# **Spring Boot and Swagger - Documenting RESTful Services (REST API)**

## **Introduction**

REST does not specify a documentation standard or a contract like SOAP (WSDL). REST gives you the flexibility to choose your documentation format and approach. But that does not mean “No documentation”. It’s a misconception that REST means No documentation. You need to document your API.

## **Let’s get Started**

**How do you document your RESTful API?**

One option is to maintain documentation manually. But that gets outdated quickly.

Another option is to generate documentation from code. And that’s the approach we would discuss in this guide.

There are multiple approaches to documenting your RESTful API

* WADL
* RESTDocs
* Swagger or OpenDocs

Swagger has picked up momentum in the last couple of years and is now the most popular REST API documentation standard. We will use Swagger in this guide.

**Generating Swagger Documentation with Spring Boot**

We would need to add a couple of dependencies related to Swagger and configure a Docket to generate Swagger Documentation. We will also use Swagger UI to have a visual representation of the Documentation and execute Test Requests.

**Adding Swagger Dependencies**

Let’s add a couple of dependencies to our Swagger Project pom.xml.

|  |
| --- |
| <**dependency**>  <**groupId**>io.springfox</**groupId**>  <**artifactId**>springfox-boot-starter</**artifactId**>  <**version**>3.0.0-SNAPSHOT</**version**>  </**dependency**> |

As we are using a SNAPSHOT version, you would need to add a repository for jfrog-snapshots in your pom.xml.

|  |
| --- |
| <**repository**>  <**id**>jfrog-snapshots</**id**>  <**name**>JFROG Snapshots</**name**>  <**url**>http://oss.jfrog.org/artifactory/oss-snapshot-local</**url**>  <**snapshots**>  <**enabled**>true</**enabled**>  </**snapshots**>  </**repository**> |

### **Adding Swagger Spring Configuration Docket**

Let’s now add the Spring configuration needed to generate Swagger Documentation.

**/src/main/java/com/example/springboot/rest/example/swagger/SwaggerConfig.java**

|  |
| --- |
| @Configuration @EnableSwagger2WebMvc public class SwaggerConfig {   public static final Contact DEFAULT\_CONTACT = new Contact(  "David Vanidestine", "http://www.example.com", "example@gmail.com");    public static final ApiInfo DEFAULT\_API\_INFO = new ApiInfo(  "Awesome API Title", "Awesome API Description", "1.0",  "urn:tos", DEFAULT\_CONTACT,   "Apache 2.0", "http://www.apache.org/licenses/LICENSE-2.0",Arrays.asList());   private static final Set<String> DEFAULT\_PRODUCES\_AND\_CONSUMES =   new HashSet<String>(Arrays.asList("application/json",  "application/xml"));   @Bean  public Docket api() {  return new Docket(DocumentationType.SWAGGER\_2)  .apiInfo(DEFAULT\_API\_INFO)  .produces(DEFAULT\_PRODUCES\_AND\_CONSUMES)  .consumes(DEFAULT\_PRODUCES\_AND\_CONSUMES);  } } |

Notes

* @Configuration - This file contains Spring configuration.
* @EnableSwagger2WebMvc - Annotation to Enable Swagger Documentation on the API
* public static final Contact DEFAULT\_CONTACT - Has the contact information of the API. This will be exposed as part of the Swagger Documentation.
* public static final ApiInfo DEFAULT\_API\_INFO - Meta information about the API - Description, Licensing etc. This will be exposed as part of the Swagger Documentation.
* private static final Set<String> DEFAULT\_PRODUCES\_AND\_CONSUMES - What content types does your API support?
* public Docket api() { - Docket to decide what kind of APIs you would want to document. In this example, we are documenting all APIs. You can filter out APIs you do not want to document with Swagger.

### **Exposing meta API information using @SwaggerDefinition**

You can also expose meta API information using @SwaggerDefinition as shown below. The information in the class is self explanatory.

|  |
| --- |
| @SwaggerDefinition(  info = @Info(  description = "Awesome Resources",  version = "V12.0.12",  title = "Awesome Resource API",  contact = @Contact(  name = "David Vanidestine",   email = "David@example.com",   url = "http://www.example.com"  ),  license = @License(  name = "Apache 2.0",   url = "http://www.apache.org/licenses/LICENSE-2.0"  )  ),  consumes = {"application/json", "application/xml"},  produces = {"application/json", "application/xml"},  schemes = {SwaggerDefinition.Scheme.HTTP, SwaggerDefinition.Scheme.HTTPS},  externalDocs = @ExternalDocs(value = "Read This For Sure", url = "http://example.com") ) public interface ApiDocumentationConfig {  } |

### **Generated Swagger Documentation**

When you restart the application, you are all set to view the documentation that is generated.

Go to URL http://localhost:8080/v2/api-docs

At the top of the documentation is the Meta Information of the API

|  |
| --- |
| {  "swagger": "2.0",  "info": {  "description": "Awesome API Description",  "version": "1.0",  "title": "Awesome API Title",  "termsOfService": "urn:tos",  "contact": {  "name": "David Vanidestine",  "url": "http://www.example.com",  "email": "example@gmail.com"  },  "license": {  "name": "Apache 2.0",  "url": "http://www.apache.org/licenses/LICENSE-2.0"  }  },  "host": "localhost:8080",  "basePath": "/",  "tags": [  {  "name": "web-mvc-endpoint-handler-mapping",  "description": "Web Mvc Endpoint Handler Mapping"  },  {  "name": "student-resource",  "description": "Student Resource"  },  {  "name": "operation-handler",  "description": "Operation Handler"  },  {  "name": "basic-error-controller",  "description": "Basic Error Controller"  }  ],  "consumes": [  "application/xml",  "application/json"  ],  "produces": [  "application/xml",  "application/json"  ], |

The paths contain the details of the resources being exposed

* You can see the different request methods, a summary of each method and all details about each request and response

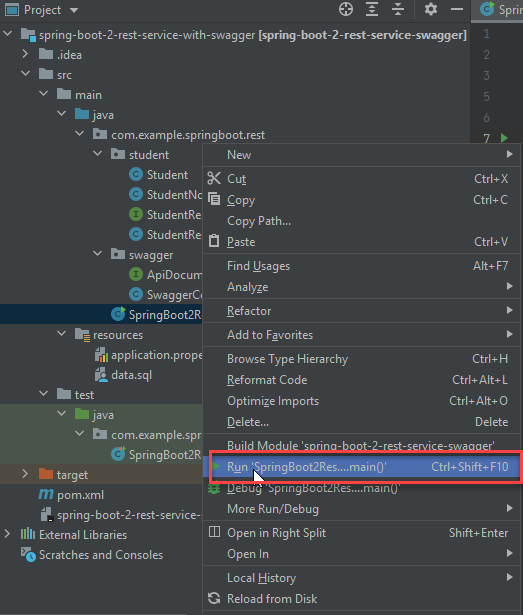
|  |
| --- |
| "paths": {  "/students": {  "get": {  "tags": [  "student-resource"  ],  "summary": "retrieveAllStudents",  "operationId": "retrieveAllStudentsUsingGET",  "consumes": [  "application/xml",  "application/json"  ],  "produces": [  "application/xml",  "application/json"  ],  "responses": {  "200": {  "description": "OK",  "schema": {  "type": "array",  "items": {  "$ref": "#/definitions/Student"  }  }  },  "401": {  "description": "Unauthorized"  },  "403": {  "description": "Forbidden"  },  "404": {  "description": "Not Found"  }  }  },  "post": {  "tags": [  "student-resource"  ],  "summary": "createStudent",  "operationId": "createStudentUsingPOST",  "consumes": [  "application/xml",  "application/json"  ],  "produces": [  "application/xml",  "application/json"  ],  "parameters": [  {  "in": "body",  "name": "student",  "description": "student",  "required": true,  "schema": {  "$ref": "#/definitions/Student"  }  }  ],  "responses": {  "200": {  "description": "OK",  "schema": {  "type": "object"  }  },  "201": {  "description": "Created"  },  "401": {  "description": "Unauthorized"  },  "403": {  "description": "Forbidden"  },  "404": {  "description": "Not Found"  }  }  }  },  "/students/{id}": {  "get": {  "tags": [  "student-resource"  ],  "summary": "Find student by id",  "description": "Also returns a link to retrieve all students with rel - all-students",  "operationId": "retrieveStudentUsingGET",  "consumes": [  "application/xml",  "application/json"  ],  "produces": [  "application/xml",  "application/json"  ],  "parameters": [  {  "name": "id",  "in": "path",  "description": "id",  "required": true,  "type": "integer",  "format": "int64"  }  ],  "responses": {  "200": {  "description": "OK",  "schema": {  "$ref": "#/definitions/Resource«Student»"  }  },  "401": {  "description": "Unauthorized"  },  "403": {  "description": "Forbidden"  },  "404": {  "description": "Not Found"  }  }  },  "put": {  "tags": [  "student-resource"  ],  "summary": "updateStudent",  "operationId": "updateStudentUsingPUT",  "consumes": [  "application/xml",  "application/json"  ],  "produces": [  "application/xml",  "application/json"  ],  "parameters": [  {  "in": "body",  "name": "student",  "description": "student",  "required": true,  "schema": {  "$ref": "#/definitions/Student"  }  },  {  "name": "id",  "in": "path",  "description": "id",  "required": true,  "type": "integer",  "format": "int64"  }  ],  "responses": {  "200": {  "description": "OK",  "schema": {  "type": "object"  }  },  "201": {  "description": "Created"  },  "401": {  "description": "Unauthorized"  },  "403": {  "description": "Forbidden"  },  "404": {  "description": "Not Found"  }  }  },  "delete": {  "tags": [  "student-resource"  ],  "summary": "deleteStudent",  "operationId": "deleteStudentUsingDELETE",  "consumes": [  "application/xml",  "application/json"  ],  "produces": [  "application/xml",  "application/json"  ],  "parameters": [  {  "name": "id",  "in": "path",  "description": "id",  "required": true,  "type": "integer",  "format": "int64"  }  ],  "responses": {  "200": {  "description": "OK"  },  "204": {  "description": "No Content"  },  "401": {  "description": "Unauthorized"  },  "403": {  "description": "Forbidden"  }  }  }  }  }, |

Definitions contain the detailed structure of the elements used in Request and Responses above.

|  |
| --- |
| "definitions": {  "Resource«Student»": {  "type": "object",  "properties": {  "id": {  "type": "integer",  "format": "int64"  },  "links": {  "type": "array",  "items": {  "$ref": "#/definitions/Link"  }  },  "name": {  "type": "string",  "description": "Name should have atleast 2 characters"  },  "passportNumber": {  "type": "string"  }  }  },  "Map«string,Link»": {  "type": "object",  "additionalProperties": {  "$ref": "#/definitions/Link"  }  },  "Student": {  "type": "object",  "properties": {  "id": {  "type": "integer",  "format": "int64"  },  "name": {  "type": "string",  "description": "Name should have atleast 2 characters"  },  "passportNumber": {  "type": "string"  }  },  "description": "All details about the student. "  },  "Link": {  "type": "object",  "properties": {  "href": {  "type": "string"  },  "templated": {  "type": "boolean"  }  }  }  } } |

**Run Application**

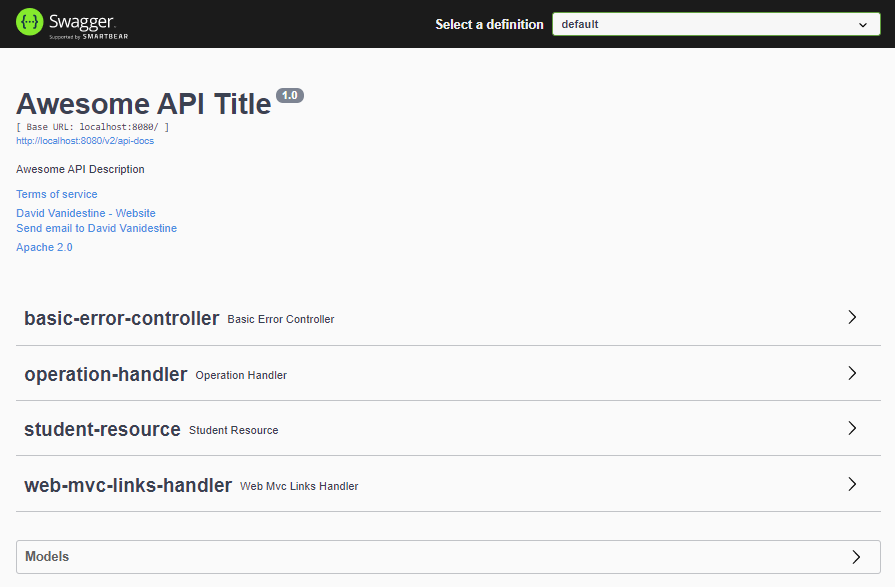
Just right click on **SpringBoot2RestServiceApplication.java** then click on Run



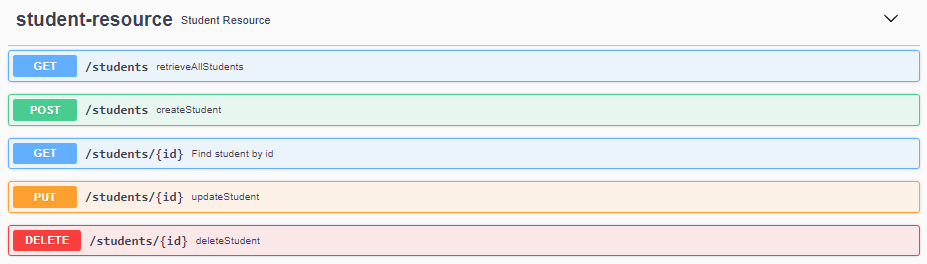
**Launching Swagger UI**

You can also use the Swagger UI available at http://localhost:8080/swagger-ui/index.html.

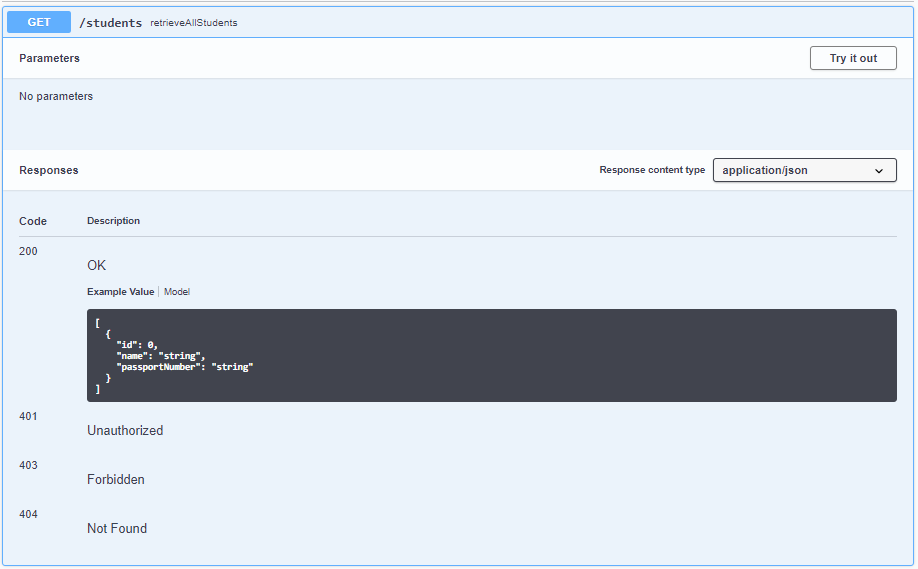
Below screenshot shows the Home Page of Swagger UI. It shows a list of all the resources that are exposed.



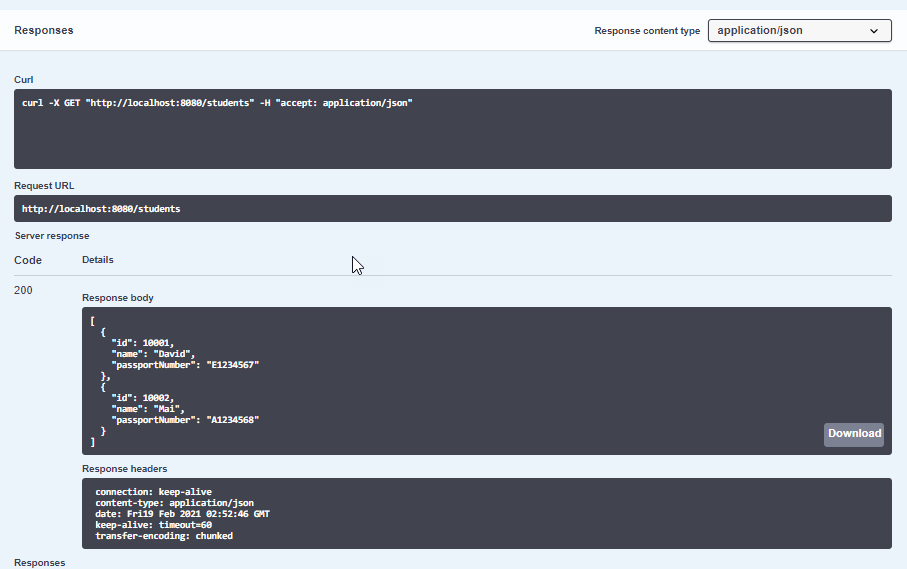
Choosing the Student resource takes you to details of the resource. It shows all the request methods that can be used with a Resource.



You can also see the details for a Specific Request Method.



You can use the ‘Try it out’ button to execute a request and see the response.



### **Customizing Swagger Documentation with Annotations**

You can add notes on the resource method to add more documentation

|  |
| --- |
| @GetMapping("/students/{id}")  @ApiOperation(value = "Find student by id",  notes = "Also returns a link to retrieve all students with rel - all-students")  public Resource<Student> retrieveStudent(@PathVariable long id) { |

Also supported is enhancing the documentation on the Request and Response Beans.

|  |
| --- |
| @Entity @ApiModel(description="All details about the student. ") public class Student {    @ApiModelProperty(notes="Name should have atleast 2 characters")  @Size(min=2, message="Name should have atleast 2 characters")  private String name; |

**Voila!!** We have successfully completed this Exercise.