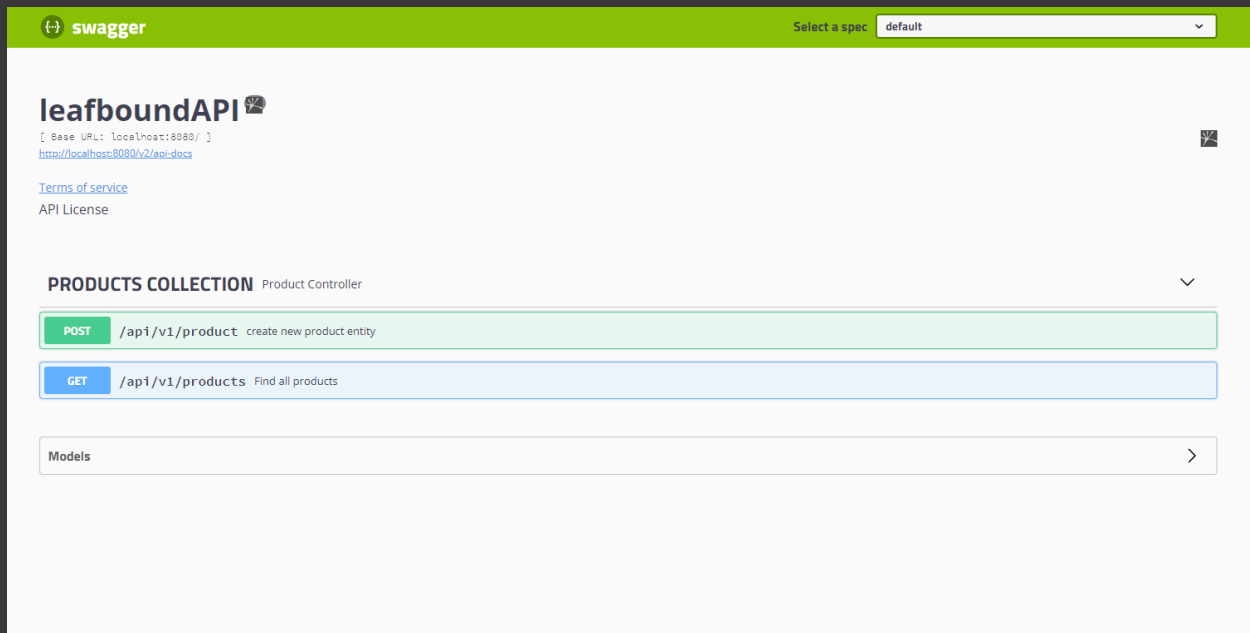


Product Collections Endpoint Testing

Endpoints to test:



The image shows the Swagger UI for the leafboundAPI. The top bar is green with the Swagger logo and a dropdown menu for selecting a spec, currently set to 'default'. The main header area displays the API name 'leafboundAPI' with a small icon, the base URL '[Base URL: localhost:8080/]', and a link to the API docs at 'http://localhost:8080/v2/api-docs'. Below this are links for 'Terms of service' and 'API License'. The 'PRODUCTS COLLECTION' section is expanded, showing two endpoints: a POST endpoint for '/api/v1/product' with the description 'create new product entity', and a GET endpoint for '/api/v1/products' with the description 'Find all products'. A 'Models' section is also visible at the bottom of the collection, with a right-pointing arrow.

swagger Select a spec default

leafboundAPI
[Base URL: localhost:8080/]
<http://localhost:8080/v2/api-docs>
[Terms of service](#)
API License

PRODUCTS COLLECTION Product Controller

POST /api/v1/product create new product entity

GET /api/v1/products Find all products

Models

Passing “Create New Product Entity” Post – Code 200

PRODUCTS COLLECTION Product Controller

POST /api/v1/product create new product entity

Parameters

Try it out

Name	Description
product <small>required</small> <small>(body)</small>	product

Example Value

Model

```
{  "author": "Test Author",  "description": "Test Description",  "edition": "Test Edition 1",  "genre": "Test Genre",  "id": 1,  "isbn": "Test ISBN 7362527439",  "language": "Test Language EN",  "list_price": 8.99,  "published_date": "2022-06-17",  "publisher": "Test Publisher",  "title": "Test Title"}
```

Parameter content type
application/json

Responses

Response content type */*

Curl

```
curl -X POST "http://localhost:8080/api/v1/product" -H "accept: */*" -H "Content-Type: application/json" -d '{"author": "Test Author", "description": "Test Description", "edition": "Test Edition 1", "genre": "Test Genre", "id": 1, "isbn": "Test ISBN 7362527439", "language": "Test Language EN", "list_price": 8.99, "published_date": "2022-06-17", "publisher": "Test Publisher", "title": "Test Title"}'
```

Request URL

```
http://localhost:8080/api/v1/product
```

Server response

Code	Details
200	<div><div>Response body</div><pre>{ "id": 1, "author": "Test Author", "publisher": "Test Publisher", "isbn": "Test ISBN 7362527439", "genre": "Test Genre", "title": "Test Title", "language": "Test Language EN", "published_date": "2022-06-17", "edition": "Test Edition 1", "description": "Test Description", "list_price": 8.99}</pre><div>Download</div></div> <div><div>Response headers</div><pre>connection: keep-alive content-type: application/json date: Fri, 17 Jun 2022 16:35:13 GMT keep-alive: timeout=60 transfer-encoding: chunked vary: Origin, Access-Control-Request-Method, Access-Control-Request-Headers</pre></div>

Responses

Code	Description
200	OK

Example of Failing “Create New Product Entity” Post– Code 400

PRODUCTS COLLECTION Product Controller

POST /api/v1/product create new product entity

Parameters

Try it out

Name	Description
product <small>required</small>	product
(body)	<div><div>Example Value</div><div>Model</div></div> <pre>{ "author": "Test Author", "description": "Test Description", "edition": "Test Edition 1", "genre": "Test Genre", "id": 1, "isbn": "Test ISBN 7362527439", "language": "Test Language EN", "list_price": 8.99, "published_date": "2022-06-1734", "publisher": "Test Publisher", "title": "Test Title" }</pre> <div>Parameter content type application/json</div>

Responses

Response content type */*

Curl

```
curl -X POST "http://localhost:8080/api/v1/product" -H "accept: */*" -H "Content-Type: application/json" -d '{ "author": "Test Author", "description": "Test Description", "edition": "Test Edition 1", "genre": "Test Genre", "id": 1, "isbn": "Test ISBN 7362527439", "language": "Test Language EN", "list_price": 8.99, "published_date": "2022-06-1734", "publisher": "Test Publisher", "title": "Test Title"}'
```

Request URL

http://localhost:8080/api/v1/product

Server response

Code	Details
400	<div>Error: <i>Undocumented</i> Response body<pre>{ "timestamp": "2022-06-17T16:47:11.634+00:00", "status": 400, "error": "Bad Request", "trace": "org.springframework.http.converter.HttpMessageNotReadableException: JSON parse error: Cannot deserialize value of type 'java.time.LocalDate' from String '2022-06-1734': Failed to deserialize java.time.LocalDate: (java.time.format.DateTimeParseException) Text '2022-06-1734' could not be parsed, unparsed text found at index 10; nested exception is com.fasterxml.jackson.databind.exc.InvalidFormatException: Cannot deserialize value of type 'java.time.LocalDate' from String '2022-06-1734': Failed to deserialize java.time.LocalDate: (java.time.format.DateTimeParseException) Text '2022-06-1734' could not be parsed, unparsed text found at index 10\nat [Source: (org.springframework.util.StreamUtils\$NonClosingInputStream); line: 19, column: 21] (through reference chain: com.testbound.models.Product[\"published_date\"])\r\n\tat org.springframework.http.converter.json.AbstractJackson2HttpMessageConverter.readJavaType(AbstractJackson2HttpMessageConverter.java:391)\r\n\tat org.springframework.http.converter.json.AbstractJackson2HttpMessageConverter.read(AbstractJackson2HttpMessageConverter.java:343)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.AbstractMessageConverterMethodArgumentResolver.readWithMessageConverters(AbstractMessageConverterMethodArgumentResolver.java:185)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingMethodProcessor.readWithMessageConverters(RequestResponseBodyMethodProcessor.java:160)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingMethodProcessor.resolveArgument(RequestResponseBodyMethodProcessor.java:133)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingMethodProcessor.resolveArgument(RequestResponseBodyMethodProcessor.java:122)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingMethodProcessor.resolveArgument(RequestResponseBodyMethodProcessor.java:179)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingMethodProcessor.resolveArgument(RequestResponseBodyMethodProcessor.java:146)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingMethodProcessor.invokeAndHandle(ServletInvocableHandlerMethod.java:117)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.invokeHandlerMethod(RequestMappingHandlerAdapter.java:895)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.RequestMappingHandlerAdapter.handleInternal(RequestMappingHandlerAdapter.java:888)\r\n\tat org.springframework.web.servlet.mvc.method.annotation.AbstractHandlerMethodAdapter.handle(AbstractHandlerMethodAdapter.java:87)\r\n\tat org.springframework.web.servlet.DispatcherServlet.doDispatch(DispatcherServlet.java:1867)\r\n\tat org.springframework.web.servlet.DispatcherServlet.doService(DispatcherServlet.java:1963)\r\n\tat org.springframework.web.servlet.FrameworkServlet.processRequest(FrameworkServlet.java:1096)\r\n\tat org.springframework.web.servlet.FrameworkServlet.doPost(FrameworkServlet.java:989)\r\n\tat javax.servlet.http.HttpServlet.service(HttpServlet.java:681)\r\n\tat</pre></div>

Response headers

```
connection: close
content-type: application/json
date: Fri, 17 Jun 2022 16:47:11 GMT
transfer-encoding: chunked
vary: Origin, Access-Control-Request-Method, Access-Control-Request-Headers
```

Download

Passing “Find All Products” Get – Code 200

GET

/api/v1/products

Find all products

Parameters

No parameters

Cancel

Execute

Clear

Responses

Response content type */*

Curl

curl -X GET "http://localhost:8080/api/v1/products" -H "accept: */*"

Request URL

http://localhost:8080/api/v1/products

Server response

Code

Details

200

Response body

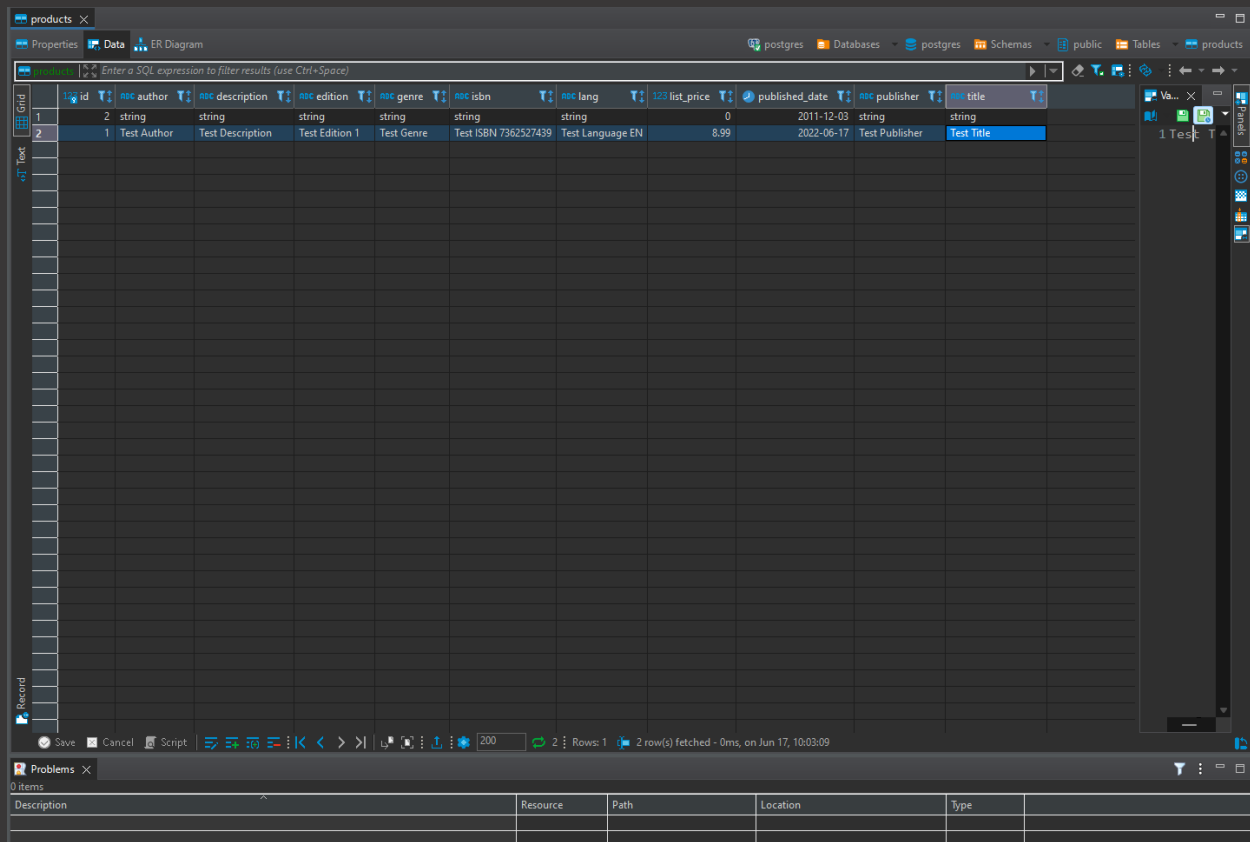
```
{
  "id": 5,
  "author": "string",
  "publisher": "string",
  "isbn": "string",
  "genre": "string",
  "title": "efdsf",
  "language": "string",
  "published_date": "2011-12-03",
  "edition": "string",
  "description": "string",
  "list_price": 0
},
{
  "id": 1,
  "author": "Test Author",
  "publisher": "Test Publisher",
  "isbn": "Test ISBN 7362527439",
  "genre": "Test Genre",
  "title": "Test Title",
  "language": "Test Language EN",
  "published_date": "2022-06-17",
  "edition": "Test Edition 1",
  "description": "Test Description",
  "list_price": 8.99
}
```

Download

Response headers

```
connection: keep-alive
content-type: application/json
date: Fri, 17 Jun 2022 16:50:56 GMT
keep-alive: timeout=60
transfer-encoding: chunked
vary: Origin, Access-Control-Request-Method, Access-Control-Request-Headers
```

Example of Data being communicated with Database



The screenshot displays a database management interface with a table named 'products'. The table has the following columns: id, author, description, edition, genre, isbn, lang, list_price, published_date, publisher, and title. The first row contains the values: 1, Test Author, Test Description, Test Edition 1, Test Genre, Test ISBN 7362527439, Test Language EN, 8.99, 2011-12-03, Test Publisher, and Test Title. The second row contains the values: 2, string, string, string, string, string, string, 0, 2022-06-17, string, and string. The interface also shows a sidebar with 'Gold' and 'Record' tabs, a bottom status bar indicating '2 rows: 1' and '2 row(s) fetched - 0ms, on Jun 17, 10:03:09', and a 'Problems' tab at the bottom.

id	author	description	edition	genre	isbn	lang	list_price	published_date	publisher	title
1	Test Author	Test Description	Test Edition 1	Test Genre	Test ISBN 7362527439	Test Language EN	8.99	2011-12-03	Test Publisher	Test Title
2	string	string	string	string	string	string	0	2022-06-17	string	string