## Описание

Реализована группа микросервисов, в соответствии с тестовым заданием (Тестовое задание (python-dev). T-digital.docx).

Транспортный узел

Сервис, отвечающий за связь поставщиков данных (сервис «Шлюз данных») и потребителей данных («Логгер», «Консоль»).

Поставщик данных, передает сообщения на транспортный узел в формате JSON. В сообщении, помимо основных данных, передается поле topic, которое позволяет связать потребителей и поставщиков сообщений. Topic передается в каждом сообщении, которое поставщик отправляет на транспортный узел.

Потребитель данных, при подключении, использует поле headers, при установке соединения, в котором указан topic, на который подписывается данный потребитель (пример: Логгер подписан на “BTCUSDT\_1m”). Потребители не отправляют сообщения на транспортный узел, только получают.

Шлюз данных

Сервис подключается к API биржи Binance, с использованием библиотеки *python-binance* через протокол WebSocket.

Является поставщиком данных, пересылает (не изменяя полей) сообщения от Binance API на транспортный узел. Также аггрегирует поступившие сообщения (по времени сервера Binance) за одну минуту и отправляет полученные данные на транспортный узел.

Логгер

Осуществляет запись в файл аггрегированных по сделкам данных, в текстовом формате (*timestamp: 2021-04-27 12:03; open: 55000; low: 54990; high: 55023; close: 55019).*

Является потребителем данных, подписан на топик “BTCUSDT\_1m”, куда приходят данные за одну минуту.

Консоль

Выводит данные в реальном времени, подписан на топик “BTCUSDT”.

## Установка с использованием make

make build

## Установка без использования make

Windows: build.bat

\*nix: build.sh

При установке создается новая виртуальное окружение venv, обновляется pip, устанавливаются необходимые пакеты из файла requirements.txt.

## Запуск сервисов

make run

Либо run.bat (в Windows), либо run.sh (в \*nix).

## Примечание

Требование возможности запуска в любой среде, наложило ограничения на использование стороннего ПО в данном решении.

Условия задачи позволяют определить схему, на которой явно видны поставщики данных (в данной работе Шлюз данных) и потребители (Логгер, Консоль). При этом один поставщик может рассылать сообщения сразу нескольким потребителям. Для реализации данной схемы, можно было использовать один из брокеров сообщений. Например Kafka, RabbitMQ и другие. Для развертывания микросервисов, можно использовать Docker.

### Схема

