295 - Backend

OpenAPI & Swagger Tools

|  |  |
| --- | --- |
| Richtzeit  3h | Sozialform  Einzelarbeit |

## Einführung

Für unsere REST-API wollen wir ebenfalls eine Dokumentation zur Verfügung stellen.

## Dependency springdoc-openapi-ui hinzufügen

Folgenden dependencies-Block sollte bereits in deinem build.gralde sein. Es kann sein, dass du eine neuere Version hast.

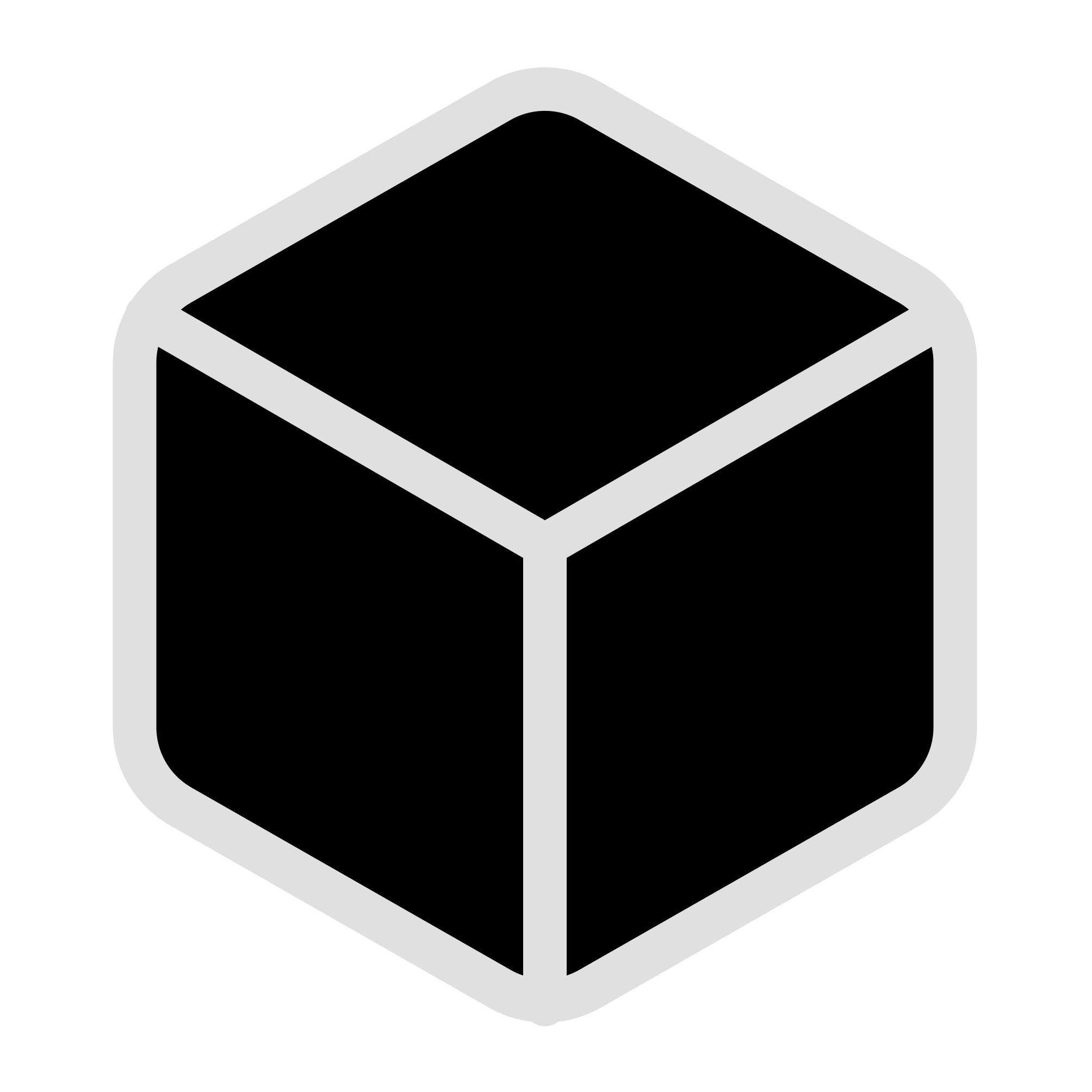
implementation 'org.springdoc:springdoc-openapi-ui:1.6.10'

Diese bewirkt, dass unter folgendem Link mit GET eine Ressource zur Verfügung steht:

<http://localhost:8080/v3/api-docs/>

Nämlich erhältst du über diese Adresse eine Beschreibung deiner REST-API im JSON-Format. Die ganze JSON-Struktur und die Namen der Attribute sind vom OpenAPI-Standard vorgegeben. Jedoch kann es für eine:n Entwickler:in relativ mühsam und zeitaufwändig sein, diese JSON-Beschreibung zu lesen und zu interpretieren. Eine benutzerfreundlichere Darstellung dieser Informationen findest du unter diesem Link: <http://localhost:8080/swagger-ui.html>

Wie du siehst, ist die grafische Darstellung so umgesetzt wie bei <https://petstore.swagger.io>. Es fehlen jedoch noch aussagekräftige Beschreibungen der einzelnen Parameter, Rückgabewerte u.s.w. In den folgenden Abschnitten zeigen wir dir anhand des TagController, wie du dies umsetzen kannst.

Was weiter benötigt wird und bereits im Template ist, sind die OpenAPIConfiguration.java und die @SecurityScheme-Annotation in der SecurityConfiguration.java. Diese werden benötigt, um den Info-Teil der Spezifikation zu generieren. Wir gehen hier nicht genauer darauf ein, du darfst jedoch gerne etwas damit rumspielen.

## Dokumentation anhand des PersonController

Die gesamte Dokumentation der einzelnen Controller-Methoden geschieht über Java-Annotationen. Diese Anleitung orientiert sich am Code des PersonController, dein Auftrag ist es, das analog für TagController und ItemController zu tun.

@GetMapping  
@Operation(summary = "Get all users.")  
@ApiResponses(value = {  
 @ApiResponse(responseCode = "200", description = "Users found",  
 content = @Content(schema = @Schema(implementation = PersonResponseDTO.class)))  
})

Das ist die ganze Dokumentation für das findAll().

Mit @Operation(summary = "Text…") erstellen wir die Beschreibung, die über Swagger UI sichtbar ist. Mit @ApiResponses(value = {…}) beschreiben wir die verschiedenen möglichen HTTP-Responses. responseCode und description sollten selbsterklärend sein. Mehrere @ApiResponse-Annotationen können kommasepariert der @ApiResponses-Annotation mitgegeben werden.

Mit content = @Content(…) beschreiben wir ein mögliches Antwort-Format. Hier könnte auch ein MediaType beschrieben werden. Mit @Schema(implementation = PersonResponseDTO.class) wird dem Benutzer in der Swagger UI-Dokumentation ein Beispiel gegeben, wie die Ressource in der http-Response in JSON dargestellt wird. Zudem werden dem Benutzer die verwendeten Datentypen und allfällige Validierung angezeigt:

|  |  |
| --- | --- |
|  |  |

Hat eine Response kein body, so muss dies mit content = @Content in der @ApiResponse angegeben werden.

Methodenparameter können wir wie folgt dokumentieren:

public ResponseEntity<?> findById(@Parameter(description = "Id of user to get") @PathVariable Integer id) {

Hierbei spielt es keine Rolle, ob dies eine @PathVariable, ein @RequestBody oder ein @RequestParam ist.

Nun solltest du alle Werkzeuge kennen, um deine API sauber zu dokumentieren.