StrongBerry

Manual de Instalação de Software

Índice

Repositórios utilizados

O StrongBerry possui um total de 5 repositórios utilizados para software, todos estão na organização PI2-Grupo-8 no GitHub. Os repositórios utilizados, são:

- Frontend - **https://github.com/PI2-Grupo-8/frontend**
- API de Usuários - **https://github.com/PI2-Grupo-8/user-api**
- API de Veículos - **https://github.com/PI2-Grupo-8/vehicle-api**
- API de Dados dos Sensores - **https://github.com/PI2-Grupo-8/sensors-data-api**
- Gateway - **https://github.com/PI2-Grupo-8/gateway**

Link da organização no GitHub: **https://github.com/PI2-Grupo-8**

Instalação do Docker e Docker Compose

Todos os serviços de APIs e o frontend foram desenvolvidos com a utilização de Docker e Docker Compose, sendo esses suas únicas dependências.

Para instalar o Docker no Ubuntu utilize os comandos abaixo:

```
$ sudo apt-get update
```

```
$ sudo apt-get install docker-ce docker-ce-cli containerd.io
```

Após a instalação do Docker, utilize os comandos abaixo para fazer a instalação do Docker Compose:

```
$ sudo curl -L "https://github.com/docker/compose/releases/download/1.29.2/docker-compose-$(uname -s)-$(uname -m)" -o /usr/local/bin/docker-compose
```

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

Caso esteja utilizando algum outro sistema operacional ou tenha alguma dificuldade em fazer as instalações, consulte a documentação:

- Docker: **https://docs.docker.com/engine/install/**
- Docker Compose: **https://docs.docker.com/compose/install/**

API de Veículos

A API de veículos é um dos microsserviços de backend do StrongBerry desenvolvido com o framework ExpressJS. O serviço é responsável pelo gerenciamento de veículos, comandos e trabalhos.
O projeto se encontra no repositório **https://github.com/PI2-Grupo-8/vehicle-api.**

Variáveis de ambiente

A API de veículos utiliza de variáveis de ambiente para configurar as informações do banco de dados, chave secreta para a criptografia dos tokens de acesso e para definir qual ambiente está sendo utilizado. Crie um arquivo **.env** na raiz do repositório, seguindo o modelo do **example.env**. Esse arquivo deve conter as seguintes variáveis:

- **DB_USER**: Usuário de acesso ao banco de dados;
- **DB_PASS**: Senha de acesso ao banco de dados;
- **DB_DEV**: Nome da base de dados de desenvolvimento;
- **DB_TEST**: Nome da base de dados de teste;
- **DB_HOST**: Host do banco de dados (caso utilize do banco de dados do docker utilize o nome do serviço, **vehicle_db**);
- **NODE_ENV**: Ambiente de desenvolvimento (**development** ou **test**);
- **SECRET**: Chave segredo para criptografia do token;

Observações:

- *As variáveis do .env também são utilizadas para configurar o banco de dados do docker.*
- *A chave de segredo deve ser igual em todas as APIs*

Executar o projeto

Para executar a API e o banco de dados é necessário ter Docker e Docker Compose instalados e as variáveis de ambiente configuradas. Utilize o comando abaixo para subir os containers da aplicação:

```
$ docker-compose up
```

Após a execução do comando, a API estará disponível na porta **3001**, podendo ser acessada por **localhost:3001**.

Executar o testes

Os testes da aplicação utilizam de Docker e Docker Compose e necessitam das variáveis de acesso ao banco de dados. Os testes utilizam de um próprio ambiente de testes. Para executar os testes do projeto, utilize o comando abaixo:

```
$ docker-compose run --rm -e NODE_ENV=test vehicle_api bash
-c "yarn && yarn jest --coverage --forceExit --runInBand"
```

API de Dados dos Sensores

A API de dados dos sensores é um dos microsserviços de backend do StrongBerry desenvolvido com o framework ExpressJS. O serviço é responsável pelo gerenciamento de dados coletados e de alertas. O projeto da API de dados dos sensores se encontra no repositório **https://github.com/PI2-Grupo-8/sensors-data-api.**

Variáveis de ambiente

A API de dados dos sensores utiliza variáveis de ambiente para configurar as informações do banco de dados, chave secreta para a criptografia dos tokens de acesso e para definir qual ambiente está sendo utilizado. Crie um arquivo **.env** na raiz do repositório, seguindo o modelo do **example.env**. Esse arquivo deve conter as seguintes variáveis:

- **DB_USER:** Usuário de acesso ao banco de dados;
- **DB_PASS:** Senha de acesso ao banco de dados;
- **DB_DEV:** Nome da base de dados de desenvolvimento;
- **DB_TEST:** Nome da base de dados de teste;
- **DB_HOST:** Host do banco de dados (caso utilize do banco de dados do docker utilize o nome do serviço, **sensors_data_db**);
- **NODE_ENV:** Ambiente de desenvolvimento (**development** ou **test**);
- **SECRET:** Chave segredo para criptografia do token;

Observações:

- *As variáveis do .env também são utilizadas para configurar o banco de dados do docker.*
- *A chave de segredo deve ser igual em todas as APIs*

Executar o projeto

Para executar a API e o banco de dados é necessário ter Docker e Docker Compose instalados e as variáveis de ambiente configuradas. Utilize o comando abaixo para subir os containers da aplicação

```
$ docker-compose up
```

Após a execução do comando, a API estará disponível na porta **3003**, podendo ser acessada por **localhost:3003.**

Executar o testes

Os testes da aplicação utilizam de Docker e Docker Compose e necessitam das variáveis de acesso ao banco de dados. Os testes utilizam de um próprio ambiente de testes. Para executar os testes do projeto, utilize o comando abaixo:

```
$ docker-compose run --rm -e NODE_ENV=test sensors_data_api bash -c "yarn && yarn jest --coverage --forceExit --runInBand"
```

API de Usuários

A API de usuários é um dos microsserviços de backend do StrongBerry desenvolvido com o framework ExpressJS. O serviço é responsável pela autenticação do sistema e pelo gerenciamento de usuários.
O projeto se encontra no repositório **https://github.com/PI2-Grupo-8/user-api.**

Variáveis de ambiente

A API de usuários utiliza de variáveis de ambiente para configurar as informações do banco de dados, sistema de emails, chave segredo para a criptografia dos tokens de acesso e para definir qual ambiente está sendo utilizado. Crie um arquivo **.env** na raiz do repositório, seguindo o modelo do **example.env**. Esse arquivo deve conter as seguintes variáveis:

- **DB_USER**: Usuário de acesso ao banco de dados;
- **DB_PASS**: Senha de acesso ao banco de dados;
- **DB_DEV**: Nome da base de dados de desenvolvimento;
- **DB_TEST**: Nome da base de dados de teste;
- **DB_HOST**: Host do banco de dados (caso utilize do banco de dados do docker utilize o nome do serviço, **user_db**);
- **NODE_ENV**: Ambiente de desenvolvimento (**development** ou **test**);
- **MAILER_EMAIL_ID**: Email utilizado pelo sistema para enviar emails de recuperação de senha;
- **MAILER_PASSWORD**: Senha do email utilizado pelo sistema;
- **MAILER_SERVICE_PROVIDER**: Serviço de email utilizado;
- **SECRET**: Chave segredo para criptografia do token;

Observações:

- *As variáveis do .env também são utilizadas para configurar o banco de dados do docker.*
- *A chave de segredo deve ser igual em todas as APIs*

Executar o projeto

Para executar a API e o banco de dados é necessário ter Docker e Docker Compose instalados e as variáveis de ambiente configuradas. Utilize o comando abaixo para subir os containers da aplicação:

```
$ docker-compose up
```

Após a execução do comando, a API estará disponível na porta **3002**, podendo ser acessada por **localhost:3002.**

Executar o testes

Os testes da aplicação utilizam de Docker e Docker Compose e necessitam das variáveis de acesso ao banco de dados. Os testes utilizam de um próprio ambiente de testes. Para executar os testes do projeto, utilize o comando abaixo:

```
$ docker-compose run --rm -e NODE_ENV=test user_api bash -c "yarn && yarn jest --coverage --forceExit --runInBand"
```

Frontend

O frontend da aplicação foi desenvolvido em React JS. O frontend é responsável pela interação do usuário com o sistema, funcionando como um painel de controle.
O projeto se encontra no repositório **https://github.com/PI2-Grupo-8/frontend.**

Variáveis de ambiente

O frontend utiliza de variáveis de ambiente para configurar as urls das APIs que estão sendo utilizadas. Para configurar essas variáveis crie um arquivo .env na raiz do repositório, utilizando como modelo o arquivo example.env. Esse arquivo precisa conter as seguintes variáveis:

- **REACT_APP_AUTH_API**: Endereço para a API de Usuários;
- **REACT_APP_VEHICLE_API**: Endereço para a API de Veículos;
- **REACT_APP_SENSORS_DATA_API**: Endereço para a API de Dados dos Sensores;

Executar o projeto

Para executar o projeto é necessário ter Docker e Docker Compose instalados e as variáveis de ambiente configuradas. Utilize o comando abaixo para subir o container da aplicação:

```
$ docker-compose up
```

O projeto estará disponível na porta **3000**, sendo possível acessar localmente por **localhost:3000.**

Gateway

Componente de gateway do StrongBerry é responsável pela comunicação entre o sistema embarcado e as APIs. O gateway foi desenvolvido para ser executado em uma ESP 32 LoRa que deve ser localizada perto da área de trabalho do veículo.
O projeto se encontra no repositório **https://github.com/PI2-Grupo-8/gateway.**

Dependências

Diferente dos serviços de backend e frontend, o gateway não utiliza de docker e possui dependências que precisam ser instaladas. É necessário também ter uma placa ESP 32 conectada no computador via cabo USB para conseguir executar o programa.

Ambiente de desenvolvimento da ESP 32

O sistema utiliza o ambiente de desenvolvimento da ESP 32, que deve ser instalado no computador. Utilize o comando abaixo para fazer a instalação no Linux Ubuntu ou Debian.

$ sudo apt-get install git wget flex bison gperf python3 python3-pip python3-setuptools cmake ninja-build ccache libffi-dev libssl-dev dfu-util libusb-1.0-0

Para a instalação em outros sistemas operacionais, veja os links abaixo:

- Linux: **https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/linux-setup.html**
- Windows: **https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/windows-setup.html**
- MacOS: **https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/macos-setup.html**

ESP-IDF

O projeto utiliza da biblioteca ESP-IDF da Espressif, para instalá-la é necessário baixar o repositório com os seguintes comandos:

```
$ mkdir -p ~/esp

$ cd ~/esp

$ git clone --recursive https://github.com/espressif/esp-idf.git
```

Dessa forma a ESP-IDF será salva em **~/esp/esp-idf** e para instalar, é necessário entrar na pasta e executar o script de instalação. Veja os comandos abaixo:

```
$ cd ~/esp/esp-idf

$ ./install.sh esp32
```

Com a biblioteca instalada, execute o comando abaixo dentro da pasta do projeto para incluir as variáveis no PATH:

```
$ . $HOME/esp/esp-idf/export.sh
```

Após as instalações, dentro da pasta do projeto execute o comando abaixo para configurar o projeto para ser utilizado na ESP 32.

```
$ idf.py set-target esp32
```

Caso tenha algum problema em algum dos passos de instalação mencionados, acesse a documentação da Espressif no link abaixo:

- **https://docs.espressif.com/projects/esp-idf/en/latest/esp32/get-started/index.html#get-started-connect**

Variáveis de ambiente

As variáveis de ambiente são configuradas na interface de configuração da Espressif. Rode o comando abaixo para acessar a interface:

$ idf.py menuconfig

O gateway utiliza variáveis de ambiente para configurar o acesso às APIs, o código de fábrica do veículo e para o token de acesso às APIs. Para acessar essas variáveis na interface de configuração acesse a opção **StrongBerry Configuration** e configure as seguintes variáveis:

- **Vehicle API Route**: Endereço para a API de Veículos;
- **Sensors Data API Route**: Endereço para a API de Dados dos Sensores;
- **Code**: Código de fábrica do veículo;
- **Token**: Token para acesso às APIs;

Observação: Caso esteja executando as APIs no seu computador, utilize o seu IP e as portas de cada serviço.

Executar o projeto

Com o ambiente configurado e com os serviços de backend e frontend em execução e com os links devidamente colocados nas variáveis de ambiente do gateway, execute o gateway seguindo os passos abaixo:

PASSO 1 - Crie no sistema um veículo com o código do veículo fornecido nas variáveis do Gateway. Caso não tenha sido alterado, o código é XLR8;

PASSO 2 - Conecte a ESP 32 no computador;

PASSO 3 - Rode o comando abaixo para gravar o firmware na placa e monitorar o terminal:

$ idf.py -p [PORT] flash monitor

Troque **[PORT]** pela porta onde a ESP 32 está conectada no seu computador. Veja abaixo o padrão de portas para cada sistema:

- Linux - **/dev/tty;**
- Windows - **COM1;**
- MacOS - **/dev/cu;**

Exemplo:
$ idf.py -p /dev/ttyUSB0 flash monitor

PASSO 4 - Ligue a ESP 32;

PASSO 5 - Por outro dispositivo acesse o sinal de WiFi StrongBerry que será emitido pela placa;

PASSO 6 - Neste outro dispositivo conectado à rede do gateway acesse o endereço **192.168.4.1** via browser, esse endereço dará acesso a uma página como a da imagem abaixo. Forneça os dados de nome da rede e senha do WiFi local e clique em submit. Isso permitirá com que o Gateway se conecte à Internet.

Após todos os passos, o gateway estará conectado ao backend e pronto para enviar e receber dados.