senvoAPI

senvo operates on shipment-related data from e-commerce and logistics sys-tems. This API stores and provides shipment invoice data.

Installation

These steps will guide you through the process of local installation and launch of the API.

1. Open the “senvoAPI.zip” file and extract the folder.

2. To run the project it is recommended to use Docker Compose. Follow this Docker guide to install it.

3. Open Docker App to start the daemon.

4. Run the following commands inside the path `/senvoAPI/src/` to build the docker image and run the project:

   ```
   $ docker-compose up --build
   ```

5. Once all the files have been installed, you can navigate to `http://localhost:8000` using your browser and you will get the following response:

   ```
   {"message": "Hello World"}
   ```

6. The documentation of the API is available at `http://localhost:8000/docs#/`

Endpoints

Shipments: Upload a list of shipments

```
POST localhost:8000/api/v1/shipments/
```

More information on this endpoint is available at http://localhost:8000/docs#/default/post_shipments_api_v1_shipments__post

Shipments: Read a list of shipments

```
GET localhost:8000/api/v1/shipments/
```

More information on this endpoint is available at http://localhost:8000/docs#/default/get_shipments_api_v1_shipments__get

Testing

You can run four performance tests after running the project.

1. Ensure Docker Compose is Running: First, make sure that your Docker Compose environment is up and running. If it isn't already running, start it without running the tests:

   ```
   $ docker-compose up --build
   ```

2. Open a Shell into the test Container: Once your containers are up and running, you can open a shell into the test container with the following command:

   ```
   $ docker-compose exec test /bin/sh
   ```

3. Run Your Tests Inside the Container: After running the docker-compose exec command, you should now have an interactive shell session inside the test container.

   Inside the shell, you can run your tests with:

   ```
   # pytest tests/test_performance.py
   ```

   This command will execute the pytest command for the test_performance.py file inside the container.

Monitoring

You can monitor the API using Apitally. Apitally is an API monitoring & analytics tool for Python and Node.js web applications that is easy to use and respects your data privacy.

This feature allows:

- Monitor traffic to the FastAPI application on the simple dashboard.
- Keep track of errors, response times and payload sizes.
- Understand how individual API consumers use your application.
- Monitor your application's uptime and receive alerts when it's down.

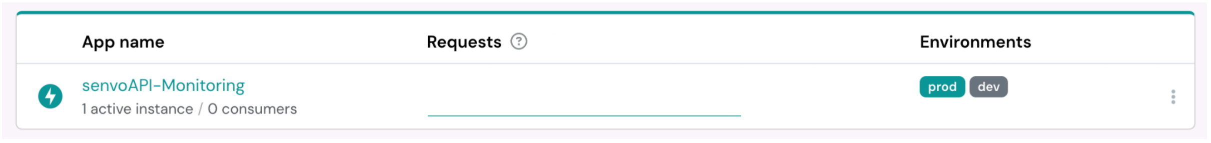
senvoAPI is already connected and configured to interact with Apitally.

To access the dashboard follow the steps below:

1. Navigate to Apitally Login Page and log in with the following credentials:

```
email address: apitestingsenv@gmail.com
password:      Apisenvotesting28
```

2. You can access the Dashboard by clicking on "senvoAPI-Monitoring"



App name	Requests	Environments
senvoAPI-Monitoring 1 active instance / 0 consumers		prod dev

Software Architecture and API Development

Technologies used: senvoAPI was developed using Python as the main programming language. FastAPI was used to build the API because this framework, based on standards such as OpenAPI and JSON Schema, allows for efficient and secure development. SQLAlchemy and PostgreSQL were used for effective database connection and management. In addition, application containerization with Docker and Docker Compose was used to ensure fast and consistent deployments and easy integration with CI/CD tools.

Modular Architecture: The project is organized in clear modules (controllers, routers, models, schemas, repositories) to facilitate code readability, maintenance and scalability.

Testing: Automated testing with Pytest was implemented to ensure code quality and prevent regression, increasing the reliability of the software.

Monitoring: The use of a monitoring platform was implemented to monitor API requests, errors, performance and uptime.