

Cu Swagger putem testa endpoint-urile fara Postman si vedea si info despre ele.

**Dependenta**

<dependency>  
 <groupId>org.springdoc</groupId>  
 <artifactId>springdoc-openapi-starter-webmvc-ui</artifactId>  
 <version>2.1.0</version>  
</dependency>

Dependenta data va adauga si Swagger si tot

**General Config**

Ar fi bine sa facem o clasa separata si sa o marcam cu @Configuration

Folosim apoi @OpenAPIDefinition

@Configuration  
@OpenAPIDefinition(  
 info = @Info(  
 title = "REST API Documentation",  
 description = "REST API Documentation",  
 version = "V1.0.0",  
 contact = @Contact(  
 name = "Endava",  
 url = "endava.com",  
 email = "endava@endava.com"  
 ),  
 license = @License(  
 name = "My License",  
 url = "Endava.com/license"  
 )  
 ),  
 externalDocs = @ExternalDocumentation(  
 url = "test.com",  
 description = "Another documentation"  
 ),  
 servers = {@Server(  
 url = "google.com",  
 description = "Another external server"  
 ),  
 @Server(  
 url = "http://wwww.localhost:3306",  
 description = "Local server"  
 )  
 }  
)  
public class OpenAPI {  
}

**Customize Swagger with annotations**

@Tag – adauga detalii despre controller

@Tag(  
 name = "User REST API",  
 description = "ADD/UPDATE/DELETE/GET user(s)"  
)  
  
@RestController  
@RequestMapping("users")  
public class Controller {

@Operation – descrie ce face un endpoit

@ApiResponse – status returneaza un endpoint

@Operation(  
 summary = "GET all users",  
 description = "GET a list of all users"  
)  
@ApiResponse(  
 responseCode = "200",  
 description = "HTTP 200 SUCCESS"  
)  
  
@GetMapping  
public List<UserDto> userDtoList() {  
 return userDtoList;  
}

**Customize Swagger Models**

@Schema(  
 name = "UserDTO schema",  
 description = "UserDTO description",  
)  
public class UserDto {  
 private int id;  
  
 @NotBlank  
 @Size(min = 3, max = 15)  
 @Schema(  
 name = "username",  
 description = "username",  
 example = "testUser"  
 )  
 private String username;  
  
 @NotBlank  
 @Size(min = 8, max = 20)  
 @Schema(  
 name = "password",  
 description = "password",  
 example = "test122"  
 )  
 private String password;  
  
 @NotBlank  
 @Email  
 @Schema(  
 name = "email",  
 description = "email",  
 example = "test@gmail.com"  
 )  
 private String email;  
}

@DeleteMapping("{id}")  
@Schema(  
 example = "localhost:3306/1",  
 description = "Delete a user by id"  
)  
@ApiResponse(  
 content = @Content(  
 examples = @ExampleObject(  
 value = "Success"  
 )  
 )  
)  
public String deleteUser(@PathVariable int id) {  
 return "Success";  
}