**Crypto recommendation service**  
Project structure specification  
app version: 1.0.0  
created: 21.3.2023  
author: Sebastian Jaks

1. **Project overview**

The service is designed to read prices of cryptocurrencies from data files. It exposes HTTP API with several endpoints for retrieving statistical data about those cryptocurrencies. Those endpoints are described in separate file („Crypto recommendation service - endpoints spec“).

The project code is divided into three main layers – DAO, Business and REST API. Those layers correspond to the first three packages described in the next chapter.

1. **Overview of main packages**

*business*

Contains single class *CurrencyService* that contains logic that provides statistical data for the REST API. Represents a layer between DAO and REST APIS. Class is responsible for generating statistics and creating DTO objects for the REST API.

*dao.files\_storage*

Contains classes responsible for reading data from the filesystem. Contains two classes:   
  
*CurrencyParser* – Responsible for parsing the input file.  
*CurrencyFileReader* – Responsible for searching for input files, calls the parser and generates output for layers above DAO layer.

*rest\_api*

Contains the front-end functionality of the service. Consists of three packages:

*controllers* – Contains class CryptoStatisticsController that defines all HTTP endpoints.  
*filters* – Contains currently two classes responsible for rate-limiting user requests and block excessive traffic.  
*validators* – Contains class ParametersValidator that contains functionality to validate HTTP request parameters.

*dto*

Contains two packages:

*currency\_data* – Contains DTO objects representing data loaded from filesystem.  
*rest\_objects* – Contains DTO objects used to be exported by specific HTTP endpoints. Those objects have the structure that correspond to the data structure that the endpoints export (in form of JSON).

1. **Configuration**

The application is expected to be configured by external configuration file. This file should be located in the config directory under the current working directory. Example of the filepath: *current\_directory/config/application.properties*

For now there are only two parameters expected to be present in the file:

*user-request-limit* – Integer defining the maximum number of user requests per defined interval. Users are defined by the source IP address.

*user-request-limit-interval* – Integer defining the time interval (in seconds) for which the *user-request-limit* should be applied.

1. **Containerization**

The project is ready to be deployed as a docker container. The dockerfile is located in the code repository in the folder named „docker“.

The application stores/loads data to/from three folders thad should be mapped to host folders. Those folders are:

*/app/logs –* Folder where the application stores logs.

*/app/input\_data –* Folder from where the application reads the files with cryptocurrencies.

*/app/config –* Folder where the config file should be located.

Example of the command to start the container:

*docker run --rm -p -d 8080:8080   
-v local\_path\_to\_input\_files\_folder:/app/input\_files   
-v local\_path\_to\_logs\_folder:/app/logs   
-v local\_path\_to\_config\_folder:/app/config crypto\_recommendation\_service*

1. **Possible future improvements**

* Do not read the data from disc with every request – use WatchService to read the data only when changed and store them in memory.
* Store the data in SQL database and then read from the database (add new DAO layer). This will allow making statistics of older or historical data.
* Create IP blacklist in SQL database to permanenty block malicious traffic.