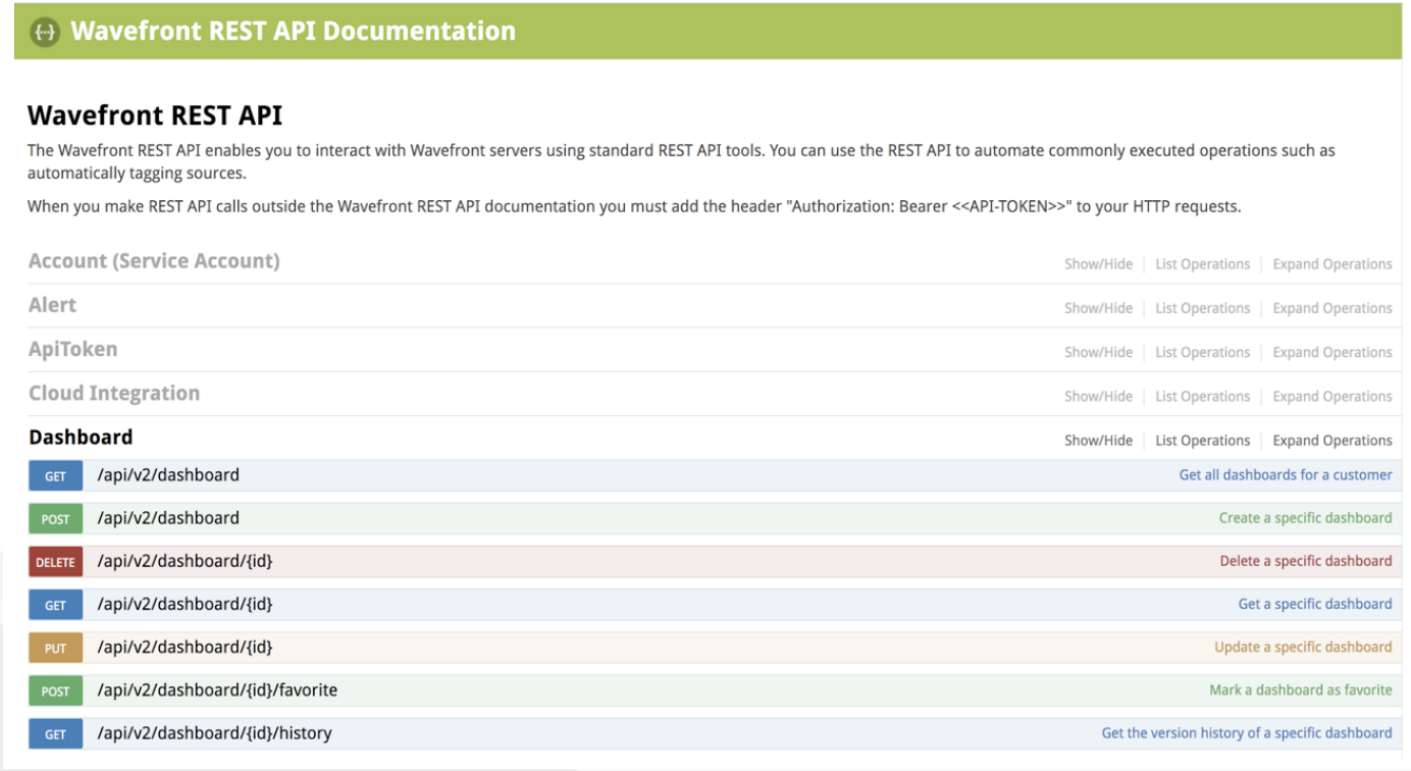
Swagger API Docs

Spring Boot Configure **Swagger** API Documentation

API Documentation

API documentation is technical content that documents the API. It includes instructions on how to effectively use and integrate the API. It also provides updates on the API's lifecycle such as new versions or retirement. Some aspects of API documentation can be generated automatically via Swagger or other documents.



Why use API Documentation?

Here example list api on some application.

Application will be growth based on adding new features top on it.

A large reason why API documentation is important is to increase API adoption. Comprehensive documentation on all of the functionality, how to effectively use and integrate, and updates on the API lifecycle improves the experience for those using your APIs.

APIs List

No.	API	CAS #	Pharmaceutical	Status
1	Mefenamic acid	61-68-7	non-steroidal anti-inflammatory drug (NSAID). Tradename: PONSTEL	US DMF filed. Inspected by US FDA for prior approval of API with no 483. ANDA approval is expected end of 2012. Commercial production presently for non-regulatory market
2	Talniflumate	66898-62-2	Anitinflammatory, mucoregulator. Tradename TALMAIN (Korea), LOMUCIN	No SFDA file, but in commercial production for non-regulatory market
3	Tolfenamic acid	13710-19-5	non-steroidal anti-inflammatory drug (NSAID). Tradename: CLOTAM	No SFDA file, but in commercial production for non-regulatory market
4	Calcium Phenylpyruvate	51828-93-4	Combination of amino acids for use in managing protein metabolism during renal insufficiency. Tradename: Ketosteril	Product licenses approved by SFDA. Final documentation underway. SFDA inspection expected in 2Q 2013 with approval thereafter.
5	α-Ketovaline Calcium	51828-94-5		
6	α-Ketoleucine Calcium	51828-95-6		
7	D,L-α-Ketoisoleucine Calcium	66872-75-1		
8	2-Hydroxy-4-(methylthio)butyric acid calcium salt	4857-44-7		
9	Fexofenadine HCl	15439-40-8	Antihistamine, Tradename: Allegra	In commercial production for non-regulatory market. US DMF in preparation will be filed in Q4, 2012
10	Venlafaxine	93413-69-5	Antidepressant Tradename: Effexor	In commercial production for non-regulatory market.
11	Edaravone	89-25-8	Stroke recovery Traded by Mitsubishi	Process development completed, documents will be filed with SFDA in Q2 2013
12	Doxapram	7081-53-0	Respiratory stimulant. Tradename: Dopram	Process development completed and US DMF will be filed with FDA in Q3 2013

API Documentation Tools

These tools have different UI but functionality its same for creating api documentation.

Why Swagger?

Because swagger have many functionality and have standard format like **OAS 3.0**

- OpenAPI Specification
- Open Source
- Greate API docs UI for frontend or mobile developer
- Used many big companies like Microsoft, National Geographic and many more.

OpenAPI Specification Definition

The **OpenAPI** Specification, formerly known as the Swagger Specification, is the world's standard for defining RESTful interfaces.

The OAS enables developers to design a technology-agnostic API interface that forms the basis of their API development and consumption.

OpenAPI Specifications Example

```yaml
openapi: 3.0.3
info:
  title: Swagger Petstore - OpenAPI 3.0
  version: 1.0.0
servers:
  - url: http://localhost:8080/api
paths:
  /pet:
    get:
      tags:
        - pet
      summary: Get list pet
      description: Get all pet data
      operationId: getPet
      responses:
        '200':
          description: Successful operation
          content:
            application/json:
              schema:
                type: array
                items:
                  $ref: '#/components/schemas/Pet'
```

Configure API Docs on Spring Boot

Preparation

We need prepare some prerequisities like below

- `springdoc-openapi-starter-webmvc-ui` dependencies
- Spring Boot project
- IDE
- Glass of coffee 😄

In Your `pom.xml`

We need to import libraries to use functionality swagger `OAS 3.0` on spring boot.

Spring boot starter validation to **validate bean configuration** that we create for swagger

Springdoc OpenAPI starter web mvc ui to generate automatic `API docs` and `SwaggerUI`

```xml
<dependency>
  <groupId>org.springframework.boot</groupId>
  <artifactId>spring-boot-starter-validation</artifactId>
</dependency>

<dependency>
  <groupId>org.springdoc</groupId>
  <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>
  <version>2.0.2</version>
</dependency>
```

Swagger Configuration

Create `Bean Configuration` by injecting OpenAPIConfiguration `@Configuration`

```
@Configuration
public class OpenAPIConfiguration {

  @Bean
  public OpenAPI openAPI() {
    return new OpenAPI()
            .info(new Info()
            .title("Rawlabs.ID")
            .description("Demo for Spring Boot")
            .version("1.0.0")
            .contact(new Contact()
                .name("Maverick")
                .url("https://piinalpin.com/")
                .email("any@email.com")));
    }

}
```

Spring Boot Configuration `Properties`

```
server.port=8080
server.servlet.context-path=/api
spring.application.name=demo-springboot

springdoc.packagesToScan=com.rawlabs.demospringboot.controller
springdoc.swagger-ui.enabled=true
springdoc.swagger-ui.path=/swagger-ui
springdoc.api-docs.path=/api-docs
```

The `application.properties` file will create configuration properties to enabling `SwaggerUI` and `API Docs` pre-defined path.

SwaggerUI will be generated to **http://localhost:8080/api/swagger-ui**
Base path for **servlet context** is `/api`

Define the Schema

The annotation `@Schema` will create OpenAPI **Schema** to including field on endpoint example payload.

- `description` : The description of the field
- `requiredMode` : Define that field is required or not required
- `example` : Provide the example value

```java
@Data
@Builder
@NoArgsConstructor
@AllArgsConstructor
@Entity
@Table(name = "book")
public class Book {

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    @Schema(
            description = "Generated ID",
            requiredMode = Schema.RequiredMode.REQUIRED,
            example = "1"
    )
    private Long id;

    @Column(name = "title", nullable = false)
    @Schema(
            description = "Book title",
            requiredMode = Schema.RequiredMode.REQUIRED,
            example = "Mastering Spring Boot"
    )
    private String title;

    @Column(name = "price", nullable = false)
    @Schema(
            description = "Book price",
            requiredMode = Schema.RequiredMode.REQUIRED,
            example = "120000"
    )
    private Integer price;

}
```

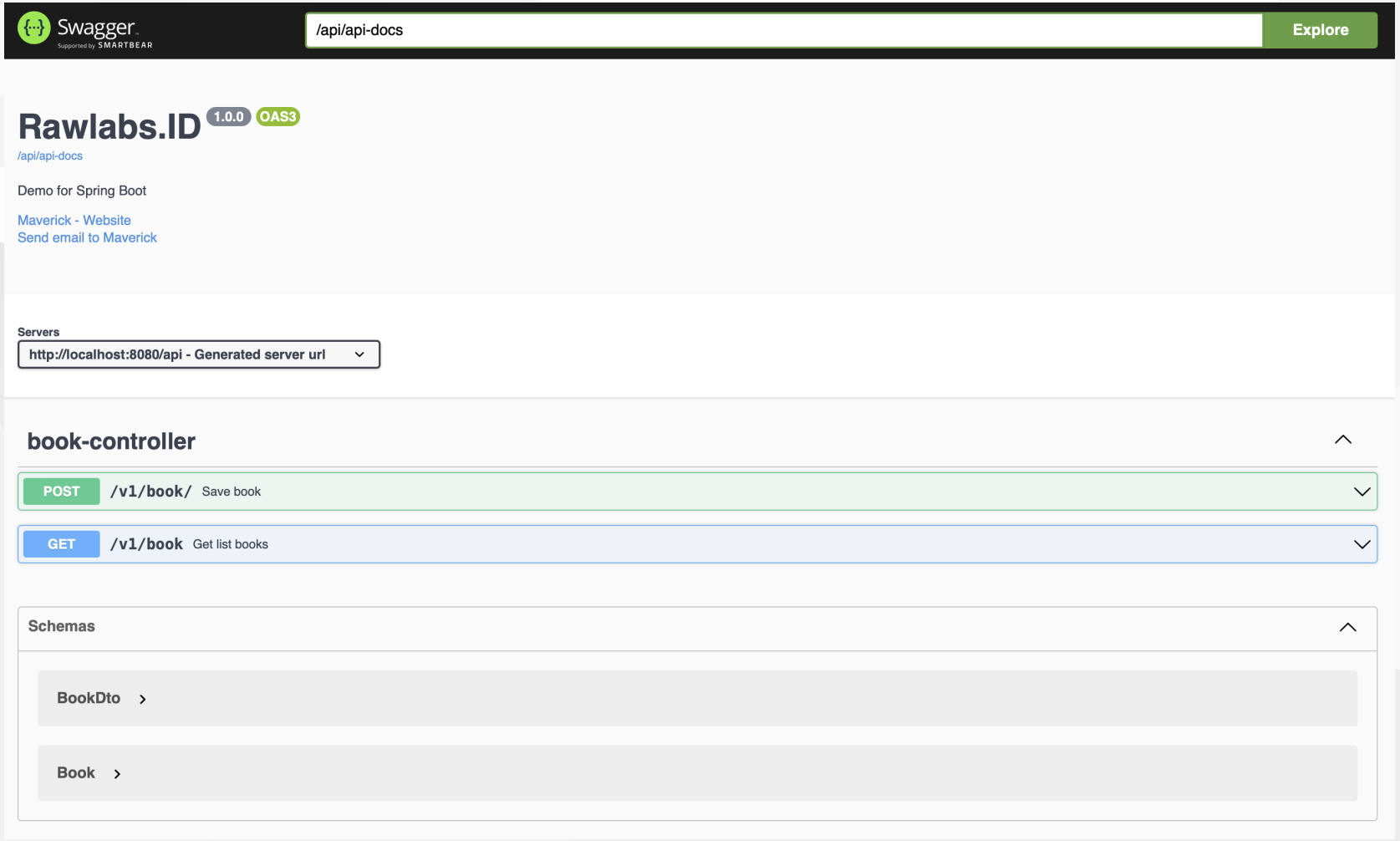
API Docs on Controller

- `@ApiResponses` to store list of response
- `@ApiResponse` to define individual response api docs.

```java
@GetMapping(value = "", produces = MediaType.APPLICATION_JSON_VALUE)
@Operation(summary = "Get list books")
@ApiResponses(value = {
        @ApiResponse(responseCode = "200", description = "Success")
})
public List<Book> getBooks() {
    return bookService.getBooks();
}

@PostMapping(value = "/", produces = MediaType.APPLICATION_JSON_VALUE)
@Operation(summary = "Save book")
@ApiResponses(value = {
        @ApiResponse(responseCode = "200", description = "Success")
})
public Book saveBook(@RequestBody BookDto request) {
    return bookService.save(request);
}
```

Generated SwaggerUI




THANK YOU