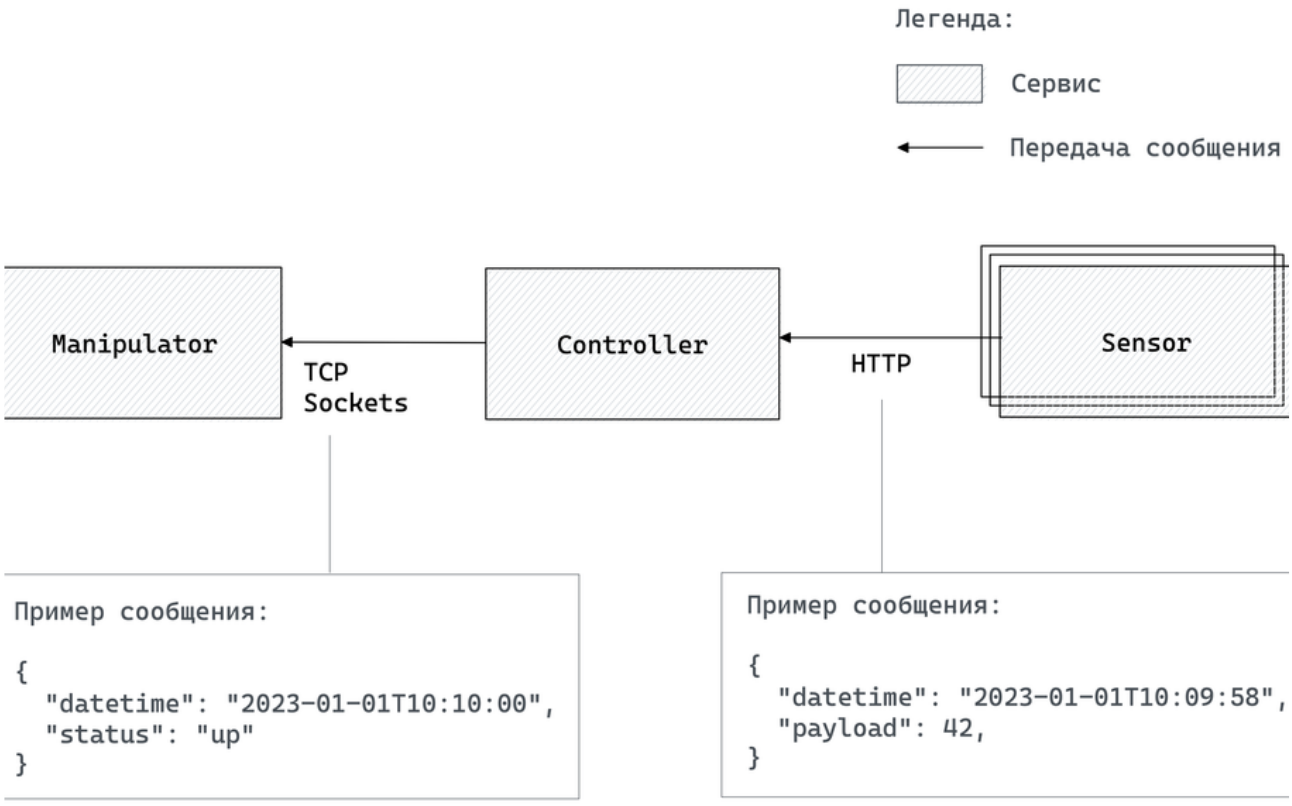
# [Backend] Test task



## Архитектура

На диаграмме изображены компоненты, которые нужно реализовать и обеспечить их взаимодействие.




Динамический вид системы

## Описание компонентов системы

### Sensor

Компонент, генерирующий данные, на основе которых контроллер принимает решение. Всего 8 сенсоров, каждый генерирует 300 сообщений в секунду, сообщения равномерно распределены по секунде и отправляются раздельно (один запрос на одно сообщение). Алгоритм генерации данных на усмотрение.

### Controller

Компонент, который по TCP соединению управляет манипулятором, на базе данных от сенсоров. Алгоритм принятия решения о статусе на усмотрение, но должен использовать данные с сенсоров.

API контроллера должен соответствовать REST API.

Обработка данных с сенсоров происходит параллельно/асинхронно, однако важно обрабатывать сообщения в интервалы 5 секунд. Т. е. каждые 5 секунд принимается решение об управляющем сигнале для манипулятора.

Outdated информация не должна приниматься во внимание при принятии решения. Сообщение считается outdated, если информация из этого сообщения имеет время создания раньше принятия последнего решения об управляющем сигнале.

Контроллер должен уметь через API отдавать список отправленных управляющих сигналов за указанный интервал времени. При этом повторяющиеся записи должны объединяться в общий интервал.

**Пример**

История управляющих сигналов:

`[12:00 - 12:05 UP], [12:05 - 12:10 UP], [12:10 - 12:15 UP], [12:15 - 12:20 DOWN], [12:20 - 12:25 DOWN]`

Результат запроса списка сигналов:

`[12:00 - 12:15 UP], [12:15 - 12:25 DOWN]`

### Manipulator

Компонент, который принимает сигналы по TCP соединению и выводит их в консоль/логи для демонстрации.

## Формат сообщений

### Sensor → Controller

```
{
  "datetime": "%Y-%m-%dT%H:%M:%S",
  "payload": int
}
```

### Controller → Manipulator

```
{
  "datetime": "%Y-%m-%dT%H:%M:%S",
  "status": "up" | "down"
}
```

Для демонстрации необходимо выводить изменение статуса контроллера как в логи контроллера, так и на веб сервере. Условие для изменения статуса задать произвольное.

## Результат

- docker-compose конфигурация
- исходный код системы - основной код на Python 3, допускается использование любых библиотек/фреймворков/инструментов
- документация по запуску системы (например, README.md)
- репозиторий в GitHub с результатами предыдущих пунктов

Наличие CI и тестов приветствуется, но не является обязательным