-sicode\_v1-web . [+] Building 2.3s (18/18) FINISHED docker:desktop-linux => [internal] load build definition from Dockerfile 0.0s => => transferring dockerfile: 904B 0.0s => [internal] load metadata for docker.io/library/ruby:3.1.0 1.9s => [internal] load .dockerignore 0.0s => => transferring context: 745B 0.0s => [internal] load build context 0.1s => => transferring context: 31.72kB 0.1s => [ 1/13] FROM docker.io/library/ruby:3.1.0@sha256:249deb7f2b1a01f034141f529a2daeecdfd2c04aa1e2b456bf328d899779 0.0s => CACHED [ 2/13] RUN apt-get update -qq && apt-get install -y build-essential curl libpq-dev 0.0s => CACHED [ 3/13] RUN curl -fsSL https://deb.nodesource.com/setup\_18.x | bash - 0.0s => CACHED [ 4/13] RUN apt-get install -y nodejs 0.0s => CACHED [ 5/13] RUN npm install -g yarn 0.0s => CACHED [ 6/13] WORKDIR /app 0.0s => CACHED [ 7/13] COPY Gemfile /app/Gemfile 0.0s => CACHED [ 8/13] COPY Gemfile.lock /app/Gemfile.lock 0.0s => CACHED [ 9/13] COPY package.json /app/package.json 0.0s => CACHED [10/13] COPY yarn.lock /app/yarn.lock 0.0s => CACHED [11/13] RUN gem install bundler -v 2.5 0.0s => CACHED [12/13] RUN bundle install 0.0s => [13/13] COPY . ./app 0.1s => exporting to image 0.1s => => exporting layers 0.1s => => writing image sha256:dbd17e9e758f20d16c29051d3ff9132d3808905a83c9268db0fe501bddafd4d9 0.0s => => naming to docker.io/library/sicode-sicode\_v1-web 0.0s View build details: docker-desktop://dashboard/build/desktop-linux/desktop-linux/sowogyt9xakgbv6ueafr9sgz1 2 warnings found (use docker --debug to expand): - LegacyKeyValueFormat: "ENV key=value" should be used instead of legacy "ENV key value" format (line 6) - JSONArgsRecommended: JSON arguments recommended for CMD to prevent unintended behavior related to OS signals (line 37) What's next: View a summary of image vulnerabilities and recommendations → docker scout quickview C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1>docker run -it --rm -v "C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1:/app" sicode-sicode\_v1-web /bin/sh: 1: [./bin/rails,: not found

ChatGPT

Parece que ocorreu um erro ao tentar executar o contêiner, relacionado ao comando especificado da imagem Docker. A mensagem /bin/sh: 1: [./bin/rails,: not found sugere que o comando que você está tentando executar pode estar incorreto ou formatado de maneira errada.

### Passos para Resolver o Problema

1. **Verifique o CMD no Dockerfile:** Abra seu Dockerfile e verifique a última linha (que geralmente contém o comando CMD). Certifique-se de que está definido corretamente. Por exemplo:

Se você deve iniciar uma aplicação Ruby on Rails, o comando pode ser algo como:

Dockerfile

CMD ["./bin/rails", "server", "-b", "0.0.0.0"]

Ou talvez você precise iniciar um arquivo específico diretamente:

Dockerfile

 CMD ["ruby", "seu\_arquivo\_principal.rb"]

Assegure-se de que não há vírgulas desnecessárias ou espaços extras no comando.

 **Utilize Comando Interativo para Debug:** Para diagnosticar mais a fundo, você pode executar o contêiner em um shell interativo para verificar os arquivos e executar comandos manualmente:

bash

docker run -it --rm -v "C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1:/app" sicode-sicode\_v1-web /bin/bash

Isso abrirá um terminal Bash dentro do contêiner. A partir daí, você pode verificar se os arquivos estão presentes na pasta /app:

bash

ls /app

E tente executar o comando manualmente, como:

bash

1. ./bin/rails server -b 0.0.0.0
2. (ou qualquer outro comando que você estava tentando executar).
3. **Corrija Qualquer Erro no Path ou Diretório:** Se o comando estiver correto, mas ainda não funcionar, verifique se o diretório e os arquivos que você está tentando acessar estão presentes e têm as permissões corretas.

### Exemplo de um Dockerfile para uma Aplicação Rails

Se você está configurando uma aplicação Rails, um exemplo típico de Dockerfile pode parecer assim:

Dockerfile

FROM ruby:3.1.0

# Instala dependências

RUN apt-get update -qq && apt-get install -y build-essential curl libpq-dev nodejs

# Configura o diretório de trabalho

WORKDIR /app

# Adiciona os arquivos de configuração (Gemfile e Gemfile.lock)

COPY Gemfile Gemfile.lock ./

# Instala as gems

RUN gem install bundler && bundle install

# Copia o restante dos arquivos do projeto

COPY . .

# Expõe a porta

EXPOSE 3000

# Comando para iniciar a aplicação

CMD ["./bin/rails", "server", "-b", "0.0.0.0"]

### Resumo

1. Verifique e corrija o comando no CMD do Dockerfile.
2. Utilize um shell interativo para diagnosticar problemas.
3. Assegure-se de que todos os arquivos necessários estejam presentes.

Depois de fazer essas correções, tente rodar o contêiner novamente e veja se o problema persiste. Se ainda houver erros, compartilhe suas mensagens de erro e vou ajudar a resolver!

Diretoria Financeira

C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1>docker run -it --rm -v "C:\Users\mferreirasantos\Documents\sicode-SICODE\_v1:/app" sicode-sicode\_v1-web /bin/bash root@20d885e64def:/app# ls /app Dockerfile Procfile.dev app config.ru docker-compose.yml node\_modules spec tmp Gemfile README.md bin db lib package.json storage vendor Gemfile.lock Rakefile config doc log public test yarn.lock root@20d885e64def:/app# ./bin/rails server -b 0.0.0.0 => Booting Puma => Rails 7.1.3 application starting in development => Run `bin/rails server --help` for more startup options Puma starting in single mode... \* Puma version: 6.4.2 (ruby 3.1.0-p0) ("The Eagle of Durango") \* Min threads: 5 \* Max threads: 5 \* Environment: development \* PID: 8 \* Listening on http://0.0.0.0:3000 Use Ctrl-C to stop

ChatGPT

Ó! Parece que agora a sua aplicação Ruby on Rails está sendo executada corretamente dentro do contêiner Docker. O servidor Puma está rodando na porta 3000 e está acessível em http://0.0.0.0:3000.

### Acessando a Aplicação

Para acessar a aplicação que você está rodando dentro do contêiner, siga os passos abaixo:

1. **Obtenha o IP do Contêiner (se necessário):** Se você estiver executando o Docker no Windows e não conseguir acessar a aplicação diretamente no navegador via http://localhost:3000, você pode precisar obter o IP do contêiner. Use o seguinte comando para descobrir o endereço IP:

bash

1. docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' <container\_id>
2. Substitua <container\_id> pelo ID do contêiner que você obteve ao iniciar o contêiner. O comando docker ps pode ajudá-lo a encontrar esse ID.
3. **Acesso via Navegador:** Acesse sua aplicação no navegador através de:
   * http://localhost:3000 (tente primeiro essa opção)
   * Ou utilizando o IP encontrado no comando acima, se necessário.

### Executando a Aplicação de Forma Persistente

Se você deseja executar a aplicação de forma mais simples (em vez de ter que entrar em um terminal interativo toda vez), você pode ajustar o CMD ou ENTRYPOINT no seu Dockerfile ou usar o docker-compose.

### Usando Docker Compose

**Se você estiver usando docker-compose, você pode iniciar a aplicação com o seguinte comando:**

Navegue para o diretório dos seus arquivos de configuração e execute:

bash

docker-compose up

Certifique-se de que seu docker-compose.yml está bem configurado. Um exemplo básico poderia ser algo assim:

yaml

version: '3'

services:

web:

build: .

volumes:

- .:/app

ports:

- "3000:3000"

command: ./bin/rails server -b 0.0.0.0

### Resumo

* Sua aplicação agora está funcionando dentro do contêiner Docker, acessível na porta 3000.
* Você pode acessar a aplicação através de http://localhost:3000 ou pelo IP do contêiner.
* Para uma execução mais suave, considere usar o Docker Compose.

Se tiver mais dúvidas ou precisar de mais assistência, sinta-se à vontade para perguntar!

Diretoria Financeira

o id do meu container é 20d885e64def

ChatGPT

Com o ID do seu contêiner 20d885e64def, você pode realizar algumas ações adicionais e obter informações úteis. Vou listar algumas opções que podem ajudá-lo a trabalhar com o contêiner e a sua aplicação:

### 1. ****Acessar o IP do Contêiner****

Se você precisar acessar o IP do contêiner diretamente, execute o seguinte comando no seu terminal:

bash

docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' 20d885e64def

Esse comando retornará o endereço IP que você pode usar para acessar a aplicação, caso não consiga acessá-la via http://localhost:3000.

### 2. ****Verificar logs do Contêiner****

Caso você precise ver quais logs estão sendo gerados pela sua aplicação enquanto o contêiner está em execução, você pode usar este comando:

bash

docker logs 20d885e64def

Esse comando exibirá a saída de log do seu contêiner, onde você poderá ver mensagens de depuração, erros e outros avisos importantes.

### 3. ****Executar Comandos no Contêiner****

Se você precisar executar comandos adicionais dentro do contêiner (por exemplo, executar migrações do banco de dados ou executar testes), você pode fazer isso com um shell interativo:

bash

docker exec -it 20d885e64def /bin/bash

Com esse comando, você terá acesso ao shell do contêiner e poderá navegar, executar scripts, e realizar outras tarefas diretamente.

### 4. ****Parar o Contêiner****

Se você quiser parar o contêiner a partir do terminal, pode usar:

bash

docker stop 20d885e64def

### 5. ****Remover o Contêiner****

Caso você precise remover o contêiner após o trabalho ou para fazer uma nova construção, você pode usar:

bash

docker rm 20d885e64def

### Acessando sua Aplicação

Lembre-se, se a sua aplicação já estiver rodando na porta 3000, você deve ser capaz de acessá-la no navegador via http://localhost:3000. Se você encontrar problemas para acessar, tente:

* Certifique-se de que o Docker está configurado corretamente.
* Usar o IP que você obteve no primeiro passo.

Se você continuar a encontrar problemas ou tiver mais perguntas, fique à vontade para perguntar!

Diretoria Financeira

C:\Users\mferreirasantos>docker inspect -f '{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}' 20d885e64def template parsing error: template: :1: unclosed action C:\Users\mferreirasantos>docker logs 20d885e64def root@20d885e64def:/app# ls /app Dockerfile Procfile.dev app config.ru docker-compose.yml node\_modules spec tmp Gemfile README.md bin db lib package.json storage vendor Gemfile.lock Rakefile config doc log public test yarn.lock root@20d885e64def:/app# ./bin/rails server -b 0.0.0.0 => Booting Puma => Rails 7.1.3 application starting in development => Run `bin/rails server --help` for more startup options Puma starting in single mode... \* Puma version: 6.4.2 (ruby 3.1.0-p0) ("The Eagle of Durango") \* Min threads: 5 \* Max threads: 5 \* Environment: development \* PID: 8 \* Listening on http://0.0.0.0:3000 Use Ctrl-C to stop C:\Users\mferreirasantos>docker exec -it 20d885e64def /bin/bash root@20d885e64def:/app#

ChatGPT

Parece que você está progredindo bem com a utilização do Docker e sua aplicação Ruby on Rails. Vamos resolver o erro que você encontrou ao tentar obter o IP do contêiner e, em seguida, verificaremos algumas outras informações que você pode precisar.

### Corrigindo o Comando para Obter o IP do Contêiner

O erro template parsing error: template: :1: unclosed action indica que a sintaxe do comando docker inspect estava incorreta, provavelmente devido a problemas com o uso de aspas simples e as chaves. No Windows, você pode precisar usar aspas duplas em volta de toda a expressão:

bash

docker inspect -f "{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}" 20d885e64def

Esse comando deve retornar o IP do contêiner. Caso não retorne nada, mas você sabe que o contêiner está em execução, você pode ignorar essa parte e continuar acessando a aplicação através de http://localhost:3000.

### Confirmando o Acesso à Aplicação

Agora que você já está com o servidor Puma rodando e ouvindo em http://0.0.0.0:3000, você deve conseguir acessar sua aplicação através do navegador em:

* **URL**: http://localhost:3000

Se não funcionar, e se você obteve o IP do contêiner, experimente acessar o IP diretamente.

### Utilizando o Terminal Interativo do Contêiner

Dado que você já conseguiu executar o comando docker exec -it 20d885e64def /bin/bash, você está dentro do contêiner e pode fazer várias ações:

* **Comandos Rails**: Você pode executar comandos Rails diretamente, como rodar migrações:

bash

 ./bin/rails db:migrate

 **Executar Testes**: Caso você tenha testes definidos, pode também executá-los:

bash

* ./bin/rails test
* **Verificar Dependências**: Se houver necessidade de verificar dependências ou algo mais, você pode usar comandos gem, npm, ou yarn dependendo de qual parte você precisa.

### Próximos Passos

1. **Acesse sua aplicação**: Tente acessar no navegador.
2. **Execute mais comandos**: Use o terminal do contêiner para gerenciar e testar sua aplicação.
3. **Se tiver mais dúvidas**: Sinta-se à vontade para perguntar se precisar de mais assistência!

Se você encontrar erros específicos ou precisar de ajuda com alguma outra coisa, por favor, compartilhe e estarei aqui para ajudar!

Diretoria Financeira

C:\Users\mferreirasantos>docker inspect -f "{{range .NetworkSettings.Networks}}{{.IPAddress}}{{end}}" 20d885e64def 172.17.0.2 C:\Users\mferreirasantos> C:\Users\mferreirasantos>docker exec -it 20d885e64def /bin/bash root@20d885e64def:/app# ./bin/rails db:migrate bin/rails aborted! ActiveRecord::ConnectionNotEstablished: could not connect to server: No such file or directory Is the server running locally and accepting connections on Unix domain socket "/var/run/postgresql/.s.PGSQL.5432"? Caused by: PG::ConnectionBad: could not connect to server: No such file or directory Is the server running locally and accepting connections on Unix domain socket "/var/run/postgresql/.s.PGSQL.5432"? Tasks: TOP => db:migrate (See full trace by running task with --trace) root@20d885e64def:/app#

ChatGPT

A mensagem de erro que você está recebendo ao tentar realizar a migração do banco de dados (ActiveRecord::ConnectionNotEstablished) indica que sua aplicação Ruby on Rails não está conseguindo se conectar ao banco de dados PostgreSQL.

### Passos para Solucionar o Problema

1. **Verifique a Configuração do Banco de Dados**: A primeira coisa que você deve fazer é verificar suas configurações de banco de dados no arquivo config/database.yml. Esse arquivo deve estar configurado corretamente para se conectar ao banco de dados PostgreSQL. Certifique-se de que ele contenha informações corretas, como nome do banco de dados, usuário e senha.

Um exemplo básico de uma configuração database.yml para PostgreSQL pode ser:

yaml

 default: &default

adapter: postgresql

encoding: unicode

pool: 5

username: <seu\_usuario>

password: <sua\_senha>

host: db

development:

<<: \*default

database: nome\_do\_banco\_de\_dados\_development

test:

<<: \*default

database: nome\_do\_banco\_de\_dados\_test

production:

<<: \*default

database: nome\_do\_banco\_de\_dados\_production

username: <seu\_usuario>

password: <sua\_senha>

**Substitua <seu\_usuario>, <sua\_senha> e nome\_do\_banco\_de\_dados\_development pelos valores corretos.**

 **Execução do PostgreSQL**: Certifique-se de que o PostgreSQL está em execução. Se você está utilizando o Docker para rodar o PostgreSQL, verifique se há um contêiner de banco de dados ativo. Você pode usar:

bash

docker ps

Se você não possui um contêiner de banco de dados em execução, você precisará iniciar um. Um exemplo de configuração usando Docker Compose para um banco de dados PostgreSQL pode ser o seguinte:

yaml

 version: '3'

services:

db:

image: postgres:latest

environment:

POSTGRES\_DB: nome\_do\_banco\_de\_dados\_development

POSTGRES\_USER: seu\_usuario

POSTGRES\_PASSWORD: sua\_senha

volumes:

- postgres\_data:/var/lib/postgresql/data

web:

build: .

volumes:

- .:/app

ports:

- "3000:3000"

depends\_on:

- db

command: ./bin/rails server -b 0.0.0.0

volumes:

postgres\_data:

**Substitua nome\_do\_banco\_de\_dados\_development, seu\_usuario, e sua\_senha pelos valores desejados.**

 **Iniciar o Banco de Dados com Docker Compose**: Se você tiver o docker-compose.yml configurado como mostrado acima, execute o comando:

bash

 docker-compose up

Isso irá iniciar tanto o servidor Rails quanto o PostgreSQL.

 **Realizar a Migração Novamente**: Depois que o banco de dados estiver em execução, você pode entrar no contêiner novamente ou, se estiver usando docker-compose, você pode fazer a migração diretamente:

bash

1. docker-compose exec web ./bin/rails db:migrate

### Resumo

1. Verifique e ajuste config/database.yml.
2. Certifique-se de que o contêiner do PostgreSQL está em execução.
3. Inicie o ambiente com docker-compose se aplicável.
4. Execute as migrações.

Se você continuar a encontrar problemas ou outra mensagem de erro, compartilhe-os, e ficarei feliz em ajudar!

Diretoria Financeira

Segue como está meu arquivo 'database.yml': default: &default adapter: postgresql encoding: utf8 pool: <%= ENV.fetch("RAILS\_MAX\_THREADS") { 5 } %> homologacao: <<: \*default host: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_HOST"] %> database: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_DATABASE"] %> user: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_USER"] %> password: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PASSWORD"] %> port: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PORT"] %> schema\_search\_path: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_SCHEMA"] %> development: <<: \*default host: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_HOST"] %> database: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_DATABASE"] %> user: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_USER"] %> password: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PASSWORD"] %> port: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PORT"] %> schema\_search\_path: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_SCHEMA"] %> test: <<: \*default host: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_HOST"] %> database: app\_test user: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_USER"] %> password: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PASSWORD"] %> port: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PORT"] %> schema\_search\_path: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_SCHEMA"] %> production: <<: \*default host: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_HOST"] %> database: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_DATABASE"] %> user: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_USER"] %> password: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PASSWORD"] %> port: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_PORT"] %> schema\_search\_path: <%= ENV["OPENSHIFT\_POSTGRESQL\_DB\_SCHEMA"] %>

ChatGPT

A configuração do seu arquivo database.yml parece estar bem estruturada, no entanto, está utilizando variáveis de ambiente para definições como o host, usuário, senha e porta do banco de dados PostgreSQL. Esse método é comumente usado em um ambiente onde as variáveis de ambiente são fornecidas por um serviço de hospedagem, como o Openshift.

Como você está desenvolvendo localmente, algumas coisas precisam ser ajustadas para permitir a conexão com o banco de dados PostgreSQL em um ambiente de desenvolvimento. Vamos corrigir isso.

### Configurando o database.yml para Desenvolvimento Local

Para que sua aplicação Ruby on Rails se conecte a um banco de dados PostgreSQL local, você precisará alterar a seção development do seu database.yml para algo mais específico, como abaixo:

yaml

default: &default

adapter: postgresql

encoding: utf8

pool: <%= ENV.fetch("RAILS\_MAX\_THREADS") { 5 } %>

development:

<<: \*default

host: localhost # Para desenvolvimento local

database: nome\_do\_banco\_de\_dados\_development # Nomeie o banco de dados como desejar

user: postgres # Geralmente o usuário padrão do PostgreSQL

password: sua\_senha # Substitua pela senha do seu usuário PostgreSQL

port: 5432 # A porta padrão do PostgreSQL

Certifique-se de substituir nome\_do\_banco\_de\_dados\_development pelo nome que você quer dar ao seu banco de dados para o ambiente de desenvolvimento e sua\_senha pela senha correta do usuário do PostgreSQL.

### Passos para Criar o Banco de Dados Localmente

1. **Instalar PostgreSQL** (se você ainda não tiver instalado): Certifique-se de que o PostgreSQL está instalado e executando em sua máquina local.
2. **Criar o Banco de Dados**: Abra o terminal e faça login no PostgreSQL com o usuário que você configurou. Por padrão, o usuário é postgres:

bash

psql -U postgres

Se solicitado, insira a senha. Uma vez logado no PostgreSQL, crie o banco de dados:

sql

 CREATE DATABASE nome\_do\_banco\_de\_dados\_development;

Substitua nome\_do\_banco\_de\_dados\_development pelo nome que você usou na configuração do database.yml.

 **Saia do PostgreSQL**: Após criar o banco de dados, você pode sair do cliente psql digitando:

sql

 \q

 **Rodar as Migrações**: Agora que você configurou o banco de dados, retorne ao seu contêiner Docker, utilizando:

bash

docker exec -it 20d885e64def /bin/bash

E execute a migração:

bash

1. ./bin/rails db:migrate

### Usando Variáveis de Ambiente

Se você preferir ainda usar variáveis de ambiente, você pode definir essas variáveis no seu ambiente local (por exemplo, em seu .bashrc, .bash\_profile, ou outro arquivo de configuração da sua shell) ou usar um arquivo .env com a gem dotenv-rails para carregar as variáveis necessárias.

Um exemplo de .env poderia ser:

ini

OPENSHIFT\_POSTGRESQL\_DB\_HOST=localhost

OPENSHIFT\_POSTGRESQL\_DB\_DATABASE=nome\_do\_banco\_de\_dados\_development

OPENSHIFT\_POSTGRESQL\_DB\_USER=postgres

OPENSHIFT\_POSTGRESQL\_DB\_PASSWORD=sua\_senha

OPENSHIFT\_POSTGRESQL\_DB\_PORT=5432

E assim, na configuração do seu database.yml, você não precisaria mudar nada, pois ele irá buscar essas variáveis automaticamente.

### Resumo

* Atualize o database.yml para se conectar ao PostgreSQL localmente.
* Crie o banco de dados no PostgreSQL se ainda não estiver criado.
* Execute as migrações e confirme o funcionamento da aplicação.

Se você tiver mais perguntas ou se algo não estiver claro, sinta-se à vontade para perguntar!

Diretoria Financeira

ao rodar 'psql -U postgres', deu o erro: 'psql' não é reconhecido como um comando interno ou externo, um programa operável ou um arquivo em lotes.

ChatGPT

A mensagem de erro que você recebeu — 'psql' não é reconhecido como um comando interno ou externo, um programa operável ou um arquivo em lotes — indica que o cliente de linha de comando do PostgreSQL (psql) não está instalado ou não está acessível no seu PATH do sistema.

### Passos para Solucionar

Aqui estão algumas etapas que você pode seguir para resolver esse problema:

#### 1. **Verifique se o PostgreSQL Está Instalado**

Primeiro, você precisa garantir que o PostgreSQL esteja realmente instalado em sua máquina. Você pode fazer isso acessando o Painel de Controle em **Programas e Recursos** e verificando se o PostgreSQL está listado.

### 2. ****Instalar o PostgreSQL****

Se o PostgreSQL não estiver instalado, você pode baixá-lo e instalá-lo:

1. **Baixe o Instalador**:
   * Acesse o site oficial do [PostgreSQL](https://www.postgresql.org/download/) e baixe o instalador apropriado para o seu sistema operacional (Windows).
2. **Instalação**:
   * Execute o instalador e siga as instruções na tela. Durante a instalação, você terá a opção de instalar o cliente psql. Certifique-se de selecionar essa opção.
3. **Adicionar ao PATH** (Caso não tenha feito isso automaticamente):
   * Durante a instalação ou posteriormente, adicione o diretório onde o psql está instalado ao PATH do sistema. O caminho típico é semelhante a:

text

* + C:\Program Files\PostgreSQL\<version>\bin
  + Para adicionar ao PATH:
    - Clique com o botão direito em "Este PC" ou "Meu Computador" e escolha "Propriedades".
    - Vá para "Configurações Avançadas do Sistema".
    - Clique em "Variáveis de Ambiente".
    - Em "Variáveis do sistema", encontre e selecione a variável "Path", e clique em "Editar".
    - Adicione o caminho do diretório bin do PostgreSQL.

1. **Reinicie o Prompt de Comando**:
   * Após adicionar o caminho ao PATH, feche o terminal e abra-o novamente para aplicar as mudanças.

#### 3. **Verificar a Instalação**

Após a instalação e a configuração do PATH, você pode testar se o psql está acessível digitando o seguinte comando em um novo prompt de comando:

bash

psql -U postgres

Se solicitado, insira sua senha. Isso deve abrir o cliente de linha de comando do PostgreSQL.

### 4. ****Criar o Banco de Dados****

Depois de conseguir acessar o psql, você pode proceder com os comandos para criar seu banco de dados usando o seguinte comando dentro do cliente psql:

sql

CREATE DATABASE nome\_do\_banco\_de\_dados\_development;

### Resumo

1. Verifique e, se necessário, instale o PostgreSQL.
2. Adicione o caminho do cliente psql ao PATH do seu sistema.
3. Verifique o acesso ao psql no terminal.

Se você ainda tiver dificuldades, não hesite em perguntar!