Veebiteenuste ja hajussüsteemide arendus

Loeng 6: Veebiteenuste standardid

Pelle Jakovits
jakovits@ut.ee

March 2025

Sisukord

- Veebiteenuste standardid
  - Web Services Description Language (WSDL)
    - UDDI, WSIL
  - OpenAPI

Veebiteenused

- Tarkvara, mis pakub juurdepääsu Interneti-ressurssidele kasutades standardseid veebiprotokolle (HTTP, HTTPS)

"Loosely coupled, standard-based reusable software components that semantically encapsulate discrete functionality and are distributed and programmatically accessible over standard Internet protocols"
[Sleeper, 2001]

- **Google Translate** – Teksti saatmine tõlkimiseks
- **Shrinkpictures.com** –
  Piltide saatmine nende väiksemaks tegemiseks
- **Reddit** –
  Sõnumite saatmine selleks, et neid foorumisse postitada

Veebiteenuste standarid

- Veebiteenustes on suhtlus üles ehitatud standarditele tavalistele veebi standarditele (nt XML, WSDL, HTTP).
- Veebiteenuste standarid määravad kuidas:
  - Kirjeldada veebiteenuseid, ressursse ja operatsioone
  - Avaldada, otsida ja avastada veebiteenuseid
- Standardite eesmärk on tagada, et neid jälgides ehitatud veebiteenused:
  - on võimelised koostööd tegema,
  - on platvormist sõltumatud
  - on skaleeritavad
  - lihtsustavad süsteemide integreerimist
- Võimaldavad teenusekeskset arhitektuuri (Service Oriented Architecture)

Omadused

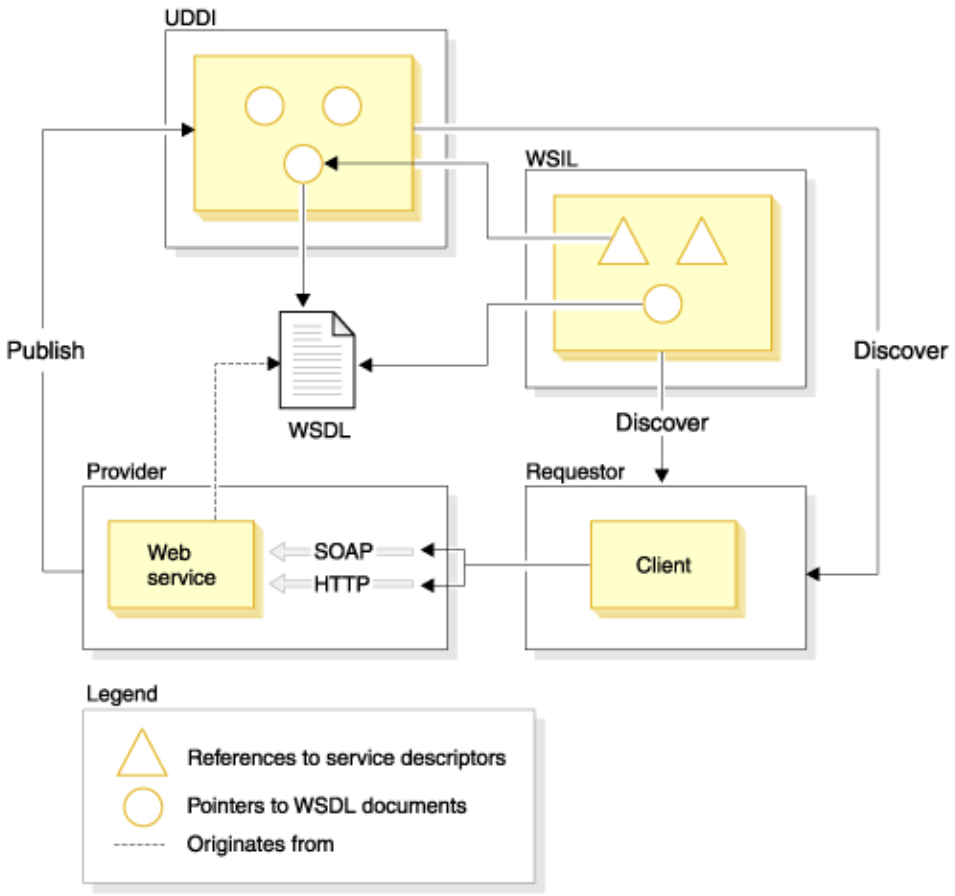
- **Taaskasutatavus** (Reusability) **-** Süsteemi pakutav teenus on implementeeritud üks kord ja seda kasutatakse korduvalt
- **Koostalitlusvõime** (Interoperability) - Veebiteenust peabs aama kasutada sõltumatult programmeerimiskeelest või platvormist
- **Integratsioon -** veebiteenused võimaldavad lihtsamat suhtlust erinevate organisatsioonide süsteemide vahel
- **Komponeeritavus** (Composability) - teenuseid saab ühendada kokku suurematekst teenusteks, mida ükski neist iseseisvalt pakkuda ei saaks

Omadused

- **Leitavus** (Discoverability) - veebiteenused saab "tõhusalt" otsida ja leida
- **Abstraheeritus** (Abstraction) - veebiteenuste sisemina loogika on peidetud teenuse tarbijate eest. Avaldatakse vaid see info, mis on vajalik teenuse tarbimiseks
- **Autonoomsus** (Autonomy) - veebiteenus omab täielikku kontrolli oma sisemise loogika üle
- **Lahtiselt ühendatud** (Loosely coupled) - Iga teenus eksisteerib iseseisvalt teistest sõltumatult
- **Staatuseta** (Statelessness) **-** teenused käsitlevad iga päringut iseseisvana, sõltumatult eelnevast suhtlusest, mis on selle jaoks väga oluline skaleeritavate teenuste pakkumine

WSDL ja UDDI

- **WSDL** - **W**eb **S**ervices **D**escription **L**anguage
  - Standard veebiteenuste, ressursside ja operatsioonide kirjeldamiseks
- **UDDI** - **U**niversal **D**escription, **D**iscovery and **I**ntegration
  - XML-põhine standard veebiteenuste kirjeldamiseks, avaldamiseks ja avastamiseks
- **WSIL** - **W**eb **S**ervices **I**nspection **L**anguage
  - XML-põhine spetsifikatsioon, mis defineerib, kuidas otse teenuse pakkujalt küsida veebiteenuste kirjeldusi



https://www.ibm.com/docs/en/rsm/7.5.0?topic=overview-web-services-standards

WSDL

- XML-formaadis kirjeldus veebiteenuste kohta
- Kirjeldab porte (viise teenuseni jõudmiseks) ja teenuseid endid
  - Mis operatsioone teenus pakub?
  - Missugused on parameetrid ja tagastusväärtused?
  - Mismoodi andmeid esitatakse?
  - Mis aadressil ja protokolliga ligi pääseb?
- Mõnevõrra analoogne IDL-iga
- Ligipääsuks SOAP või tavaline HTTP

WSDL sisu elemendid

- **Types: Andmetüübid**
  - Sageli kasutatakse XML Schema tüüpe nagu SOAP-iski
- **Messages**: Andmetüüpidest ehitatakse lihtsad **teate tüübid**
- **Port types**: **Pordi tüüp** kirjeldab operatsiooni nime ja seob sellega sisend-väljundi teatetüübid
- **Binding**: **Seosed** määravad andmevahetuseks näiteks SOAP-i, HTTP, MIME tüübi
- **Port: pordid** panevad seosed vastavusse konkreetse URL-iga
- **Service: Teenus** kirjeldab teenuse nime ja lõpp punkti

WSDL näide

```
<?xml version="1.0" encoding="utf-8"?>
<definitions
    name="StockQuote"
    targetNamespace="http://example.com/stockquote/"
    xmlns:tns="http://example.com/stockquote/"
    xmlns:xsd1="http://example.com/stockquote/schema/"
    xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
    xmlns="http://schemas.xmlsoap.org/wsdl/">
    <types>
    ...
    <message name="GetLastTradePriceInput">
    <message name="GetLastTradePriceOutput">
    ...
    <portType name="StockQuotePortType">
        <operation name="GetLastTradePrice">
    </portType>
    ...
    <binding name="StockQuoteSoapBinding" type="tns:StockQuotePortType">
        <soap:binding style="document"
            transport="http://schemas.xmlsoap.org/soap/http"/>
        <operation name="GetLastTradePrice">
    </binding>
    ...
    <service name="StockQuoteService">
        <port name="StockQuotePort" binding="tns:StockQuoteSoapBinding">
    </service>
</definitions>
```

WSDL tüüpide definitsioon

```
<types>
   <schema
      targetNamespace="http://example.com/stockquote/schema/"
      xmlns="http://www.w3.org/2001/XMLSchema">
      <element name="TradePriceRequest">
         <complexType>
            <all>
               <element name="tickerSymbol" type="string"/>
            </all>
         </complexType>
      </element>
      <element name="TradePrice">
         <complexType>
            <all>
              <element name="price" type="float"/>
            </all>
         </complexType>
      </element>
   </schema>
</types>
```

Defineerib teenuse operatsioonide sõnumite sisemise struktuuri (väljade tüübid)

WSDL sõnumite definitsioon

- Defineerib teenuse operatsioonide sõnumite tüübid)

```
<message name="GetLastTradePriceInput">
    <part name="body" element="xsd1:TradePriceRequest"/>
</message>

<message name="GetLastTradePriceOutput">
    <part name="body" element="xsd1:TradePrice"/>
</message>
```

WSDL pordi tüüpide definitsioon

- Defineerib operatsioonid, ning sisendite ja väljundite tüübid

```
<portType name="StockQuotePortType">
    <operation name="GetLastTradePrice">
        <input message="tns:GetLastTradePriceInput"/>
        <output message="tns:GetLastTradePriceOutput"/>
    </operation>
</portType>
```

WSDL seoste (binding) definitsioon

- Defineerib, kuidas suhelda teenusega

```
<binding name="StockQuoteSoapBinding"
   type="tns:StockQuotePortType">
        <soap:binding style="document"
                  transport="http://schemas.xmlsoap.org/soap/http"/
   >
        <operation name="GetLastTradePrice">
            <soap:operation
   soapAction="http://example.com/GetLastTradePrice"/>
            <input>  <soap:body use="literal"/></input>
            <output> <soap:body use="literal"/></output>
        </operation>
    </binding>
```

WSDL teenuse definitsioon

- Defineerib teenuse kirje, ja selle asukoha

```
<service name="StockQuoteService">
    <port name="StockQuotePort"
                binding="tns:StockQuoteSoapBinding">
        <soap:address location="http://location/sample"/>
    </port>
</service>
```

UDDI

- **U**niversal **D**escription **D**iscovery and **I**ntegration
- Veebiteenuste register, et kliendid ja serverid üksteist automaatselt leida suudaksid
- Teenuste registreerimine ja pärimine
- Juurdepääs SOAP protokolliga
- Praktikas tundub vähe kasutuses olevat

UDDI näide: teenuse otsimine

```
<find_service generic="2.0" xmlns="urn:uddi-org:api_v2">
    <categoryBag>
        <keyedReference
              tModelKey="uuid:6e090afa-33e5-36eb-81b7-1ca18373f457"
              keyName="WSDL type"
              keyValue="service" />
        <keyedReference
              tModelKey="uuid:d01987d1-ab2e-3013-9be2-2a66eb99d824"
              keyName="service namespace"
              keyValue="http://example.com/stockquote/" />
        <keyedReference
              tModelKey="uuid:2ec65201-9109-3919-9bec-c9dbefcaccf6"
              keyName="service local name"
              keyValue="StockQuoteService" />
    </categoryBag>
</find_service>
```

WSIL

- WSIL - Web Services Inspection Language
- Alternatiiv UDDI'le
  - UDDI on tsentraliseeritud
  - WSIL on detsentraliseeritud
- Võimaldab pöörduda otse teenusepakkuja poole ja küsida tema pakutavaid teenuseid
  - Ei ole vaja teada, millisest registrist otsida
  - Saab kasutada, kui on teada, mis organisatsiooni veebiteenuseid soovime otsida

WSIL näide

- WSIL dokument organisatsiooni kodulehel
- Kirjeldab ära teenused, mida see organisatsioon või süsteem pakub
- Teenuste kirjeldus üldjuhul WSDL standardi kaudu
- Võib ka viidata organisatsiooni sisese UDDI registrile

```
<?xml version="1.0"?>
<inspection xmlns="http://schemas.xmlsoap.org/ws/2001/10/inspection/">
  ...
  <service>
    <name xml:lang="en-US">StockQuoteService</name>
    <description referencedNamespace="http://schemas.xmlsoap.org/wsdl/"
                 location="http://localhost:8080/webservices/wsdl/stockquote/sqs.wsdl">
      <wsilwsdl:reference endpointPresent="true">
        <wsilwsdl:implementedBinding
          xmlns:interface="http://www.getquote.com/StockQuoteService-interface">
          interface:StockQuoteServiceBinding
        </wsilwsdl:implementedBinding>
      </wsilwsdl:reference>
    </description>
  </service>
  ...
</inspection>
```

OpenAPI

- Spetsifikatsioon HTTP põhiste veebiteenuste standardseks kirjeldamiseks, väljakutsumiseks ja visualiseerimiseks
  - https://github.com/OAI/OpenAPI-Specification
- Algselt tuntud kui Swaggeri spetsifikatsioon
- Võimaldab avastada ja mõista teenuse funktsionaalsust ilma juurdepääsuta lähtekoodile või dokumentatsioonile
- Kasutuse näited:
  - Interaktiivne dokumentatsioon
  - koodi genereerimine dokumentatsiooni, klientide ja serverite jaoks
  - testjuhtumite loomise automatiseerimine
- OpenAPI > REST
  - Toetab üldiselt HTTP API'sid.
  - REST on rangem
  - Aga saab täiesti REST API'de defineerimiseks ja kirjeldamiseks kasutada

OpenAPI spetsifikatsioon

- Kasutajad loovad spetsifikatsioonis veebiteenuse malli, mis sisaldab:
  - Ressursid, nende lõpp-punktid ja kirjeldused
  - Toetatud HTTP operatsioonid
  - Lubatud sisendid ja eeldatavad väljundid
- Malli formaat on YAML või JSON dokument
- OpenAPI spetsifikatsiooni kasutatakse tihti REST API dokumentatsioonina
- OpenAPI spetsifikatsiooni saab genereerida ka veebiteenuse koodi põhjal

XML vs JSON vs YAML

XML	JSON	YAML

<Servers>
<Server>
<name>Server1</name>
<owner>John</owner>
<created>123456</created>
<status>active</status>
</Server>
</Servers>
	{
	Servers: [
	{
	name: Server1,
	owner: John,
	created: 123456,
	status: active
	}
	]
	}
		Servers:
		- name: Server1
		owner: John
		created: 123456
		status: active

Allikas: https://developer.ibm.com/tutorials/yaml-basics-and-usage-in-kubernetes/

OpenAPI YAML spetsifikatsioon

```yaml
openapi: "3.0.0"
info:
  version: 1.0.0
  title: Swagger Petstore
  license:
    name: MIT
servers:
  - url: http://petstore.swagger.io/v1
paths:
  /pets:
    get:
    ---
    post:
    ---
  /pets/{petId}:
    get:
    ---
components:
  schemas:
    Pet:
    ---
    Pets:
    ---
    Error:
    ---
```

OpenAPI info blokk

```
title: Sample Pet Store App
description: This is a sample server for a pet store.
termsOfService: http://example.com/terms/
contact:
    name: API Support
    url: http://www.example.com/support
    email: support@example.com
license:
    name: Apache 2.0
    url: https://www.apache.org/licenses/LICENSE-2.0.html
version: 1.0.1
```

OpenAPI servers blokk

- Defineerib serverite aadressid, mille kaudu REST API on kätte saadav.

```
servers:
- url: https://development.gigantic-server.com/v1
  description: Development server
- url: https://staging.gigantic-server.com/v1
  description: Staging server
- url: https://api.gigantic-server.com/v1
  description: Production server
```

OpenAPI paths blokk

- Kirjeldab REST API otspunktid ja toetatud REST operatsioonid
- Näide: Kõikide koduloomade nimekira küsimine üle GET päringu

```
/pets:          - REST ots-punkt
  get:          - REST operatioon
    description: Returns all pets from the system that the user has access to
    responses:  - API vastused
      '200':    - Vastuse tüübi määrab HTTP kood
        description: A list of pets.
        content:  - Vastuse sõnumi definitsioon
          application/json:  - Vastuse sõnumi MIME formaat
            schema:          - Sõnumi struktuur
              type: array
              items:
                $ref: '#/components/schemas/pet'   - Elementide struktuur on viitena - selle sisu on kirjas schemas blokis
```

OpenAPI paths bloki näide

```
get:
  description: Returns pets based on ID
  summary: Find pets by ID
  operationId: getPetsById
  responses:
    '200':
      description: pet response
      content:
        '*/*' :
          schema:
            type: array
            items:
              $ref: '#/components/schemas/Pet'
parameters:
- name: id
  in: path
  description: ID of pet to use
  required: true
  schema:
    type: array
    items:
      type: string
  style: simple
```

- Sisendparameetrid

- /pets/id - lemmiklooma ID

- kohustuslik parameeter

OpenAPI operatiooni näide sisendi ja väljundi definitsiooniga

```yaml
summary: Updates a pet in the store with form data
operationId: updatePetWithForm
parameters:
- name: petId
requestBody:
  content:
    'application/x-www-form-urlencoded':
      schema:
        properties:
          name:
            description: Updated name of the pet
            type: string
          status:
            description: Updated status of the pet
            type: string
        required:
          - status
responses:
  '200':
    description: Pet updated.
    content:
      'application/json': {}
      'application/xml': {}
  '405':
    description: Method Not Allowed
    content:
      'application/json': {}
      'application/xml': {}
```

OpenAPI components blokk

- Defineerib erinevad REST api meetodite sisendid, ja väljundid:
  - **Schemas** – sõnumite (alam)objektide struktuurid
  - **Responses** – Vastuse tüübid
  - **Parameters** – Operatsioonide parameetrid
  - **Examples** - näited sõnumite või objektide struktuurist
  - **RequestBodies** - päringu sisu struktuurid
  - **Headers** - päise väärtused, mis on vajalikud, lubatud, või vastuse osa
  - **SecuritySchemes –** defineerib autentimise info

```
components:
 schemas:
  GeneralError:
   type: object
   properties:
    code:
     type: integer
     format: int32
    message:
     type: string
  Category:
   type: object
   properties:
    id:
     type: integer
     format: int64
    name:
     type: string
  Tag:
   type: object
   properties:
    id:
     type: integer
     format: int64
    name:
     type: string
 responses:
  NotFound:
   description: Entity not found.
  IllegalInput:
   description: Illegal input for operation.
  GeneralError:
   description: General Error
   content:
    application/json:
     schema:
      $ref: '#/components/schemas/GeneralError'
```

OpenAPI schema näited

```
application/json:
  schema:
    $ref: "#/components/schemas/Pet"
  examples:
    cat:
      summary: An example of a cat
      value:
        name: Fluffy
        petType: Cat
        color: White
        gender: male
        breed: Persian
    dog:
      summary: An example of a dog with a cat's name
      value:
        name: Puma
        petType: Dog
        color: Black
        gender: Female
        breed: Mixed
    frog:
      $ref: "#/components/examples/frog-example"
```

OpenAPI schemas blokk

- Defineerib ära mingi objekti struktuuri, väljad, andmetüübid
- Näiteks:
  - JSON sõnumi struktuur või alamstruktuur
  - XML vastus

```yaml
components:
  schemas:
    Pet:
      type: object
      discriminator:
        propertyName: petType
      properties:
        name:
          type: string
        petType:
          type: string
      required:
      - name
      - petType
```

OpenAPI SecuritySchemes blokk

- Defineerib autentimise info
  - Mis meetodid nõuavad autentimist
  - Millised autentimisviisid on kasutusel
  - Kuidas edastada autentimisinfot (nt. API_KEY, BasicAuth)
  - Kus asuvad välised autentimise teenused (nt. oAuth)

```
securitySchemes:
  api_key:
    type: apiKey
    name: api_key
    in: header
  petstore_auth:
    type: oauth2
    flows:
      implicit:
        authorizationUrl: http://example.org/api/oauth/dialog
        scopes:
          write:pets: modify pets in your account
          read:pets: read your pets
```

OpenAPI generaator

- Oskab OpenAPI spetsifikatsiooni põhjal genereerida nii kliendi kui ka serveri koodi:
  - Kliendid: 30+ keeles
  - Serverid: 15 keeles
- Loodud serverikood sisaldab:
  - Serveri kood (nt. Flask),
  - Docker fail selle üles seadmiseks konteinerina
  - Veebiliides API dokumenteerimiseks ja reaalajas testimiseks
- Kasutaja peab implementeerima ainult REST-meetodite sisu (PUT, GET, POST, DELETE) ja lisafunktsioonid (nt. DB-ühendus, autentimine, integratsioonid)

API serveri koodi genereerimine

Ada, **C#** (ASP.NET Core, NancyFx), **C++** (Pistache, Restbed, Qt5 QHTTPEngine), **Erlang**, **F#** (Giraffe), **Go** (net/http, Gin, Echo), **Haskell** (Servant, Yesod), **Java** (MSF4J, Spring, Undertow, JAX-RS: CDI, CXF, Inflector, Jersey, RestEasy, Play Framework, PKMST, Vert.x), **Kotlin** (Spring Boot, Ktor, Vertx), **PHP** (Laravel, Lumen, Mezzio (fka Zend Expressive), Slim, Silex, Symfony), **Python** (FastAPI, Flask), **NodeJS**, **Ruby** (Sinatra, Rails5), **Rust** (rust-server), **Scala** (Akka, Finch, Lagom, Play, Scalatra)

API klientide genereerimine

ActionScript, **Ada**, **Apex**, **Bash**, **C**, **C#** (.net 2.0, 3.5 or later, .NET Standard 1.3 - 2.0, .NET Core 2.0, .NET 5.0. Libraries: RestSharp, HttpClient),

C++ (Arduino, cpp-restsdk, Qt5, Tizen, Unreal Engine 4),

Clojure, **Crystal**, **Dart**, **Elixir**, **Elm**, **Eiffel**, **Erlang**, **Go**, **Groovy**, **Haskell**

Java (Apache HttpClient, Jersey1.x, Jersey2.x, OkHttp, Retrofit1.x, Retrofit2.x, Feign, RestTemplate, RESTEasy, Vertx, Google API Client Library for Java, Rest-assured, Spring 5 Web Client, MicroProfile Rest Client),

k6, **Kotlin**, **Lua**, **Nim**, **Node.js/JavaScript** (ES5, ES6, AngularJS with Google Closure Compiler annotations, Flow types, Apollo GraphQL DataStore),

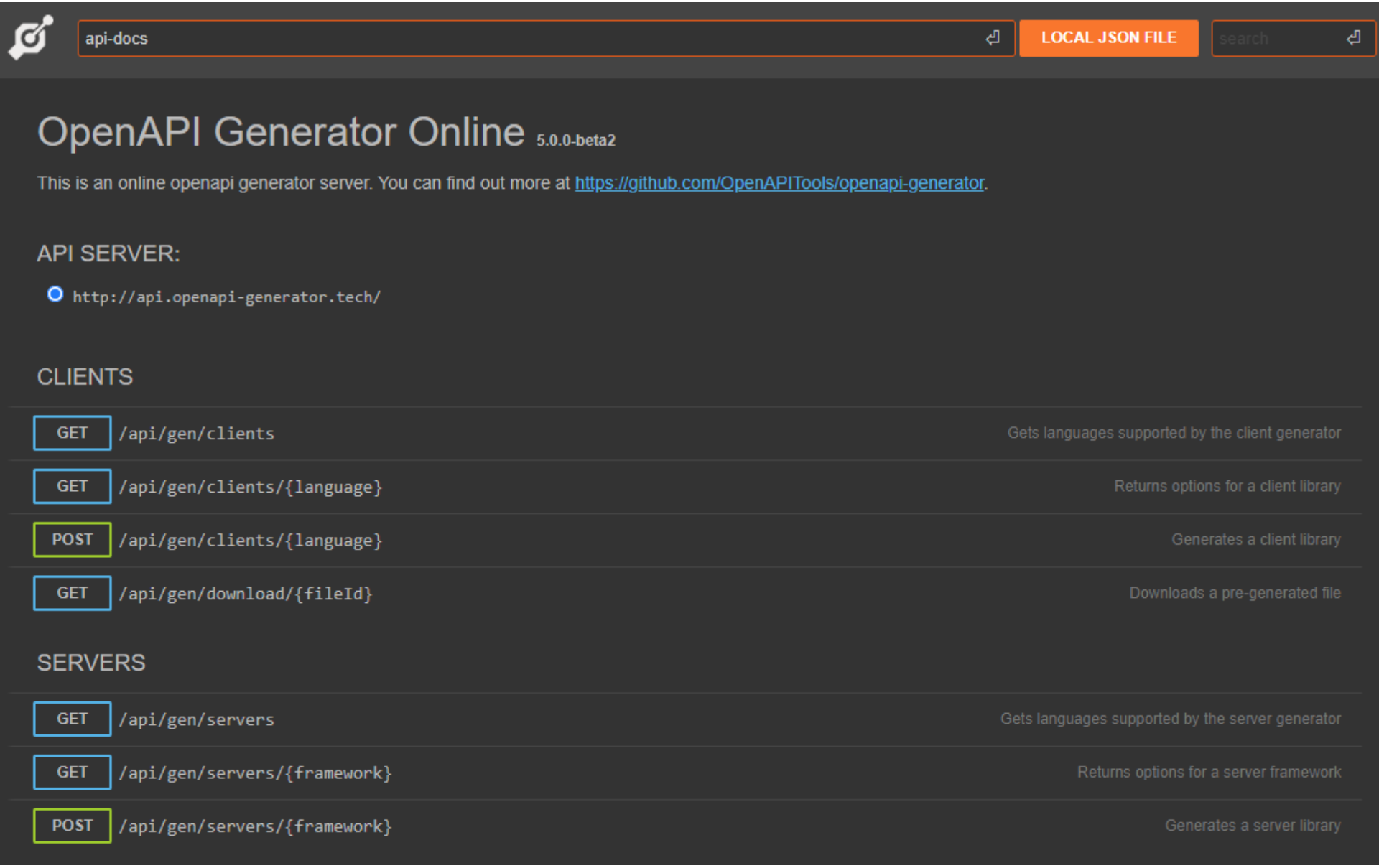
Objective-C, **OCaml**, **Perl**, **PHP**, **PowerShell**, **Python**, **R**, **Ruby**, **Rust**

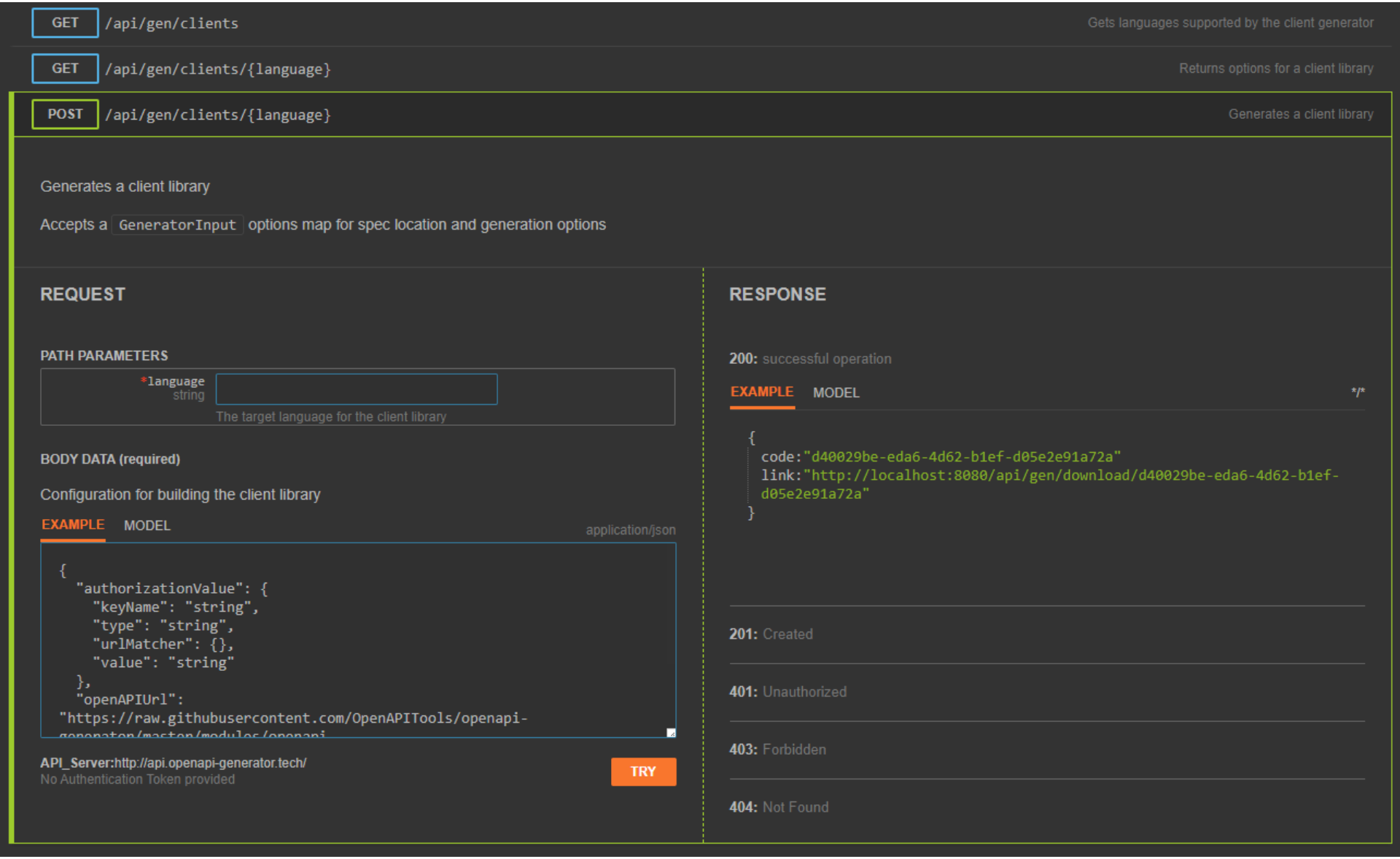
Scala (akka, http4s, scalaz, sttp, swagger-async-httpclient), **Swift**

Typescript (AngularJS, Angular (2.x - 11.x), Aurelia, Axios, Fetch, Inversify, jQuery, Nestjs, Node, redux-query, Rxjs)

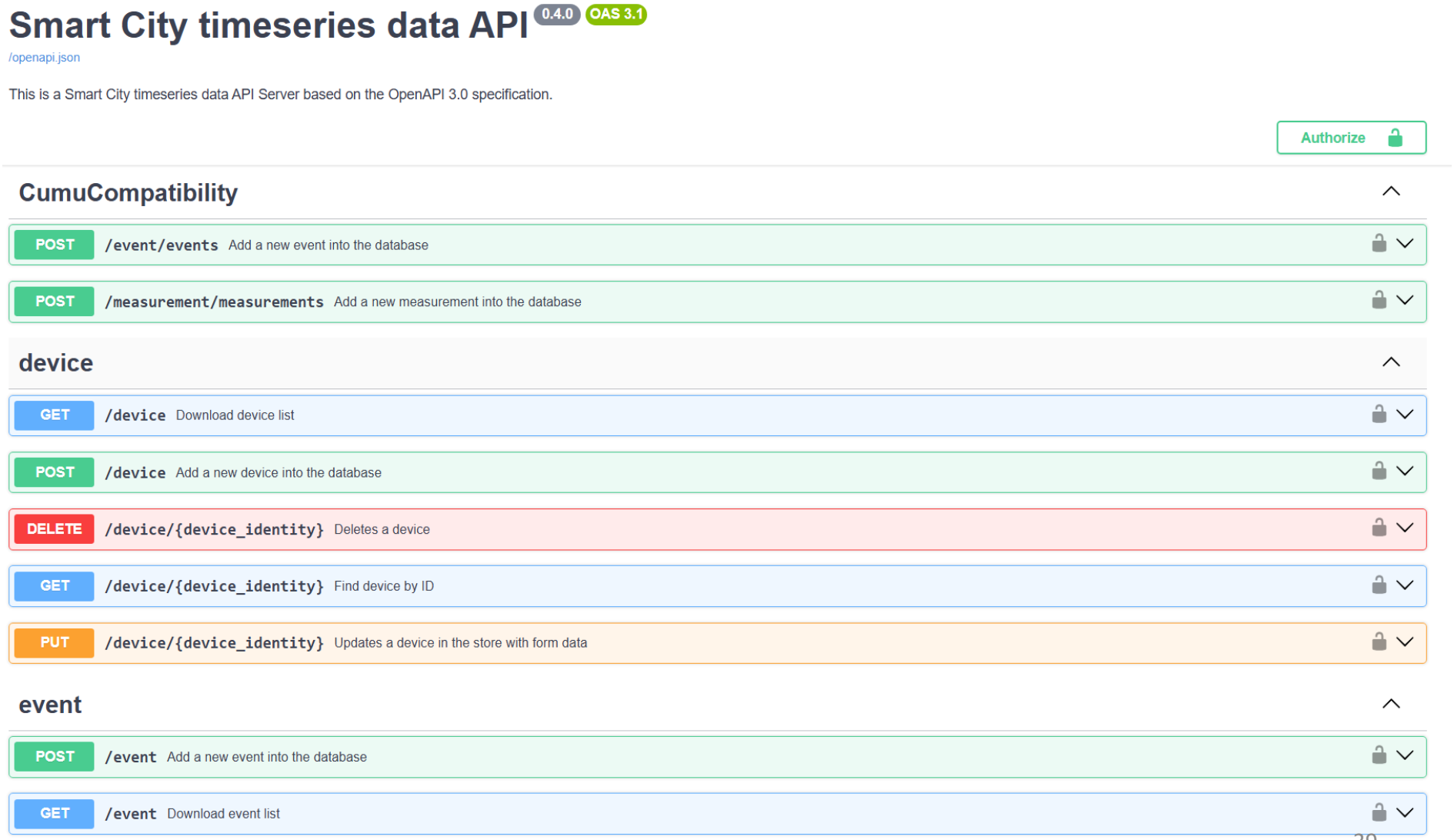
OpenAPI veebi dokumentatsioon

- OpenAPI spetsifikatsiooni põhjal genereeritakse API'le interaktiivne veebidokumentatsioon
- Lihtsustab API õppimist
  - Tihti sisaldab näite päringuid, kirjeldusi
  - andmetüüpide definitsioone
  - Päringu vastuse näiteid
- Saab kasutada API testimiseks sarnaselt Postman'ile


api-docs
LOCAL JSON FILE
search
OpenAPI Generator Online 5.0.0-beta2
This is an online openapi generator server. You can find out more at https://github.com/OpenAPITools/openapi-generator.
API SERVER:
http://api.openapi-generator.tech/
CLIENTS
GET /api/gen/clients
Gets languages supported by the client generator
GET /api/gen/clients/{language}
Returns options for a client library
POST /api/gen/clients/{language}
Generates a client library
GET /api/gen/download/{fileId}
Downloads a pre-generated file
SERVERS
GET /api/gen/servers
Gets languages supported by the server generator
GET /api/gen/servers/{framework}
Returns options for a server framework
POST /api/gen/servers/{framework}
Generates a server library


GET /api/gen/clients
Gets languages supported by the client generator
GET /api/gen/clients/{language}
Returns options for a client library
POST /api/gen/clients/{language}
Generates a client library
Generates a client library
Accepts a GeneratorInput options map for spec location and generation options
REQUEST
PATH PARAMETERS
*language
string
The target language for the client library
BODY DATA (required)
Configuration for building the client library
EXAMPLE MODEL
application/json
{
"authorizationValue": {
"keyName": "string",
"type": "string",
"urlMatcher": {},
"value": "string"
},
"openAPIUrl":
"https://raw.githubusercontent.com/OpenAPITools/openapi-
API_Server:http://api.openapi-generator.tech/
No Authentication Token provided
TRY
RESPONSE
200: successful operation
EXAMPLE MODEL
/
{
code:"d40029be-eda6-4d62-b1ef-d05e2e91a72a"
link:"http://localhost:8080/api/gen/download/d40029be-eda6-4d62-b1ef-
d05e2e91a72a"
}
201: Created
401: Unauthorized
403: Forbidden
404: Not Found

Näite API veebiliides/dokumentatsioon



Päringu tegemine veebiliidese kaudu

measurement

POST `/measurement` Add a new measurement into the database

Add a new measurement into the database

Parameters | Cancel | Reset

No parameters

Request body — application/json

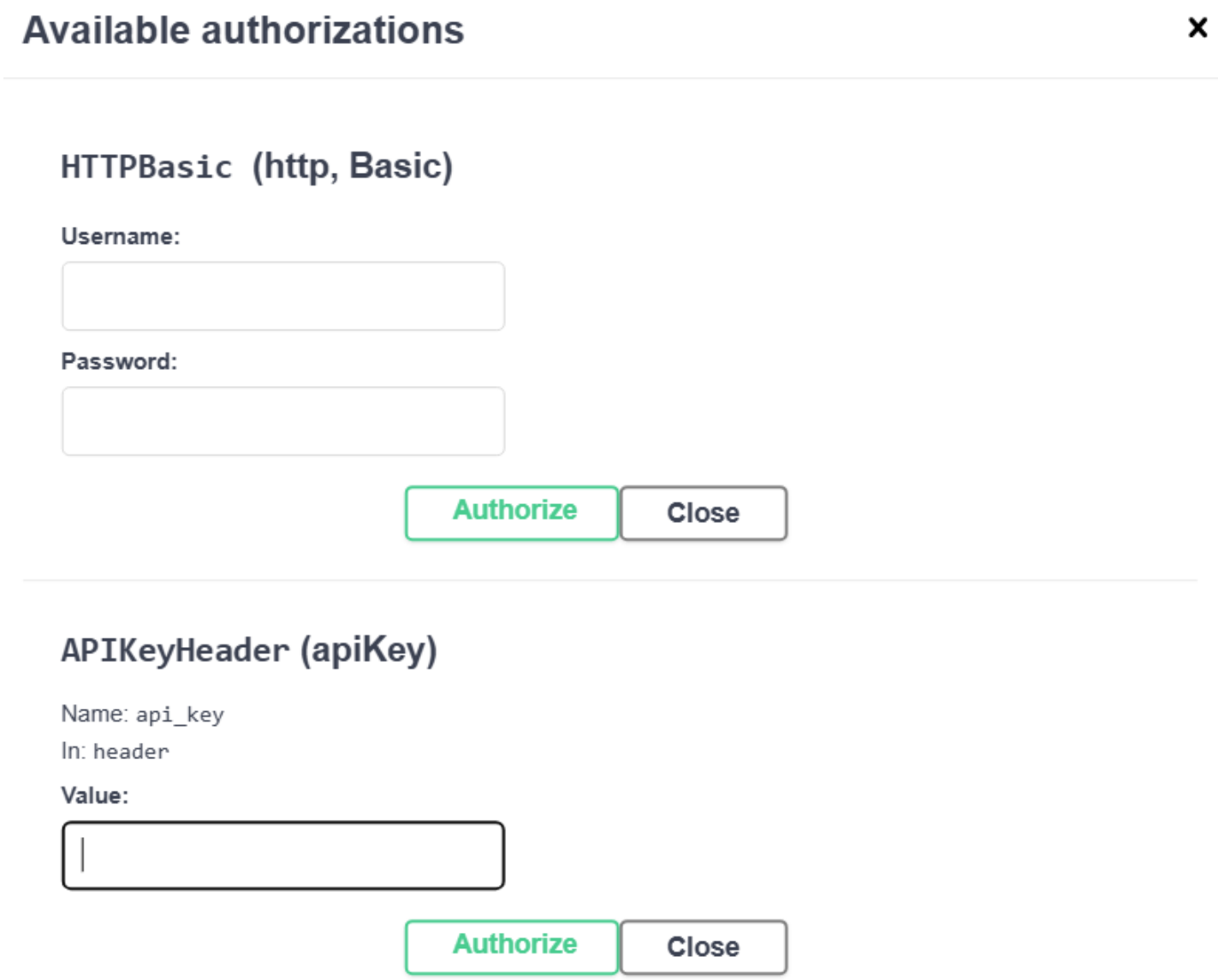
```
{
   "measurements": [
    {
    "time": "2025-02-08T05:00:16.500Z",
    "device_identity": "electricity_counter_mäe_13_2",
    "measurement_type": "electricity",
    "series": [
      {
        "series_type": "electricity-consumed",
        "unit": "kwh",
        "value": 2.632
      },
      {
        "series_type": "current-power",
        "unit": "kw",
        "value": 0.13
      }
    ]
   }
```

Execute

Responses

Code	Description	Links
201	Successful operation Media type: application/json	*No links*

Autentimise info



Päringu vastus

Responses

Curl

```
curl -X 'POST' \
  'https://reaalajaandmed.tartu.ee/measurement' \
  -H 'accept: application/json' \
  -H 'api_key: uUW-t6VfzSuNvV_                                          A' \
  -H 'Content-Type: application/json' \
  -d '{
   "measurements": [
    {
    "time": "2025-02-08T05:00:16.500Z",
    "device_identity": "electricity_counter_mäe_13_2",
    "measurement_type": "electricity",
    "series": [
      {
        "series_type": "electricity-consumed",
        "unit": "kwh",
        "value": 2.632
      },
      {
        "series_type": "current-power",
        "unit": "kw",
        "value": 0.13
      }
    ]
   }
 ]
}
'
```

Request URL

```
https://reaalajaandmed.tartu.ee/measurement
```

Server response

Code	Details
201	Response body

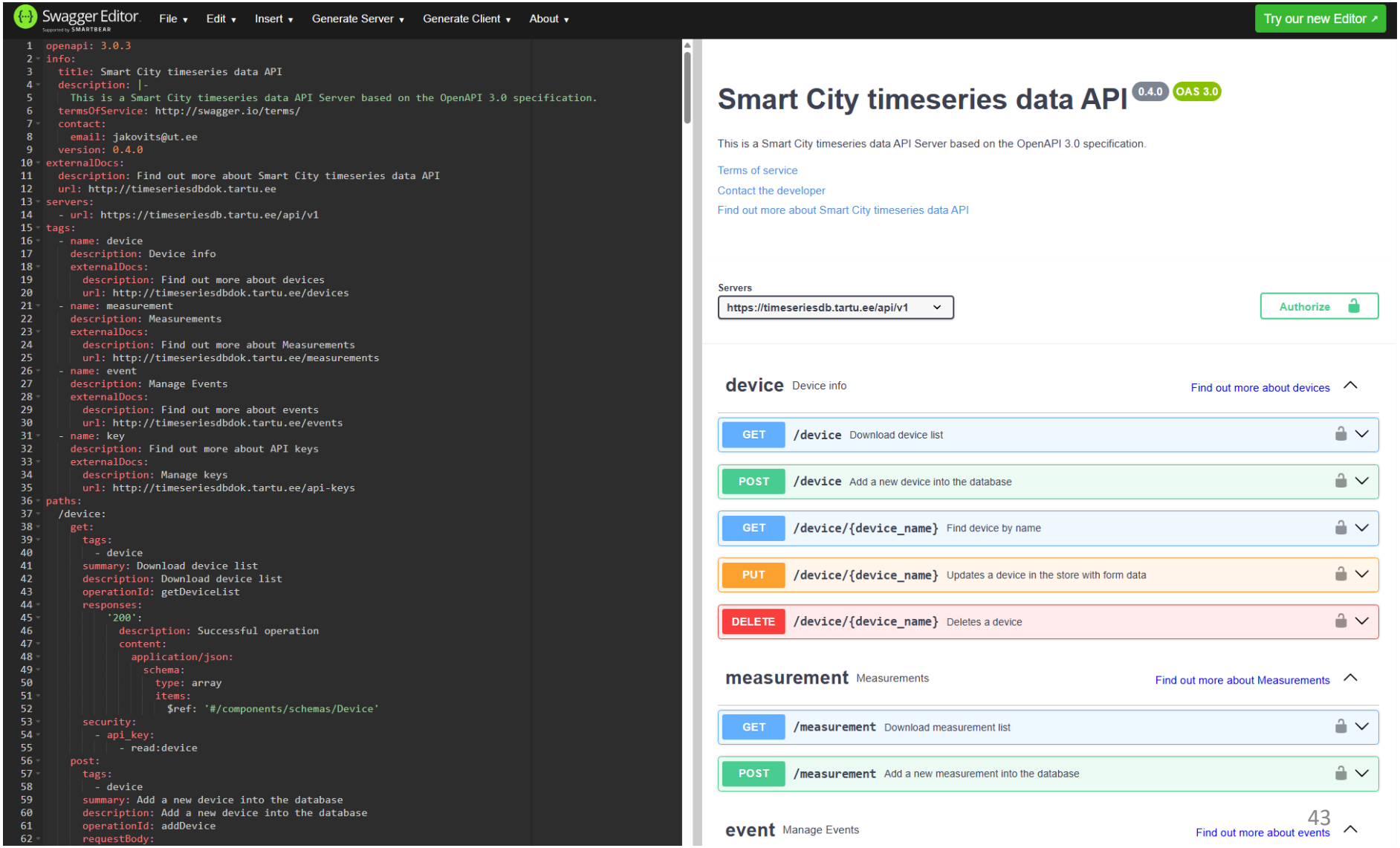
```
{
  "message": "Added measurements to the database"
}
```

Download

Response headers

```
content-length: 48
content-type: application/json
date: Wed,19 Mar 2025 09:17:45 GMT
server: nginx/1.20.1
strict-transport-security: max-age=31536000
```

Swagger Editor (https://editor.swagger.io/)



Selle nädala praktikum

- OpenAPI spetsifikatsiooni loomine Raamatu halduse API jaoks
- REST API serveri koodi genereerimine OpenAPI genereetori abil
- Genereeritud API skeletoni implementeerimine

Järgmine loeng

- Pilvetehnoloogia
- MicroSoft Azure pilveteenused