

## **Documentação para executar os microserviços – Prodigyan**

### **Passos para a implementação dos containers na AWS**

- Criar um banco de dados seguindo as orientações do documento dicionário do banco de dados na gcp.
- Criar os contêineres docker na seguinte ordem abaixo, sempre informando as variáveis de ambiente esperadas, para a conexão com o banco de dados seja feita corretamente.
- Por último, o container contendo o Nginx, deve conter o IP do container na AWS e sua respectiva porta.

### **Containers**

**Exemplo ao final do documento**

#### **Container tipos**

**Imagem:** brunonascimentto/tipos:1.0

**Porta do container:** 3001

**Protocol:** http

#### **Variáveis de ambientes esperadas:**

- DB\_HOST = {host do banco de dados}
- DB\_PORT = {porta do banco de dados}
- DB\_USERNAME = {usuário do banco de dados}
- DB\_PASSWORD = {senha do banco de dados}
- DB\_DATABASE = {nome do banco que será acessado}

#### **Linhas de comando para executar o container:**

- docker pull brunonascimentto/tipos:1.0
- docker run -p 3001:3001 -d -e DB\_HOST={ host do banco de dados } -e DB\_PORT={ porta do banco de dados } -e DB\_USERNAME={ usuário do banco de dados } -e DB\_PASSWORD={ senha do banco de dados } -e DB\_DATABASE={ nome do banco que será acessado } --name tipos brunonascimentto/tipos:1.0

**Chamada para api deste container:**

<http://{ip público do container}:3001/cliente/tipos>

**Obs:** substitua todo o {} pelo valor da variável

**Container gravar**

**Imagem:** brunonascimentto/gravar:1.0

**Porta do container:** 3002

**Protocol:** http

**Variáveis de ambientes esperadas:**

- DB\_HOST = {host do banco de dados}
- DB\_PORT = {porta do banco de dados}
- DB\_USERNAME = {usuário do banco de dados}
- DB\_PASSWORD = {senha do banco de dados}
- DB\_DATABASE = {nome do banco que será acessado}

**Linhas de comando para executar o container:**

- docker pull brunonascimentto/gravar:1.0
- docker run -p 3002:3002 -d -e DB\_HOST={ host do banco de dados } -e DB\_PORT={ porta do banco de dados } -e DB\_USERNAME={ usuário do banco de dados } -e DB\_PASSWORD={ senha do banco de dados } -e DB\_DATABASE={ nome do banco que será acessado } --name gravar brunonascimentto/gravar:1.0

**Chamada para api deste container:**

<http://{ip público do container}:3002/cliente/gravar>

**Obs:** substitua todo o {} pelo valor da variável

## Container buscar-id

**Imagem:** brunonascimentto/buscar-id:1.0

**Porta do container:** 3003

**Protocol:** http

**Variáveis de ambientes esperadas:**

- DB\_HOST = {host do banco de dados}
- DB\_PORT = {porta do banco de dados}
- DB\_USERNAME = {usuário do banco de dados}
- DB\_PASSWORD = {senha do banco de dados}
- DB\_DATABASE = {nome do banco que será acessado}

**Obs:** substitua todo o {} pelo valor da variável

**Linhas de comando para executar o container:**

- docker pull brunonascimentto/buscar-id:1.0
- docker run -p 3003:3003 -d -e DB\_HOST={ host do banco de dados } -e DB\_PORT={ porta do banco de dados } -e DB\_USERNAME={ usuário do banco de dados } -e DB\_PASSWORD={ senha do banco de dados } -e DB\_DATABASE={ nome do banco que será acessado } --name buscar-id brunonascimentto/buscar-id:1.0

**Chamada para api deste container:**

<http://{ip público do container}:3003/cliente/buscar/id/{id do cliente}>

## Container buscar-email

**Imagem:** brunonascimentto/buscar-email:1.0

**Porta do container:** 3004

**Protocol:** http

**Variáveis de ambientes esperadas:**

- DB\_HOST = {host do banco de dados}
- DB\_PORT = {porta do banco de dados}
- DB\_USERNAME = {usuário do banco de dados}
- DB\_PASSWORD = {senha do banco de dados}
- DB\_DATABASE = {nome do banco que será acessado}

**Obs:** substitua todo o {} pelo valor da variável

**Linhas de comando para executar o container:**

- docker pull brunonascimentto/buscar-email:1.0
- docker run -p 3004:3004 -d -e DB\_HOST={ host do banco de dados } -e DB\_PORT={ porta do banco de dados } -e DB\_USERNAME={ usuário do banco de dados } -e DB\_PASSWORD={ senha do banco de dados } -e DB\_DATABASE={ nome do banco que será acessado } --name buscar-email brunonascimentto/buscar-email:1.0

**Chamada para api deste container:**

http://{ip público do container}:3004/cliente/buscar/email/{email do cliente}

## **Container todos**

**Imagem:** brunonascimentto/todos:1.0

**Porta do container:** 3005

**Protocolo:** http

**Variáveis de ambientes esperadas:**

- DB\_HOST = {host do banco de dados}
- DB\_PORT = {porta do banco de dados}
- DB\_USERNAME = {usuário do banco de dados}
- DB\_PASSWORD = {senha do banco de dados}
- DB\_DATABASE = {nome do banco que será acessado}

**Linhas de comando para executar o container:**

- docker pull brunonascimentto/todos:1.0
- docker run -p 3005:3005 -d -e DB\_HOST={ host do banco de dados } -e DB\_PORT={ porta do banco de dados } -e DB\_USERNAME={ usuário do banco de dados } -e DB\_PASSWORD={ senha do banco de dados } -e DB\_DATABASE={ nome do banco que será acessado } --name todos brunonascimentto/todos:1.0

**Chamada para a api deste container:**

http://{ip público do container}:{porta}/cliente/todos

**Obs:** substituía todo o {} pelo valor da variável

## Container prodigyan-nginx

**Imagem:** brunonascimentto/prodigyan-nginx:1.0

**Porta:** 8000

**Protocolo:** http

### Variáveis de ambiente esperadas:

- IP\_CONTAINER\_TIPOS = {ip público do container tipos}
- PORT\_TIPOS = {porta do container tipos}
- IP\_CONTAINER\_GRAVAR = {ip público do container gravar}
- PORT\_GRAVAR = {porta do container gravar}
- IP\_CONTAINER\_BUSCAR\_ID = {ip público do container buscar-id}
- PORT\_BUSCAR\_ID = {porta do container buscar-id}
- IP\_CONTAINER\_BUSCAR\_EMAIL = {ip público do container buscar-email}
- PORT\_BUSCAR\_EMAIL = {porta do container buscar-email}
- IP\_CONTAINER\_TODOS = {ip público do container todos}
- PORT\_TODOS = {porta do container todos}

### Linhas de comando para executar o container:

- docker pull brunonascimentto/prodigyan-nginx:1.0
- docker run -d -p 8000:80 -e IP\_CONTAINER\_TIPOS={ ip público do container tipos } -e PORT\_TIPOS={ porta do container tipos } -e IP\_CONTAINER\_BUSCAR\_EMAIL={ ip público do container buscar-email } -e PORT\_BUSCAR\_EMAIL={ porta do container buscar-email } -e IP\_CONTAINER\_BUSCAR\_ID={ ip público do container buscar-id } -e PORT\_BUSCAR\_ID={ porta do container buscar-id } -e IP\_CONTAINER\_GRAVAR={ ip público do container gravar } -e PORT\_GRAVAR={ porta do container gravar } -e IP\_CONTAINER\_TODOS={ ip público do container todos } -e PORT\_TODOS={ porta do container todos } --name prodigyan-nginx brunonascimentto/prodigyan-nginx:1.0

**Obs:** substitua todo o \${} pelo valor da variável

**Chamada para a api deste container:**

http://{ip público do container}:8000/cliente/todos

/cliente/buscar/email/{email do cliente}

/cliente/buscar/id/{id do cliente}

/cliente/gravar

/cliente/tipos

## Exemplos

CONTAINERS

Remove ✕

**Container name**  
Container names must contain only alphanumeric characters and hyphens. A hyphen (-) can separate words but cannot be at the start or end of the name.

**Image**  
Enter the image reference from a public registry, such as DockerHub.

**Configuration**  
Optionally specify a command, the environment variables, and the ports to open on your container.  
Launch command:

**Environment variables**

Key	Value (optional)	
<input type="text" value="DB_HOST"/>	<input type="text" value="host do banco de dados"/>	✕
<input type="text" value="DB_PORT"/>	<input type="text" value="porta do banco de dados"/>	✕
<input type="text" value="DB_USERNAM"/>	<input type="text" value="usuário do banco de dados"/>	✕
<input type="text" value="DB_PASSWOR"/>	<input type="text" value="senha do banco de dados"/>	✕
<input type="text" value="DB_DATABASE"/>	<input type="text" value="nome do banco que será acessado"/>	✕

+ Add variable

**Open ports**  
Your application code for this container must listen to a port specified here.

Port	Protocol	
<input type="text" value="3001"/>	<input type="text" value="HTTP"/>	✕

+ Add port

+ Add container entry

i

 You can have up to 10 containers in a deployment

Figura 1 Exemplo de como os dados devem ser inseridos



Remove X

**Container name**  
Container names must contain only alphanumeric characters and hyphens. A hyphen (-) can separate words but cannot be at the start or end of the name.

**Image**  
Enter the image reference from a public registry, such as DockerHub.

**Configuration**  
Optionally specify a command, the environment variables, and the ports to open on your container.  
Launch command:

**Environment variables**

Key	Value (optional)	
<input type="text" value="IP_CONTAINEI"/>	<input type="text" value="ip público do container tipos"/>	X
<input type="text" value="PORT_TIPOS"/>	<input type="text" value="porta do container tipos"/>	X
<input type="text" value="IP_CONTAINEI"/>	<input type="text" value="ip público do container gravar"/>	X
<input type="text" value="PORT_GRAVAI"/>	<input type="text" value="porta do container gravar"/>	X
<input type="text" value="IP_CONTAINEI"/>	<input type="text" value="ip público do container buscar-id"/>	X
<input type="text" value="PORT_BUSCAI"/>	<input type="text" value="porta do container buscar-id"/>	X

+ Add variable

**Open ports**  
Your application code for this container must listen to a port specified here.

Port	Protocol	
<input type="text" value="8000"/>	<input type="text" value="HTTP"/>	X

+ Add port

+ Add container entry

You can have up to 10 containers in a deployment

Figura 2 Exemplo do container Nginx