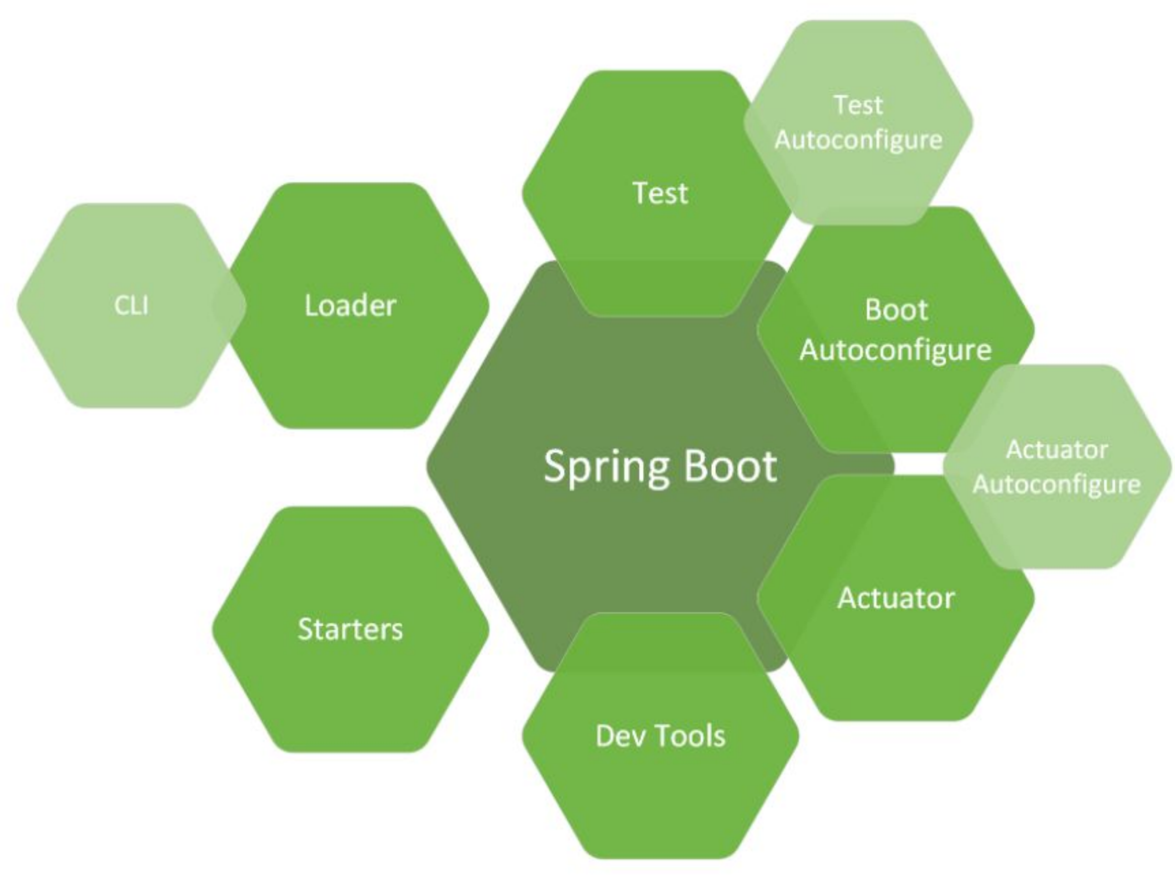
Projekt - część 2

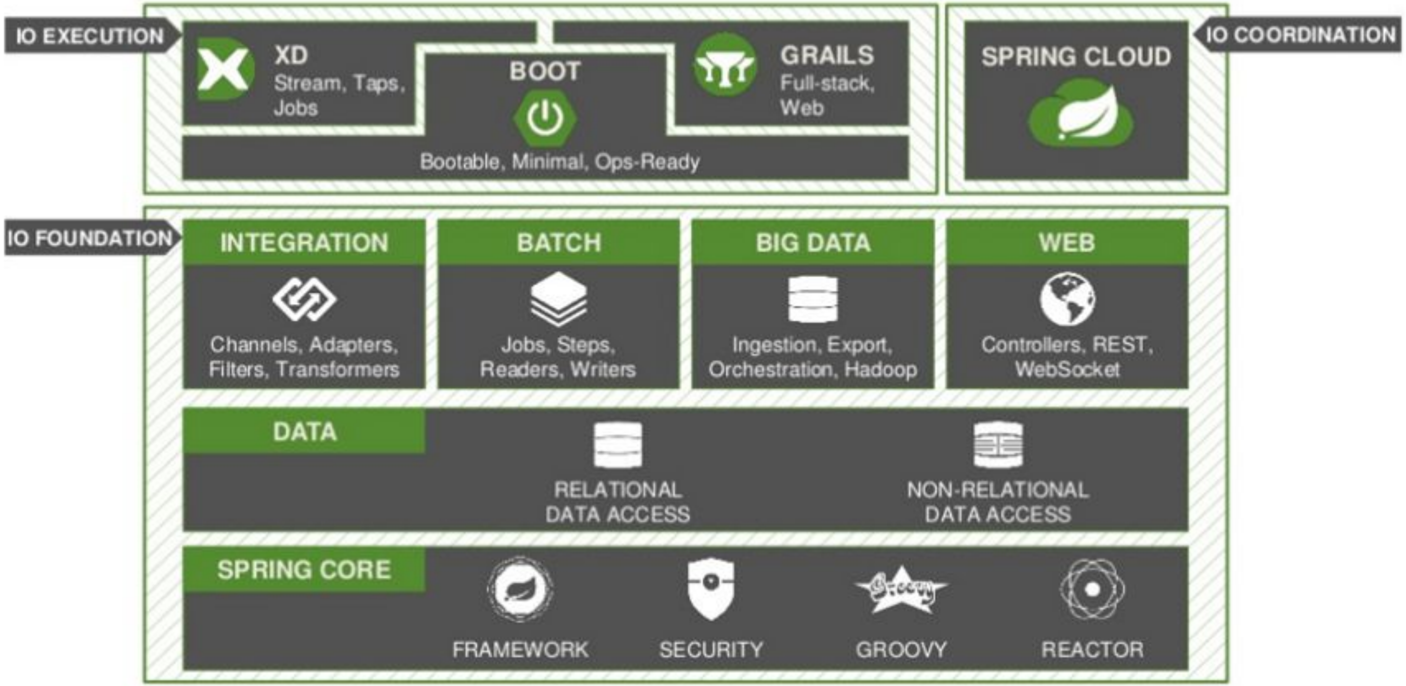
•••

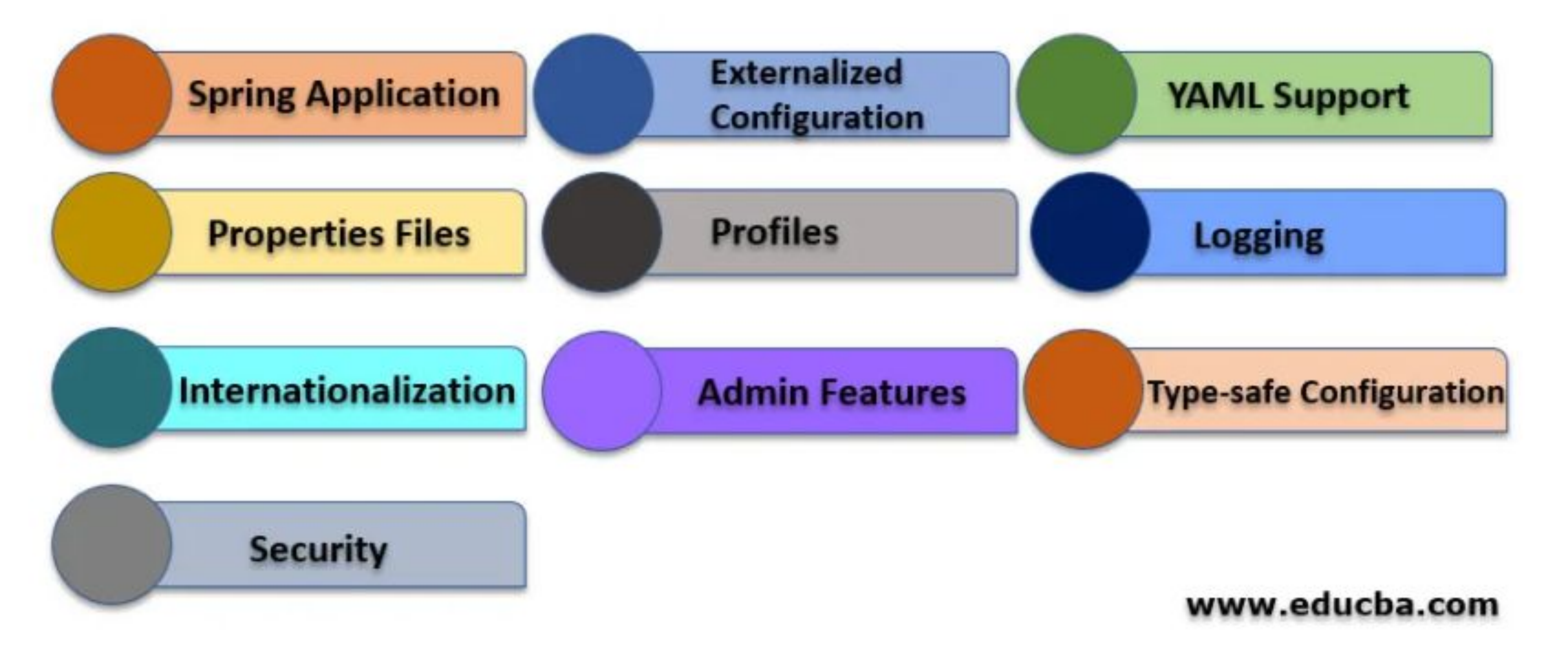
Rozkład jazdy - dzień 2

Slot 1:	9:00 - 10:20	Spring Boot
Slot 2:	10:30 - 11:40	Rest API
Slot 3:	11:50 - 13:00	Bazy danych
Przerwa:	13:00 - 13:30	
Slot 4:	13:30 - 15:00	
Slot 5:	15:10 - 16:30	


Test
Autoconfigure
Test
CLI
Loader
Boot
Autoconfigure
Spring Boot
Actuator
Autoconfigure
Actuator
Starters
Dev Tools

Spring IO Platform


IO EXECUTION
XD
Stream, Taps, Jobs
BOOT
Bootable, Minimal, Ops-Ready
GRAILS
Full-stack, Web
SPRING CLOUD
IO COORDINATION
IO FOUNDATION
INTEGRATION
Channels, Adapters, Filters, Transformers
BATCH
Jobs, Steps, Readers, Writers
BIG DATA
Ingestion, Export, Orchestration, Hadoop
WEB
Controllers, REST, WebSocket
DATA
RELATIONAL DATA ACCESS
NON-RELATIONAL DATA ACCESS
SPRING CORE
FRAMEWORK
SECURITY
GROOVY
REACTOR



Spring Application

Externalized Configuration

YAML Support

Properties Files

Profiles

Logging

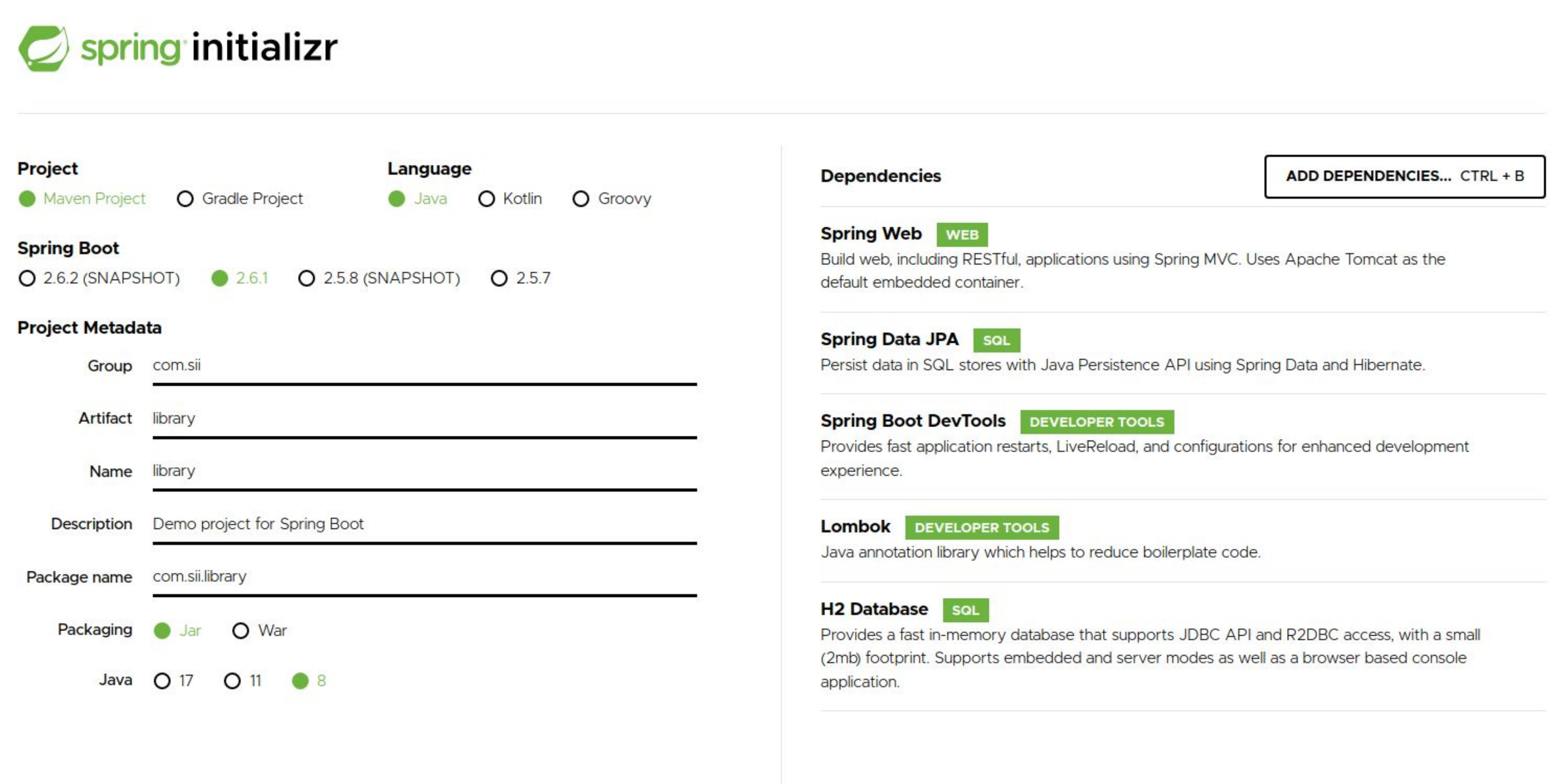
Internationalization

Admin Features

Type-safe Configuration

Security

www.educba.com



Project

● Maven Project ○ Gradle Project

Language

● Java ○ Kotlin ○ Groovy

Spring Boot

○ 2.6.2 (SNAPSHOT) ● 2.6.1 ○ 2.5.8 (SNAPSHOT) ○ 2.5.7

Project Metadata

Group: com.sii

Artifact: library

Name: library

Description: Demo project for Spring Boot

Package name: com.sii.library

Packaging: ● Jar ○ War

Java: ○ 17 ○ 11 ● 8

Dependencies

Spring Web WEB
Build web, including RESTful, applications using Spring MVC. Uses Apache Tomcat as the default embedded container.

Spring Data JPA SQL
Persist data in SQL stores with Java Persistence API using Spring Data and Hibernate.

Spring Boot DevTools DEVELOPER TOOLS
Provides fast application restarts, LiveReload, and configurations for enhanced development experience.

Lombok DEVELOPER TOOLS
Java annotation library which helps to reduce boilerplate code.

H2 Database SQL
Provides a fast in-memory database that supports JDBC API and R2DBC access, with a small (2mb) footprint. Supports embedded and server modes as well as a browser based console application.

Spring Boot - Top 8 adnotacji

@Bean

@Service

@Component

@Configuration

@Autowired

@Qualifier

@SpringBootApplication

@Controller

@RequestMapping

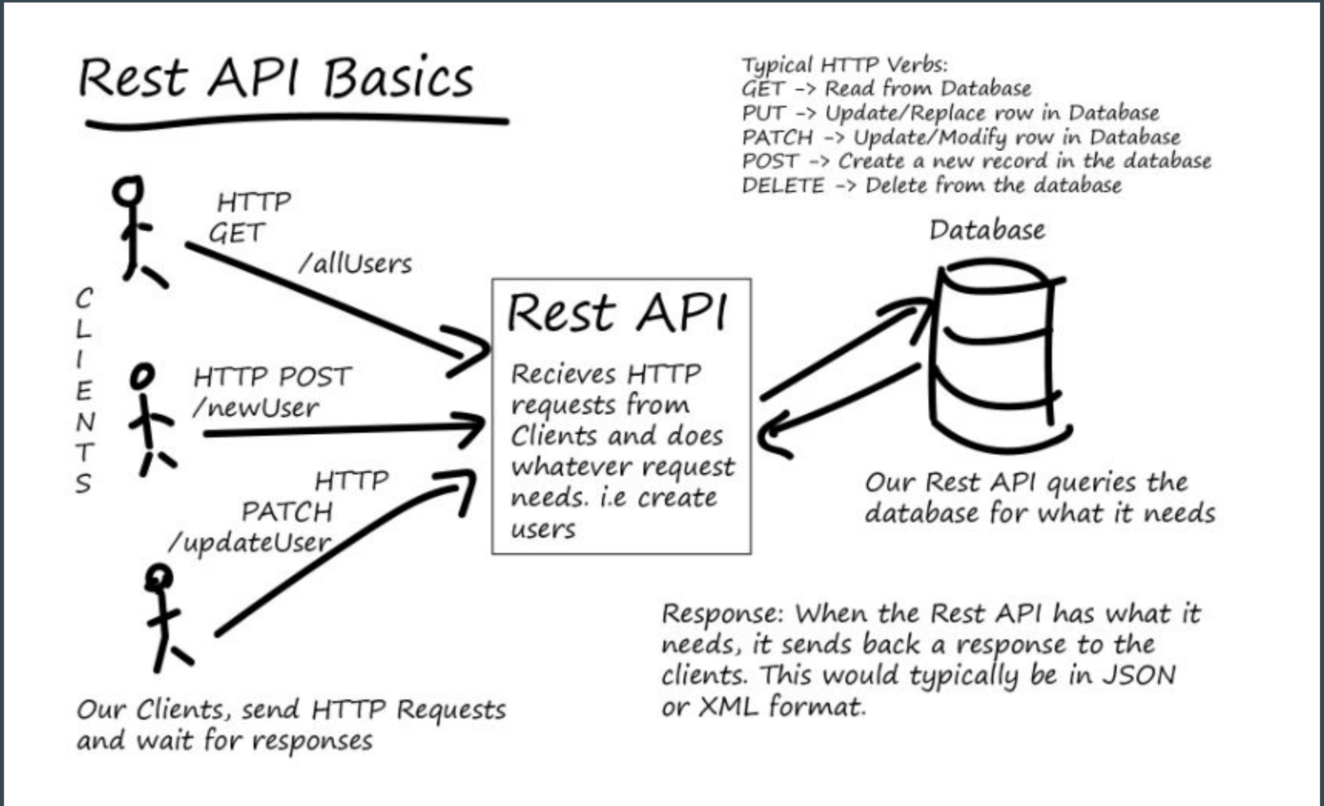
@Scheduled

@EnableScheduling

REST API

Representational State Transfer
Application Programing Interface

- Uniform interface
- Client-server architecture
- Stateless
- Cacheable
- Layered System



Rest API Basics

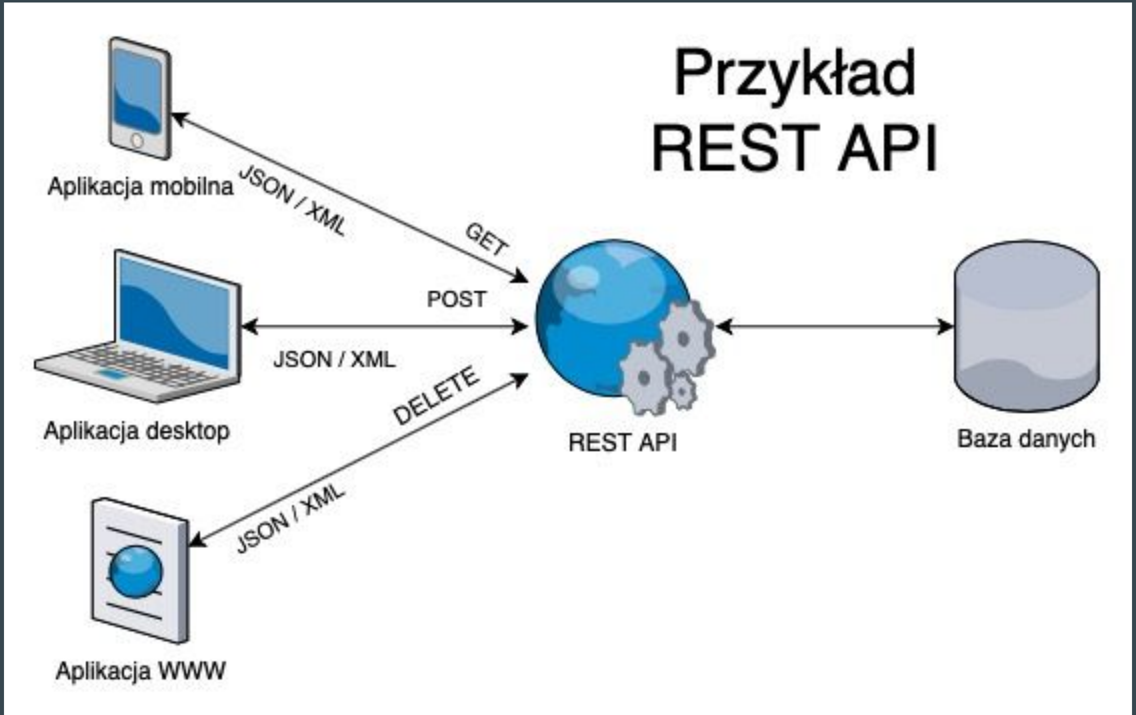
Typical HTTP Verbs:
GET -> Read from Database
PUT -> Update/Replace row in Database
PATCH -> Update/Modify row in Database
POST -> Create a new record in the database
DELETE -> Delete from the database

Recieves HTTP requests from Clients and does whatever request needs. i.e create users

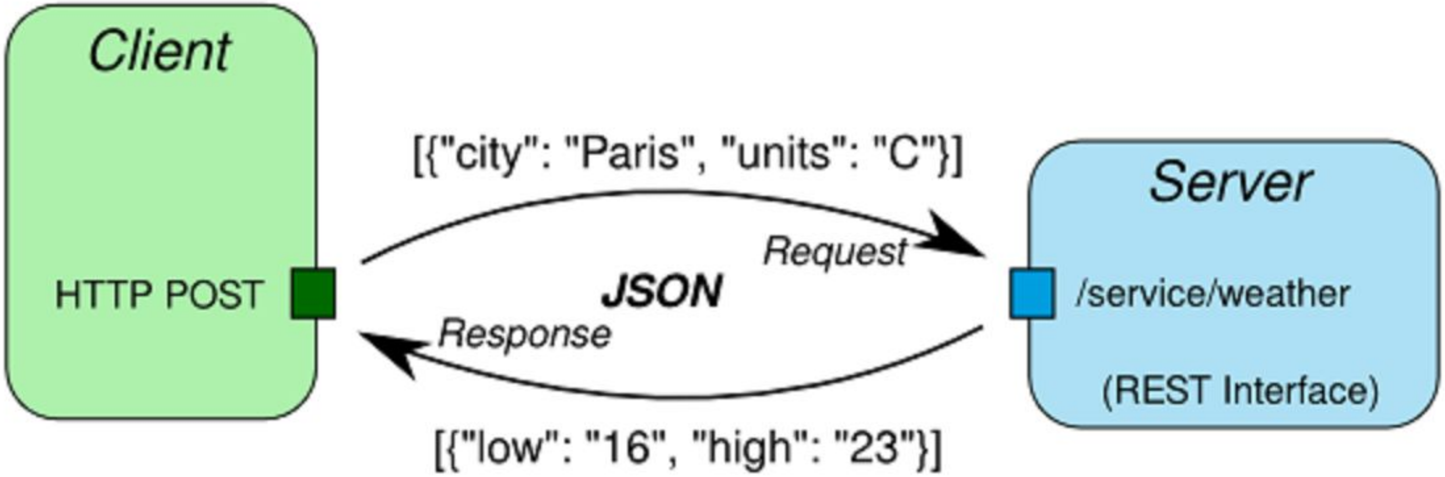
Our Rest API queries the database for what it needs

Our Clients, send HTTP Requests and wait for responses

Response: When the Rest API has what it needs, it sends back a response to the clients. This would typically be in JSON or XML format.


Przykład
REST API
Aplikacja mobilna
JSON / XML
GET
Aplikacja desktop
POST
JSON / XML
DELETE
Aplikacja WWW
JSON / XML
REST API
Baza danych

RESTful Web Service in Java



Client

HTTP POST

[{"city": "Paris", "units": "C"}]

Request

JSON

Response

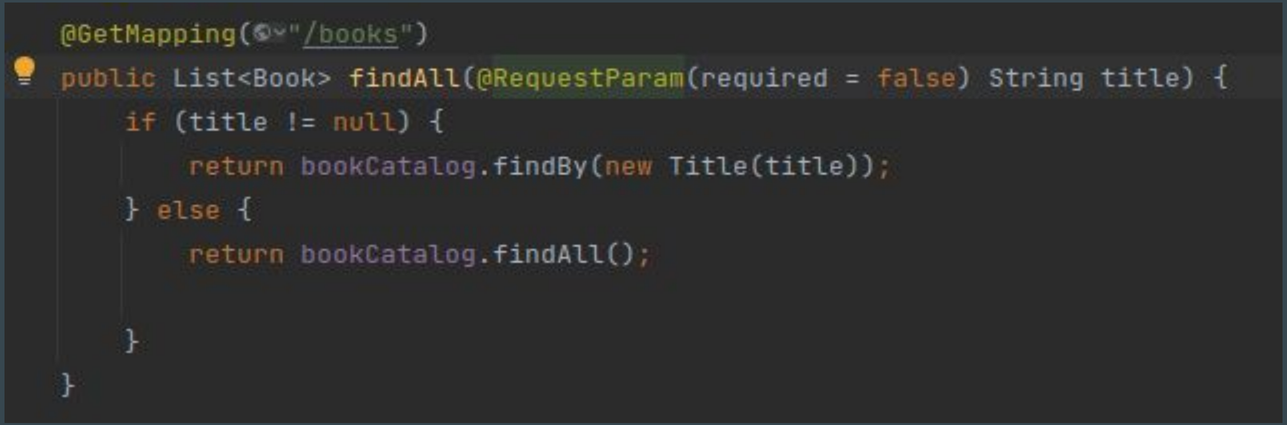
[{"low": "16", "high": "23"}]

Server

/service/weather

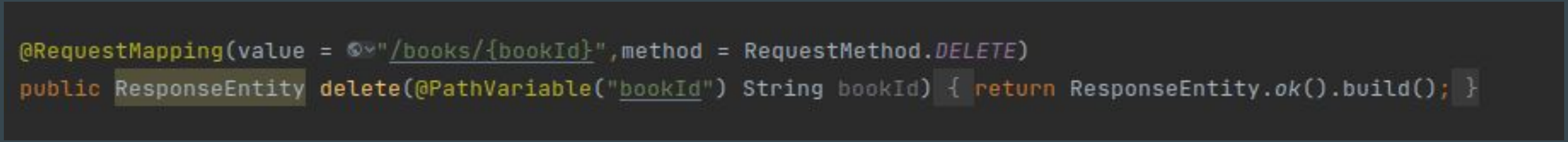
(REST Interface)

TODO: Projektujemy i implementujemy API



```java
@GetMapping("/books")
public List<Book> findAll(@RequestParam(required = false) String title) {
    if (title != null) {
        return bookCatalog.findBy(new Title(title));
    } else {
        return bookCatalog.findAll();

    }
}
```



```java
@RequestMapping(value = "/books/{bookId}",method = RequestMethod.DELETE)
public ResponseEntity delete(@PathVariable("bookId") String bookId) { return ResponseEntity.ok().build(); }
```

Bazy danych

Pogadajmy o architekturze ?

YOUR
FEEDBACK
MATTERS!

Ankieta

https://forms.office.com/Pages/ResponsePage.aspx?id=9m29-UnEH0irtTe9A2vrdm4UfNG_SqhKtw4Rkfq9QFFUOFIzSUhYQUROSFFZMlFKQU9DQVhMOTdKSi4u