

INTERACTIVE API DOCS WITH SWAGGER

ABOUT ME

- Katja Bregenzer
- Software developer at Lime Technologies
- Grew up in southern Germany and moved to Sweden in 2018
- 4+ years experiences with web development (javascript, html, css, python, java)

ABOUT LIME TECHNOLOGIES

- nearly 270 co-workers, over 60 000 users and 25 years experience in the CRM industry
- one of the leading CRM providers in the Nordic's
- offices in Lund, Stockholm, Gothenburg, Helsinki, Copenhagen and Oslo

NORDIC TRAINEE PROGRAM

- 1 year program with a mentor on your side
- kickoff with 4 weeks of education in Lund
- roles: developer, account manager, application consultant, technical project manager
- starting in January and August

lilimeo

CRM with a twist

WHY CRM?

- connecting the dots: who was in contact with whom about what
- enable the best possible customer support
- support for recurring workflows

LIME CRM

- configurations & plugins
 - **custom database structures changeable at runtime**
- how can you get the current database structure?

LIMETYPE API

LIMETYPES

- custom database tables
 - custom name
 - custom columns
- labels to connect a table to known entities/
concepts
- system columns as common factor

```
{  
  "name": "university",  
  "localname": {  
    "singular": "University",  
    "plural": "Universities"  
  },  
  "sort_order": 3,  
  "label": "company",  
  "_links": {}  
}
```

ENDPOINTS

- `api/v1/limetype/`
- `api/v1/limetype/university`
- `api/v1/limetype/university/?
embed=properties`

SWAGGER UI

*Swagger UI allows anyone [...] to **visualize and interact with the API's resources** without having any of the implementation logic in place. It's automatically generated from your **OpenAPI Specification**, with the visual documentation [...].*

Swagger UI

OPENAPI SPECIFICATION

- specification to describe a REST API
 - available endpoints
 - operation parameters
 - authentication
 - Contact information, license, terms of use...
- in YAML or JSON

LIMETYPE API WITH SWAGGER UI

- dynamic endpoints in OpenAPI specification
- use [Jinja template](#) for serverside rendered yaml specification

```
{% for lt in limetypes %}  
  /{{ lt.name }}/:  
    get:  
      summary: Retrieve the {{ lt.localname.singular | lo  
      description: Retrieves the schema for the the {{ lt  
      tags:  
        - Specific limetypes  
      parameters:  
        [...]  
      responses:  
        [...]  
{% endfor %}
```

```
/university/:
```

```
  get:
```

```
    summary: Retrieve the university schema.
```

```
    description: Retrieves the schema for the the unive
```

```
    tags:
```

```
      - Specific limetypes
```

```
    parameters:
```

```
      [...]
```

```
    responses:
```

```
      [...]
```


DEMO TIME


















Designer

+ New Table


Save


Find Table

Tables

-  **university** universities
university [↗ Company](#)
-  **Person** Persons
person [↗ Person](#)
-  **Deal** Deals
deal [↗ Project](#)
-  **To-do** To-dos
todo [↗ ToDo](#)
-  **History** History
history [↗ History](#)
-  **Document** Documents
document [↗ Document](#)
-  **Marketing activity** Marketing activities
campaign [↗ Campaign](#)
-  **Participant** Participants
participant [↗ None](#)
-  **Office** Offices
office [↗ None](#)
-  **InfoTile** InfoTiles
infotiles [↗ None](#)
-  **Solution improvement** Solution improvements
solutionimprovement [↗ None](#)
-  **Target** Target
target [↗ None](#)
-  **Localization** Localizations
localize [↗ None](#)
-  **Coworker** Coworkers
coworker [↗ User](#)
-  **Product** Products
product [↗ Product](#)
-  **Project activity** Project activities
projectactivities [↗ Project](#)
-  **student** students
student [↗ Person](#)

Designer

 New Table

 Overview

student ▼

 Save

 Reload




 Size

 Misc



Find Field

student


Save and close















university



name



 Add Relation ▼



Specific limetypes

Show/Hide | List Operations | Expand Operations

GET

/university/

Retrieve the university schema.

Implementation Notes

Retrieves the schema for the the university limetype.

Response Class (Status 200)

A limetype schema.

Model

Example Value

```
{
  "name": {},
  "label": "company",
  "localname": {
    "singular": "string",
    "plural": "string"
  },
  "sort_order": 0
}
```

Response Content Type

application/hal+json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
_embed	<div></div>	What information to embed in the response, if any.	query	string
Accept-Language	<div>en</div>		header	string

Response Messages

HTTP Status Code	Reason	Response Model	Headers
default	Unexpected error	<div><div>Model</div><div>Example Value</div></div> <div><pre>{ "error": "string" }</pre></div>	

Try it out!

universities

Show/Hide | List Operations | Expand Operations

GET	/university/	Retrieve universities
POST	/university/	Create a new university.
DELETE	/university/{id}/	Delete a university.
GET	/university/{id}/	Retrieve a university.
PUT	/university/{id}/	Update an existing university.
GET	/university/{id}/coworker/	Retrieve a related coworker.
GET	/university/{id}/deal/	Retrieve related deals.
GET	/university/{id}/history/	Retrieve related history.
GET	/university/{id}/document/	Retrieve related documents.
GET	/university/{id}/todo/	Retrieve related to-dos.
GET	/university/{id}/student/	Retrieve related students.

POST

/student/

Create a new student.

Implementation Notes

Create a single student object.

Response Class (Status 201)

A JSON representation of the created object.

Model

Example Value

```
{
  "id": 0,
  "name": "string",
  "university": 0
}
```

Response Content Type

application/hal+json

Parameters

Parameter	Value	Description	Parameter Type	Data Type
student	<pre>{ "name": "Johan", "university": 1041 }</pre>	The student data to insert.	body	<div><div>Model</div><div>Example Value</div></div> <div><pre>{ "id": 0, "name": "string", "university": 0 }</pre></div>

Parameter content type: application/json

Response Messages

HTTP Status Code	Reason	Response Model	Headers
400	Bad request.	<div><div>Model</div><div>Example Value</div></div> <div><pre>{ "limeobject_id": 0, "limetype": "string", "errors": [{ "error_message": "string", "limeproperty": "string", "error_code": "string" }] }</pre></div>	

Try it out!

THANKS FOR YOUR ATTENTION

- Nordic Trainee Program: <https://www.lime-technologies.se/trainee/>
- 1 year program with a mentor on your side
- kickoff with 4 weeks of education in Lund
- roles: developer, account manager, application consultant, technical project manager
- starting in January and August