SYT GK771 Windpark REST - Armin Daryabegi 21.11.2020

Das Ziel ist es eine **REST API** aufzusetzen welche uns Daten für eine Windkraftanlage zurückgibt. Das ganze wird mit **Java**, **Spring Boot** und **Gradle/Maven** umgesetzt.

Als erstes bin ich dem classroom mit diesem Link beigetreten. Als nächstes habe ich mit dem Befehl `git clone https://github.com/TGM-HIT/syt4-gk771-windpark-rest-adaryabegi.git` das Repository in meinen lokalen Ordner reinkopiert.

Anschließend habe ich den Ordner **windpark_REST_JSON** entpackt und dannach mit IntelliJ geöffnet. Dann konnte ich mit `mvn spring-boot:run` das Projekt starten wodurch ich nun auf **localhost:8080** gehen konnte und eine Welcomes Message bekommen habe.

REST in JAVA verstehen

Für dieses Projekt wurde Java und Spring Boot verwendet. Eine REST API besteht aus mehreren **RequestMappings** welche mit der Annotation **@RequestMapping("/")** angeschrieben wobei der Parameter **"/"** den Pfad der URL angibt.

```
@RequestMapping(value="/windengine/{windengineID}/xml",produces = {
"application/xml" } )
public WindengineData windengineDataXML( @PathVariable String windengineID ) {
  return service.getWindengineData( windengineID );
}
```

Daten als XML und JSON bekommen

Standardmäßig wurden die Daten als XML ausgegeben unter der URL **http://localhost:8080/windengine/001/data**. Ich habe dafür 2 Request Mappings gemacht undzwar:

```
@CrossOrigin(origins = "http://localhost:8080")
@RequestMapping(value="/windengine/{windengineID}/xml",produces = {
"application/xml" } )
public WindengineData windengineDataXML( @PathVariable String windengineID ) {
  return service.getWindengineData( windengineID );
}

@CrossOrigin(origins = "http://localhost:8080")
@RequestMapping(value="/windengine/{windengineID}/json",produces = {
"application/json" })
public WindengineData windengineDataJSON( @PathVariable String windengineID ) {
  return service.getWindengineData(windengineID);
}
```

Wenn man bei **RequestMapping** den Parameter **produces** hinzufügt kann man angeben ob man es in xml oder json ausgeben möchte.

Dadurch kann ich mir jetzt entweder die Daten über JSON oder XML auslesen lassen. Dazu müsste ich nun auf http://localhost:8080/windengine/001/json?mediaType=json gehen um die Daten in JSON zu bekommen. Für XML gehts so: http://localhost:8080/windengine/001/xml?mediaType=xml

Consumer Web App

Die nächste Aufgabe war eine kleine Website aufzusetzen die diese Schnittstelle, die ich davor erklärt habe, **(REST API)** benutzt. Dafür habe ich einen **nodeJs** Server aufgesetzt welcher als View Engine pug verwendet.

Hier kann ich nun die Winkraftanlage eingeben von denen ich die Daten bekommen möchte, dann wird ein Request an die **API** geschickt, und anschließend werden die Daten in einer Tabelle angezeigt.

app.js

```
const express = require("express");
const app = express();
const path = require("path");
const router = express.Router();
const fetch = require("node-fetch");
const isAfter = require("date-fns/isAfter");
const isBefore = require("date-fns/isBefore");
const { isSameDay } = require("date-fns");

app.set("view engine", "pug");
app.set("views", path.join(__dirname, "views"));
// Parse URL-encoded bodies (as sent by HTML forms)
app.use(express.urlencoded());
app.use(express.static(__dirname + "/public"));
const alldata = [];

// Parse JSON bodies (as sent by API clients)
app.use(express.json());
router.get("/", function (req, res) {
  res.render("index");
});

router.post("/", function (req, res) {
  const { windkraftID, von, bis, nach } = req.body;

  if (windkraftID) {
    fetch(`http://localhost:8080/windengine/${windkraftID}/json?
mediaType=json`)
      .then((res) => res.text())
```

```
    .then((text) => {
      var fetchedData = JSON.parse(text);
      var keyNames = [];
      for (const [key, value] of Object.entries(fetchedData)) {
        keyNames.push(key);
      }
      keyNames.shift(); // remove first item bc it is already set (static)
      alldata.push(fetchedData);
      if (von && bis) {
        const filteredData = alldata.filter((item) => {
          var timy = new Date(item.timestamp);
          var vonDate = new Date(von);
          var bisDate = new Date(bis);
          return (
            (isAfter(timy, vonDate) && isBefore(timy, bisDate)) ||
            isSameDay(vonDate, timy) ||
            isSameDay(bisDate, timy)
          );
        });

        res.render("index", {
          data: filteredData,
          keyNames,
          von,
          bis,
          nach: null,
        });
      } else if (nach) {
        const filteredData = alldata.filter((item) => {
          var timy = new Date(item.timestamp);
          var nachDate = new Date(nach);

          return isAfter(timy, nachDate) || isSameDay(nachDate, timy);
        });
        res.render("index", {
          data: filteredData,
          keyNames,
          von: null,
          bis: null,
          nach,
        });
      } else {
        res.render("index", { data: alldata, keyNames });
      }
    });
} else if (von && bis) {
  var keyNames = [];
  if (alldata.length > 0) {
    for (const [key, value] of Object.entries(alldata[0])) {
```

```
        keyNames.push(key);
      }
      keyNames.shift();
    }
    const filteredData = alldata.filter((item) => {
      var timy = new Date(item.timestamp);
      var vonDate = new Date(von);
      var bisDate = new Date(bis);
      return (
        (isAfter(timy, vonDate) && isBefore(timy, bisDate)) ||
        isSameDay(vonDate, timy) ||
        isSameDay(bisDate, timy)
      );
    });

    res.render("index", { data: filteredData, keyNames, von, bis, nach: null
});
  } else if (nach) {
    var keyNames = [];
    if (alldata.length > 0) {
      for (const [key, value] of Object.entries(alldata[0])) {
        keyNames.push(key);
      }
      keyNames.shift();
    }
    const filteredData = alldata.filter((item) => {
      var timy = new Date(item.timestamp);
      var nachDate = new Date(nach);

      return isAfter(timy, nachDate) || isSameDay(nachDate, timy);
    });
    res.render("index", {
      data: filteredData,
      keyNames,
      von: null,
      bis: null,
      nach,
    });
  } else {
    res.render("index");
  }
});

//add the router
app.use("/", router);
app.listen(process.env.port || 3000);

console.log("Running at Port 3000");
```

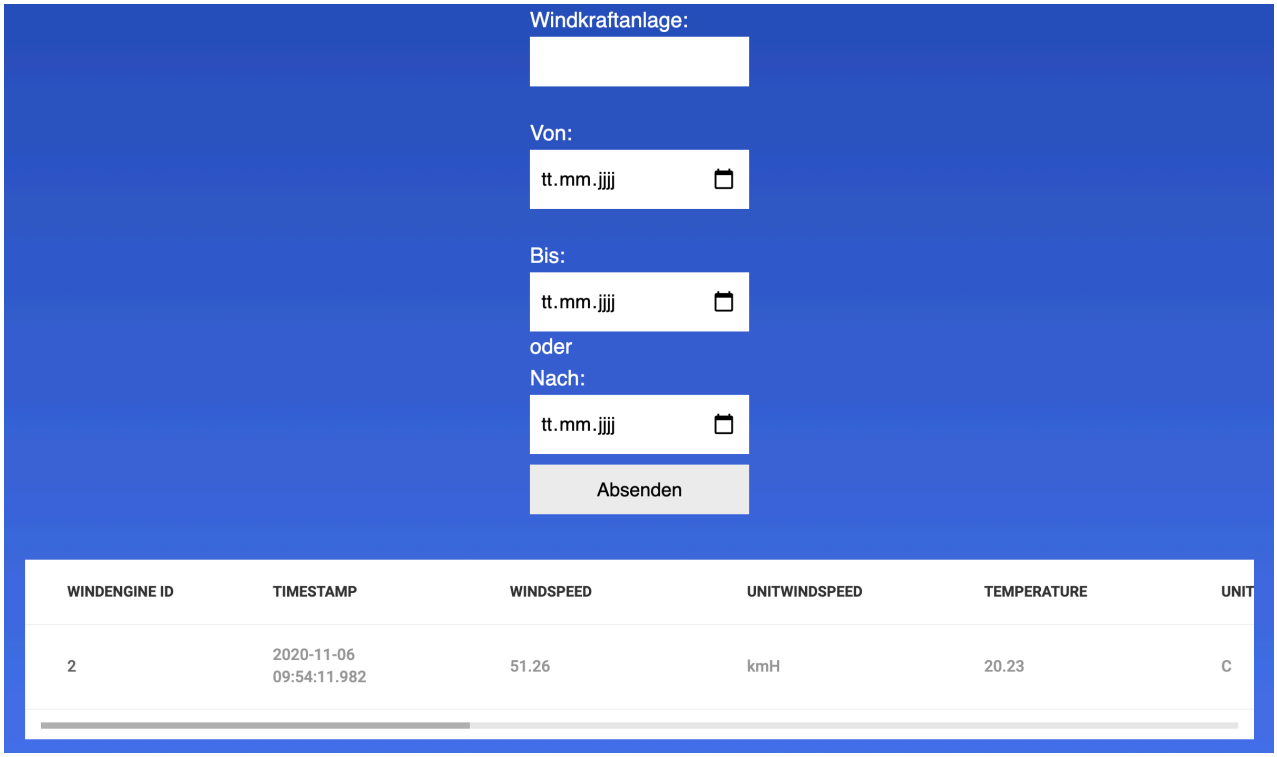
index.pug

```
head
  title Windpark Consumer

script(src='https://ajax.googleapis.com/ajax/libs/jquery/1.11.2/jquery.min.js')
  link(rel='stylesheet'
href='https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/css/bootstrap.min.css')
  link(rel='stylesheet'
href='https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/css/bootstrap-
theme.min.css')

  link(rel='icon' type='image/png' href='images/icons/favicon.ico')
  //
================================================================================
================
  link(rel='stylesheet' type='text/css'
href='vendor/bootstrap/css/bootstrap.min.css')
  //
================================================================================
================
  link(rel='stylesheet' type='text/css' href='fonts/font-awesome-
4.7.0/css/font-awesome.min.css')
  //
================================================================================
================
  link(rel='stylesheet' type='text/css' href='vendor/animate/animate.css')
  //
================================================================================
================
  link(rel='stylesheet' type='text/css' href='vendor/select2/select2.min.css')
  //
================================================================================
================
  link(rel='stylesheet' type='text/css' href='vendor/perfect-scrollbar/perfect-
scrollbar.css')
  //
================================================================================
================
  link(rel='stylesheet' type='text/css' href='css/util.css')
  link(rel='stylesheet' type='text/css' href='css/main.css')
  link(rel='stylesheet' href='css/index.css')


script(src='https://maxcdn.bootstrapcdn.com/bootstrap/3.3.1/js/bootstrap.min.js
')

body
.limiter
```

```
.container-table100
  form(action='/', method='POST').filterForm
    p
      | Windkraftanlage:
      input(type='text', name='windkraftID', value='')

      br
      | Von:
      if von
        input(type='date', name='von', value=von,id="von")
      else
        input(type='date', name='von',id="von")
      br
      |Bis:
      if bis
        input(type='date', name='bis',value=bis,id="bis")
      else
        input(type='date', name='bis',id="bis")

      |       oder
      br
      |Nach:
      if nach
        input(type='date', name='nach',value=nach,id="nach")
      else
        input(type='date', name='nach',id="nach")

    input(type='submit', value='Absenden').subm
  .wrap-table100
    .table100.ver1
      .table100-firstcol
        table
          thead
            tr.row100.head
              th.cell100.column1 Windengine ID
          tbody
            if data
              for item in data
                tr.row100.body
                  each val, key in item
                    if(key == "windengineID")
                      td.cell100.column1= val

      .wrap-table100-nextcols.js-pscroll
        .table100-nextcols
          table
            thead
              tr.row100.head
```

```
                    if keyNames
                      for item in keyNames
                        th.cell100.column2=item

                tbody
                if data
                  for item in data
                    tr.row100.body
                      each val, key in item
                        if(key != "windengineID")
                          td.cell100.column2= val



  //
=============================================================================
================
script(src='vendor/jquery/jquery-3.2.1.min.js')
//
=============================================================================
================
script(src='vendor/bootstrap/js/popper.js')
script(src='vendor/bootstrap/js/bootstrap.min.js')
//
=============================================================================
================
script(src='vendor/select2/select2.min.js')
//
=============================================================================
================
script(src='vendor/perfect-scrollbar/perfect-scrollbar.min.js')
script.
  $('.js-pscroll').each(function(){
  var ps = new PerfectScrollbar(this);
  $(window).on('resize', function(){
  ps.update();
  })
  $(this).on('ps-x-reach-start', function(){
  $(this).parent().find('.table100-firstcol').removeClass('shadow-table100-
firstcol');
  });
  $(this).on('ps-scroll-x', function(){
  $(this).parent().find('.table100-firstcol').addClass('shadow-table100-
firstcol');
  });
  });

script(src='main.js')
```

Tabelle



Erweiterung

Man könnte Sicherheitshalber noch bei der request einen auth key mitschicken der dann bei der REST überprüft wird. Wenn dieser korrekt ist ist es ein validierter Nutzer, wenn nicht dann nicht.