МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ **«БЕЛГОРОДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНОЛОГИЧЕСКИЙ УНИВЕРСИТЕТ им. В. Г. ШУХОВА» (БГТУ им. В.Г. Шухова)**

Кафедра программного обеспечения вычислительной техники и автоматизированных систем

Лабораторная работа №5  
по дисциплине: Технологии Web-программирования

Выполнил: Ляхов Степан ПВ-42

Проверил: Картамышев С. В.

Белгород 2020г.

Лабораторная работа №5

Технологии web-программирования.

**«REST API»**

**Цель**: изучить основы разработки API для web-приложений. Разработать REST API для своего проекта.

**Задание к лабораторной работе:**

1. Изучить структуру формата представления данных JSON.
2. Изучить типы запросов к API: HEAD, GET, POST, PUT, DELETE.
3. Спроектировать и реализовать собственное REST API (Получение, создание, изменение и удаление каких-либо объектов).
4. В отчёт необходимо предоставить документацию к использованию методов. (Либо словесным описание, либо через Swagger)

**Выполнение**

В данной лабораторной работе мы будем реализовывать REST API на строне бека, основываясь на базе данных из прошлого

Язык программирования: Java 8

Фреймворк: Spring, Spring Boot, JPA, Hibernate

Система сборки: Maven

База данных: PostgeSQL

В прошлой лабораторной работе у нас бы были готовы репозитории для работы с нашими сущностями. Теперь нужно написать бизнес-логику и контроллеры для работы с нашим беком.

**Разработка DTO**

Приходящий у нам JSON нужно мапить в какие-то сущности, для большей простоты работы с данными и конвертации их потом в сущности в базе данных.

Они выглядят например так:



**Разработка контроллеров**

Разберём разработку контроллера на примере



@RestController – аннотация, которая говорит, что наш контроллер будет получать и передавать данные в формате REST API.

@RequestMapping(LISTS) – так как у нас будет много контроллеров, мы создаём иерархию endpoint-ов. Конкретно в этом пример, все запросы, которые будут взаимодействовать со списком, будут начинаться с /lists

После мы описываем методы.

@GetMapping("/getAll") – аннотация, которая указывает тип запроса и по какому url нужно обратиться к этому методу.

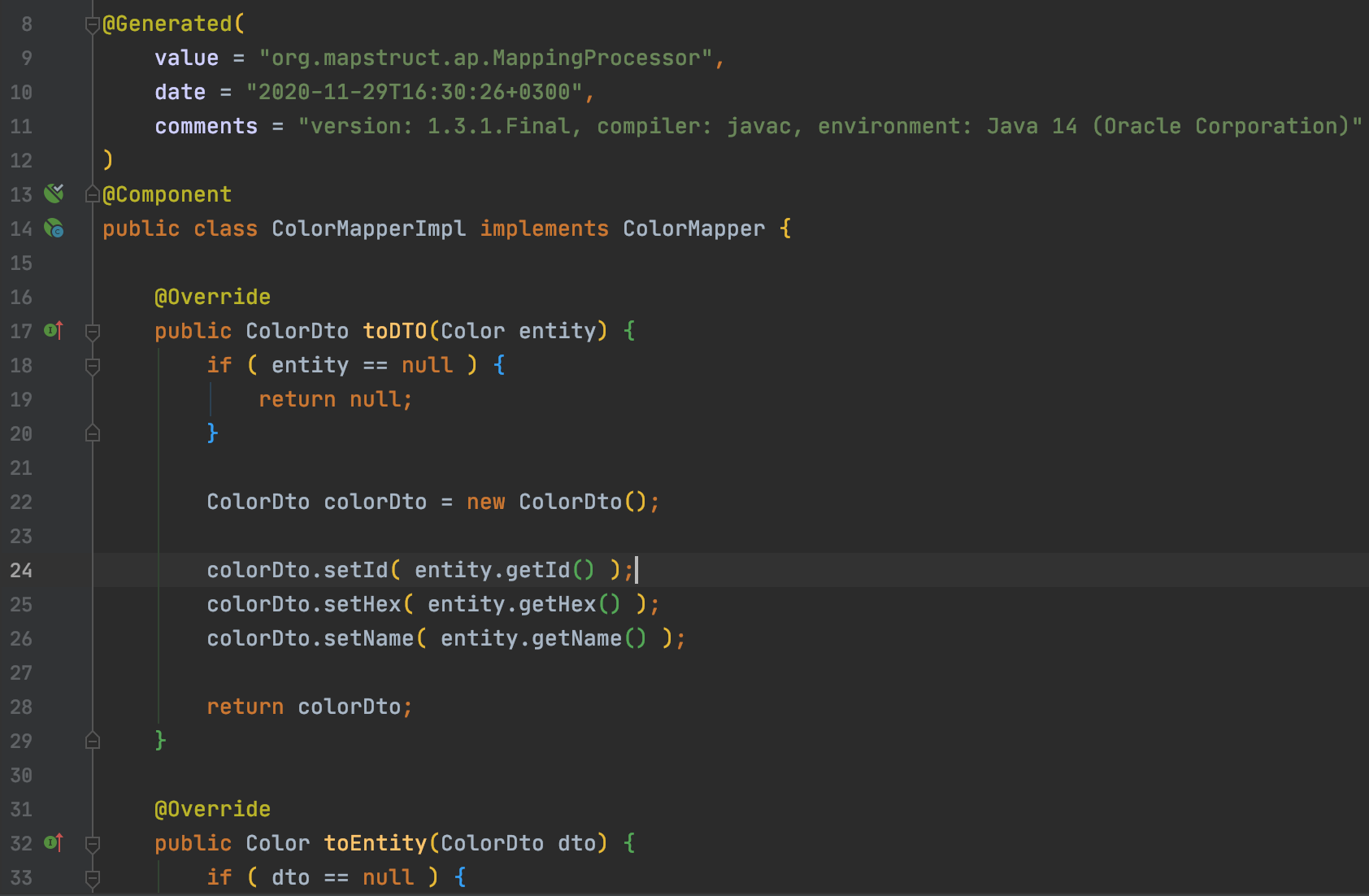
@RequestBody – аннотация, которая, преобразует наш JSON в Dto Object.

**Разработка мапперов**

С фронтом мы общаемся через DTO, а с базой данных мы общаемся через сущности. Чтобы преобразовывать одно в другое, мы будем применять мапера, а конкретно библиотеку mupstruct. Мы задаём интерфейс для мапера и объясняем, как поступать в некоторых ситуациях, реализацию библиотека сгенерит сама.

Пример:

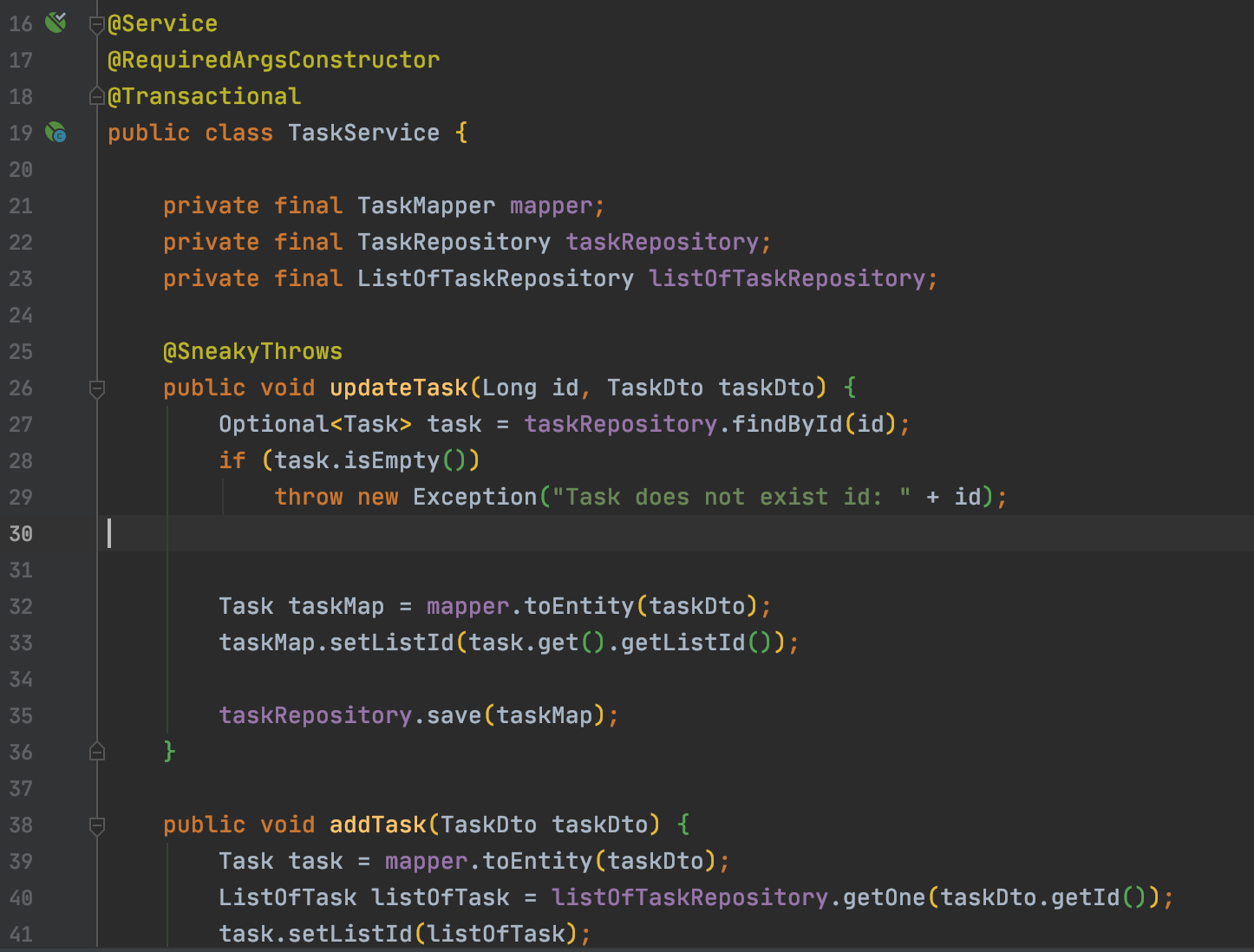




**Разработка сервисов**

Слой сервисов – это слой, который реализует бизнес-логику нашего приложения.

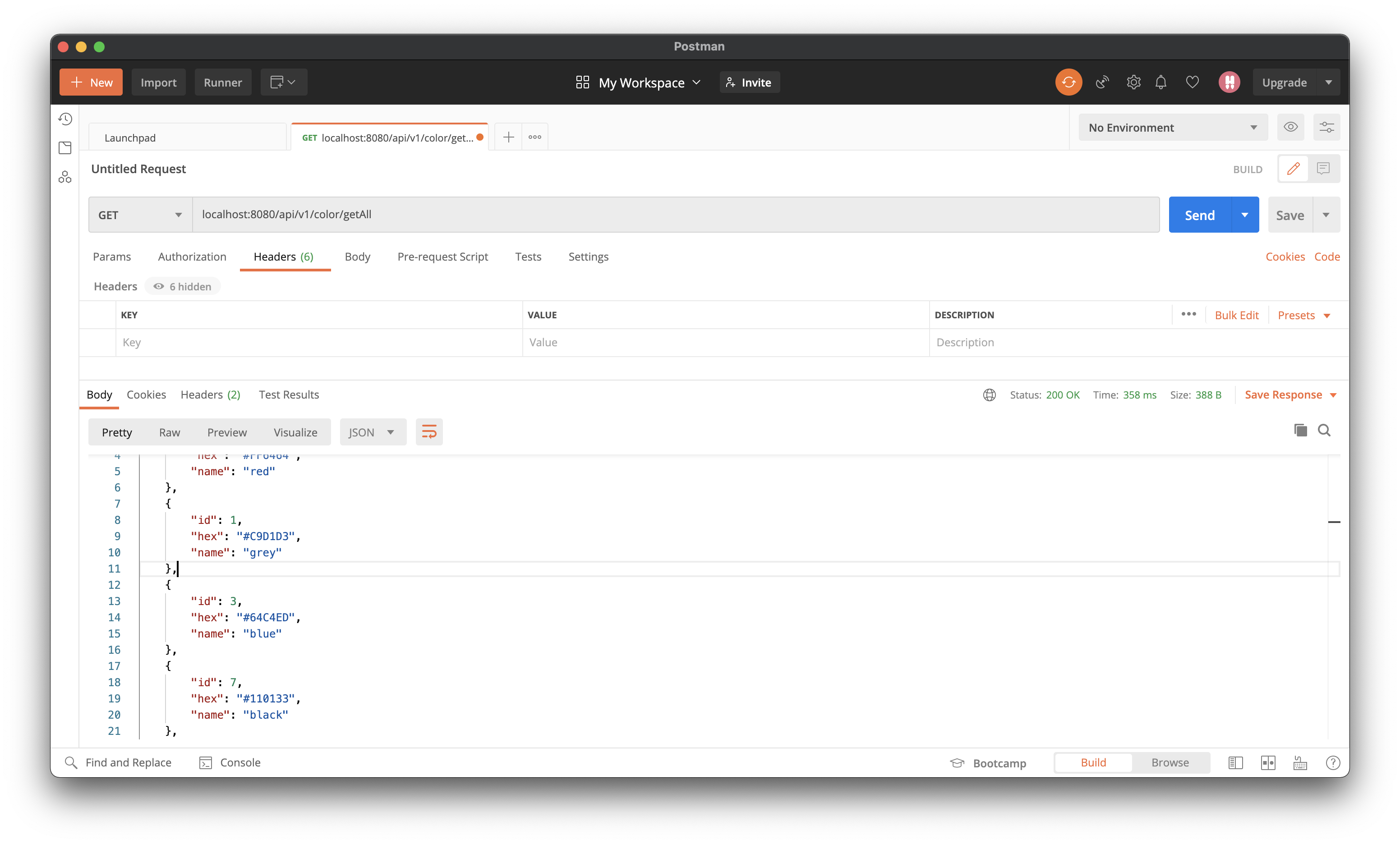
Пример:



Сервис решает, что во что мапить, что сохранять в базу и так далее.

**Пример работы с REST API**

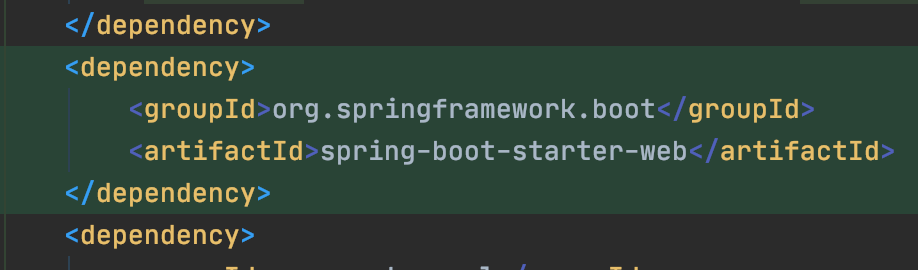
На данном примере продемонстрировать запрос на получение всех цветов.

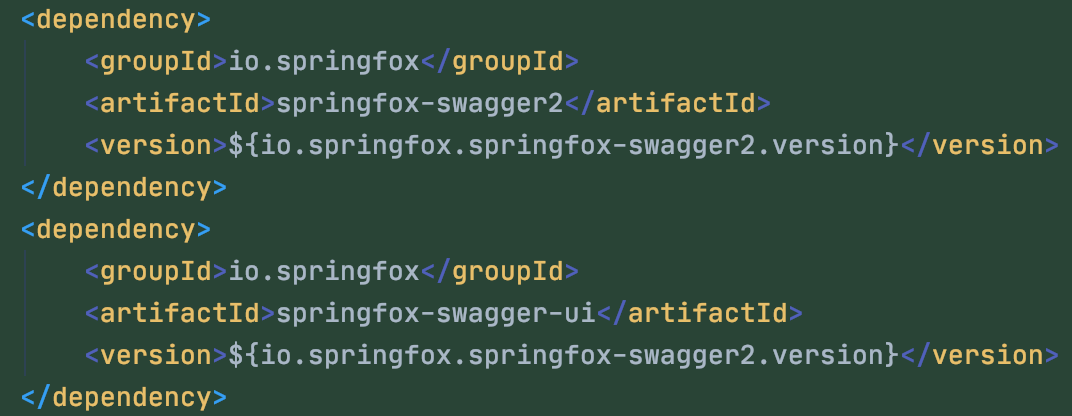


**Документация API**

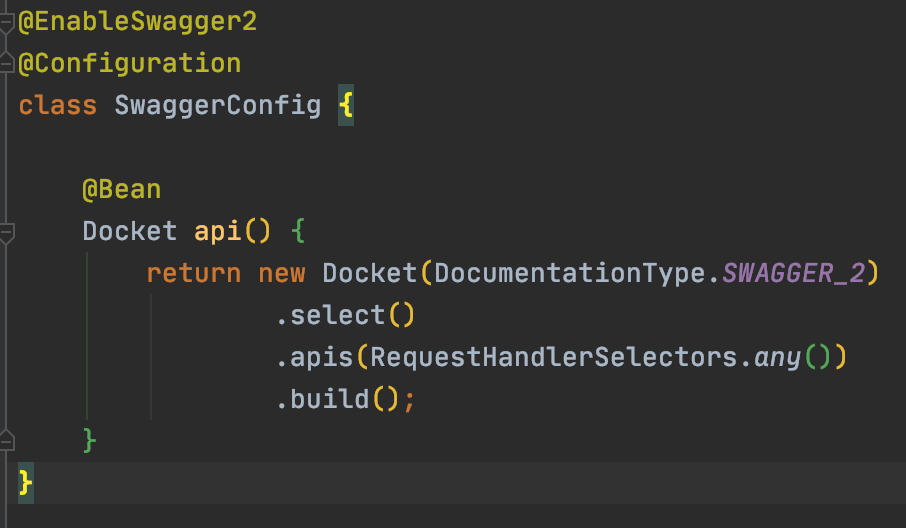
Для документации API мы будем использовать популярную библиотеку swagger.

Подключаем библиотеки





Настраиваем конфигурацию свагера



После этого доступ к документации в формате JSON можно получить по url при включённом беке.

GET URL: <http://localhost:8080/v2/api-docs>

