**Задание на разработку веб-сервисов RESTfull**

# Цель работы

Понять принципы проектирования и разработки веб-сервисов RESTfull на примере платформы Spring.

## Задание

**Приложение**: Сервер API (JSON HTTP API).  
**Средства разработки:** Java.  
**Framework**: Spring boot последней версии.  
**База данных**: любая реляционная база данных с открытым исходным кодом (Open Source).  
**Протокол**: HTTP

## Функиональные возможности

1. Добавление данных:
   1. Передаём на сервер данные в виде JSON-объекта;
   2. Сохраняем информацию в базе данных;
   3. Получаем ответ сервера – идентификатор созданной сущности.
2. Получение данных о всех сущностях:
   1. Выполняем запрос на сервер;
   2. Читаем информацию из базы данных;
   3. Получаем ответ сервера – список сущностей с их данными.
3. Получение данных о конкретной сущности:
   1. Передаём на сервер уникальный идентификатор сущности;
   2. Читаем информацию из базы данных;
   3. Получаем ответ сервера – данные сущности.
4. Редактирование данных сущности:
   1. Передаём на сервер сущность с изменёнными данными;
   2. Сохраняем данные сущности в базе;
   3. Получаем ответ сервера – сущность с изменёнными данными.

**Примечание**. Выбор сущностей осуществляется самостоятельно. Поскольку это тестовое задание, для демонстрации работы методов API достаточно включить поддержку Swagger.

## Обязательные требования

* Использование стандарта RESTfull;
* Все данные должны быть представлены в формате JSON;
* База данных должна состоять минимум из **двух** связанных таблиц;
* Использовать объектно-реляционное преобразование Spring Data JPA.

## Необязательные требования (желательно)

* Код должен быть задокументирован;
* Реализация интеграционных тестов (unit testing);
* Обработка ошибок в API-сервисах;
* Для получения списков использовать:
  + Постраничную загрузку,
  + Сортировку,
  + Фильтрацию по содержимому полей (поиск сущностей);
* Использовать миграции Flyway.

## Дополнительные задачи (по личному желанию)

1. Реализовать загрузчик файлов:
   1. Передаём на сервер файл (например, картинку);
   2. Сохранить картинку в каталоге на сервере;
   3. Получить ответ сервера – внутренний URI файла.
   4. Использовать данный URI в другой сущности.
2. Сервер API должен быть спроектирован с учетом того, что запросы 2 и 3 имеет высший приоритет (по отношению к запросам 1, 4) и должны быть выполнены максимально быстро.

## Результат задания

* Результат тестового задания должен быть предоставлен в архиве и с инструкцией по его развертыванию. Можно загрузить на <http://github.com>.
* Должен содержать краткую документацию созданного API (список запросов, параметры запросов, форматы запросов, форматы ответов и т.д.). Можно скопировать из Swagger.
* Информация о времени, потраченном на тестовое задание в разрезе: проектирование, программирование, документация и т.д.