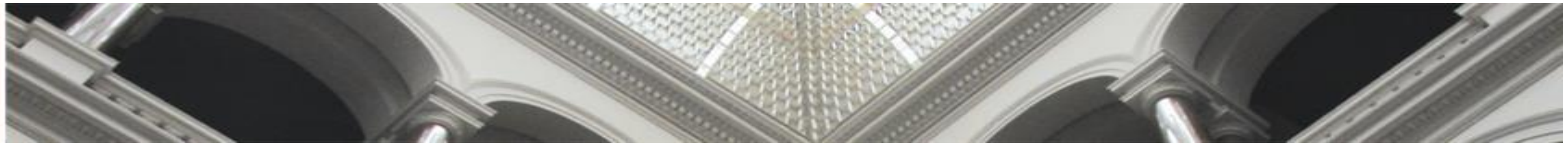
# REST – Übung 3

Implementierung einer REST-Schnittstelle



## Group 46

- Carolin Schwarz, 371802
- Fedor Vitkovskiy, 386458
- Robert Koch, 386471
- Jia Fug Liu, 382333

## Content

REST-Schnittstellenbeschreibung und –Implementierung

- -Beschreibung der Schnittstelle
- -Models
- -Controllers
- -Repositories

Client

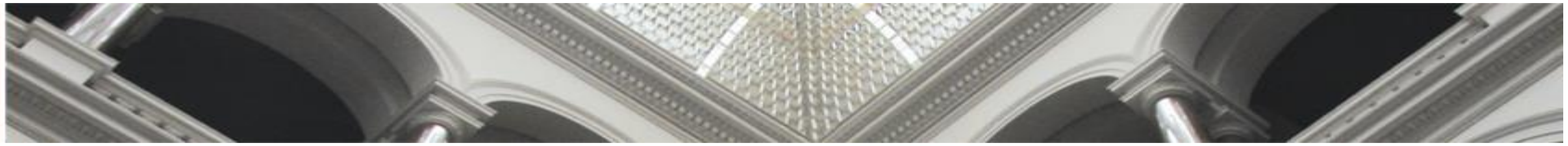
- -Beschreibung und Schnittstelle SMEmu
- -Implementierung
- -Berechnung der Stromstärke für Smartmeter mit ID=…2
- -Screenshots

Group 46

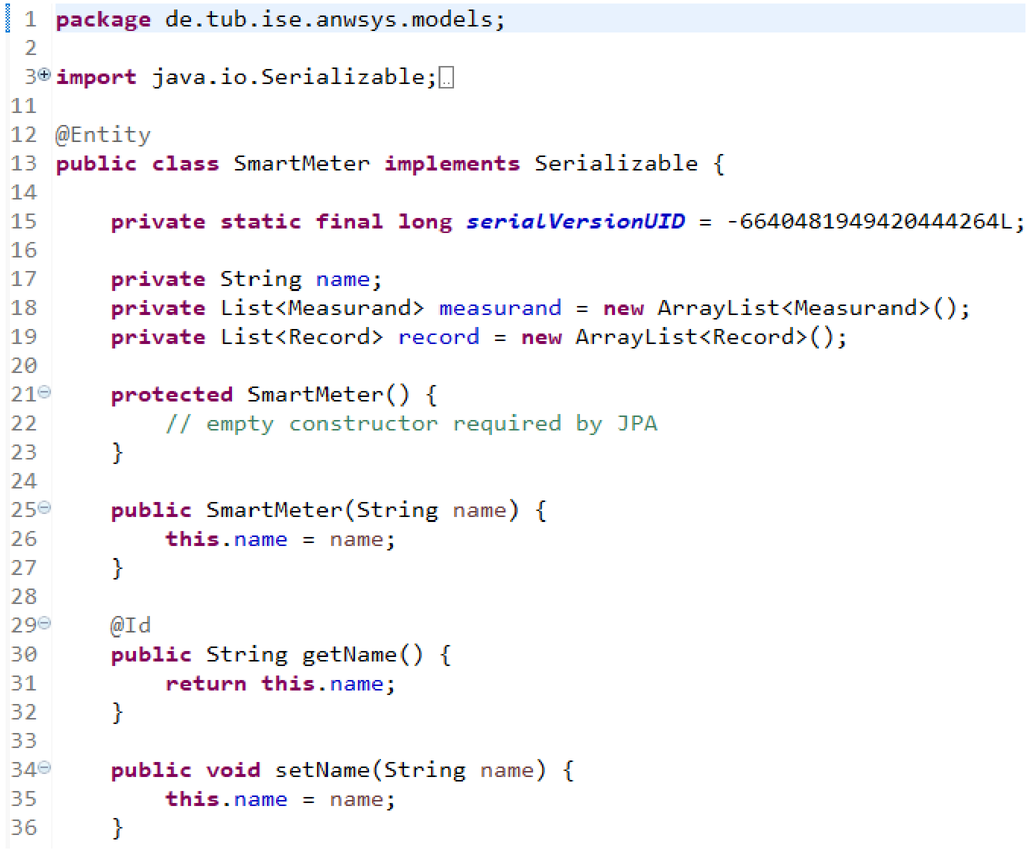
## REST-Schnittstellenbeschreibung

## Models

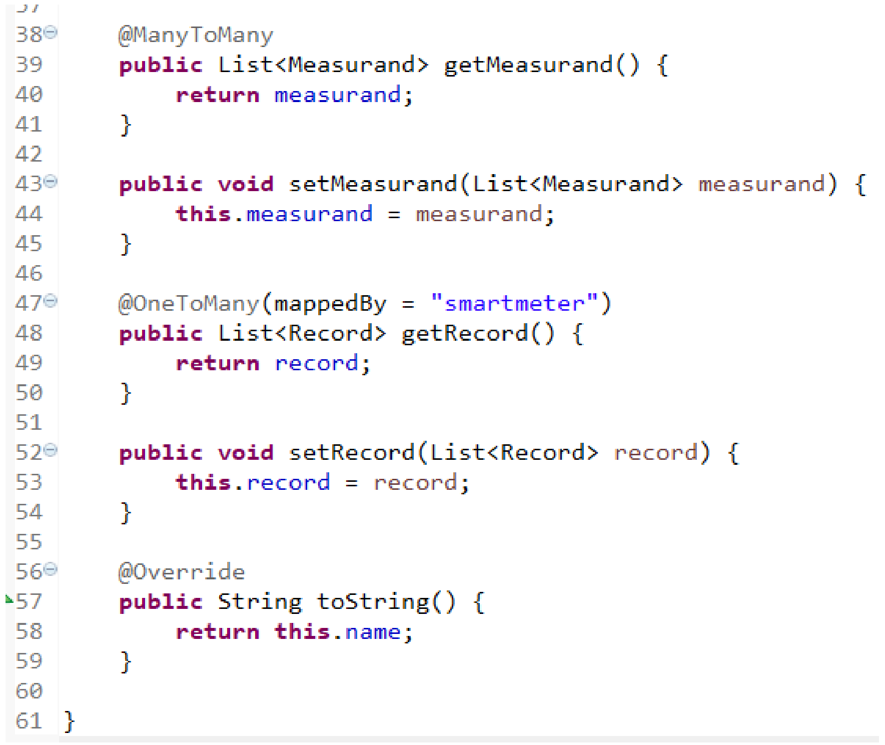
| | |
|---|---|
| Smartmeter: | Repräsentiert ein Smartmeter. Verfügt über eine Liste von Messgrößen (Measurands) und von Ablesungen (Records). Ein Smartmeter steht in einer Many-to-Many Beziehung zu den Measurands und in einer One-to-Many Beziehung zu den Records. |
| Measurand: | Repräsentiert eine Messgröße. Verfügt über eine Liste von Smartmeters und steht zu diesen in einer Many-to-Many Beziehung. |
| Record: | Repräsentiert eine Ablesung. Steht in einer Many-to-One Beziehung sowohl zu den Smartmeters als auch zu den Measurands. |



## Smartmeter

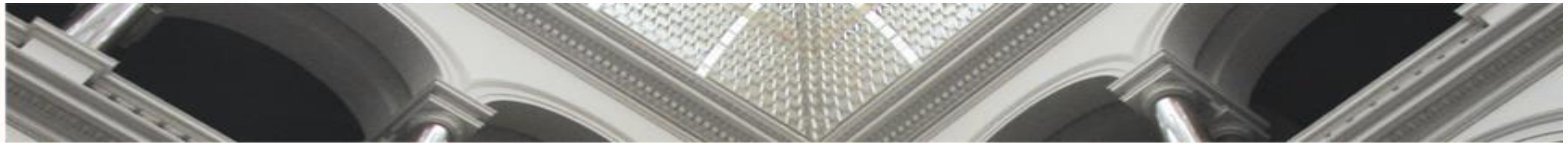


```java
package de.tub.ise.anwsys.models;

import java.io.Serializable;

@Entity
public class SmartMeter implements Serializable {

    private static final long serialVersionUID = -6640481949420444264L;

    private String name;
    private List<Measurand> measurand = new ArrayList<Measurand>();
    private List<Record> record = new ArrayList<Record>();

    protected SmartMeter() {
        // empty constructor required by JPA
    }

    public SmartMeter(String name) {
        this.name = name;
    }

    @Id
    public String getName() {
        return this.name;
    }

    public void setName(String name) {
        this.name = name;
    }
```

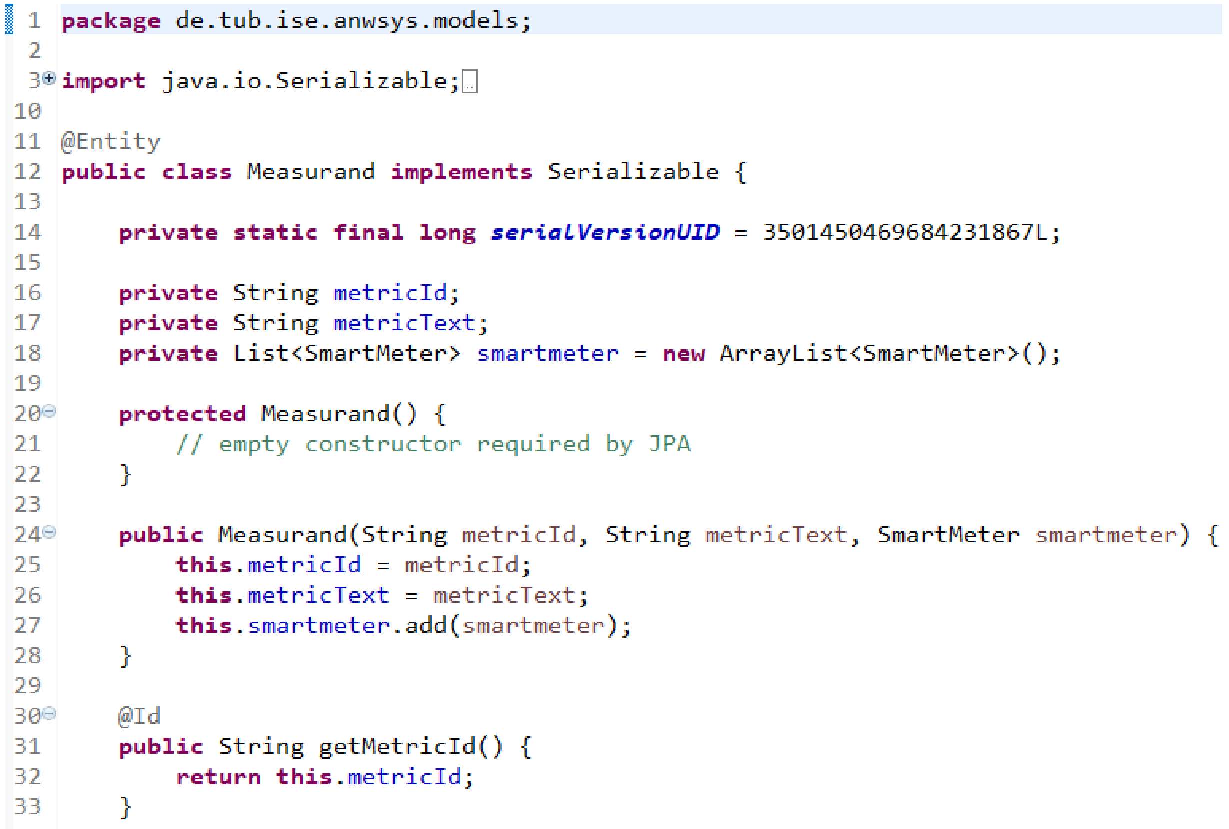


```java
    @ManyToMany
    public List<Measurand> getMeasurand() {
        return measurand;
    }

    public void setMeasurand(List<Measurand> measurand) {
        this.measurand = measurand;
    }

    @OneToMany(mappedBy = "smartmeter")
    public List<Record> getRecord() {
        return record;
    }

    public void setRecord(List<Record> record) {
        this.record = record;
    }

    @Override
    public String toString() {
        return this.name;
    }

}
```

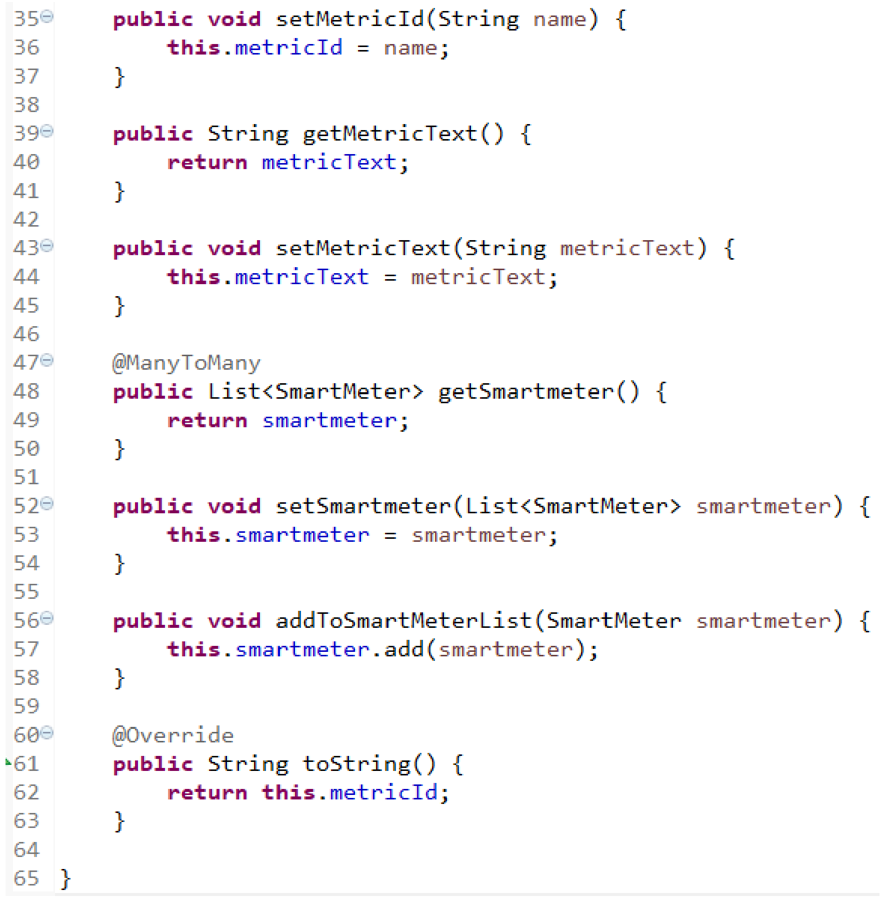
Group 46



## Measurand

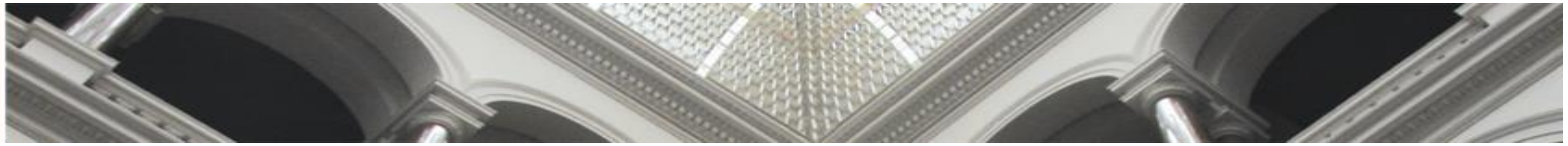


```java
package de.tub.ise.anwsys.models;

import java.io.Serializable;

@Entity
public class Measurand implements Serializable {

    private static final long serialVersionUID = 3501450469684231867L;

    private String metricId;
    private String metricText;
    private List<SmartMeter> smartmeter = new ArrayList<SmartMeter>();

    protected Measurand() {
        // empty constructor required by JPA
    }

    public Measurand(String metricId, String metricText, SmartMeter smartmeter) {
        this.metricId = metricId;
        this.metricText = metricText;
        this.smartmeter.add(smartmeter);
    }

    @Id
    public String getMetricId() {
        return this.metricId;
    }
```

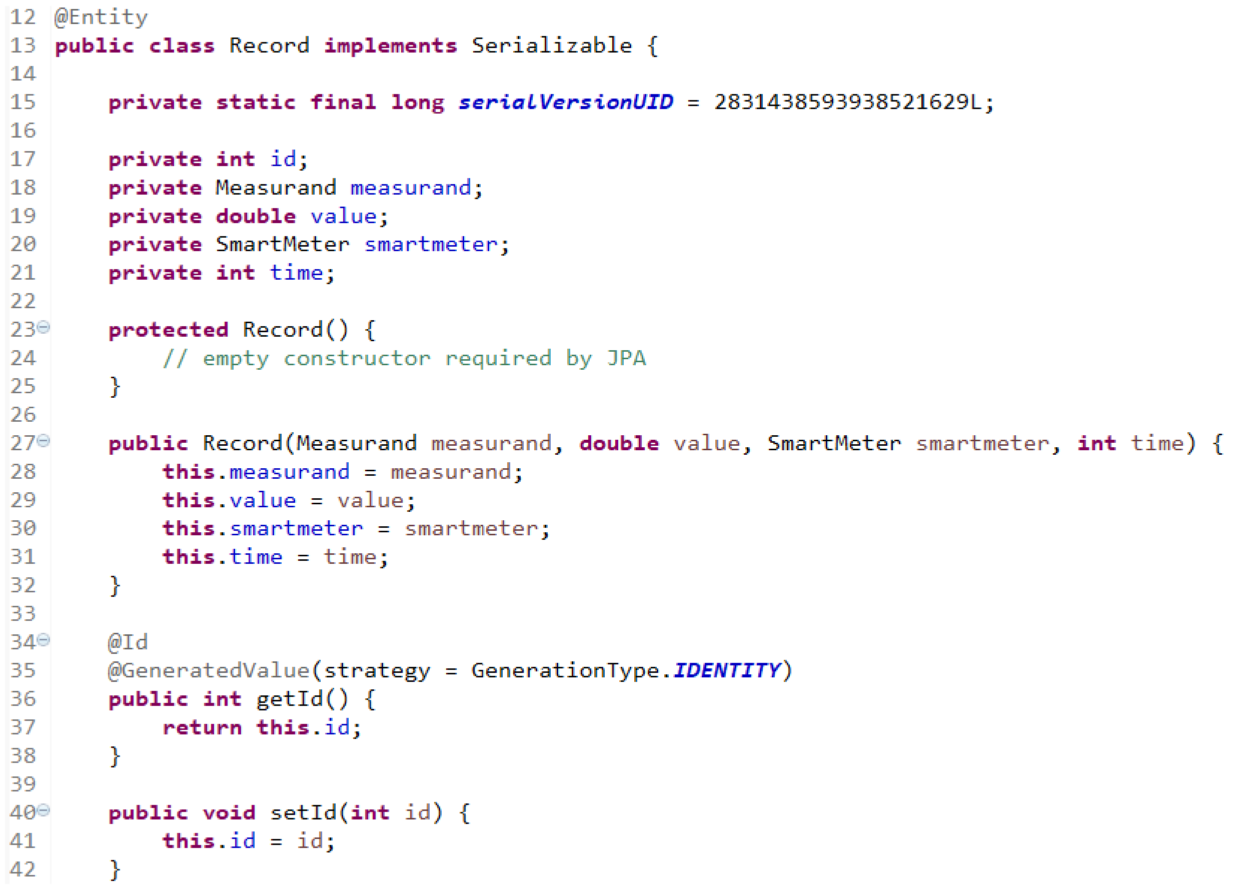


```java
    public void setMetricId(String name) {
        this.metricId = name;
    }

    public String getMetricText() {
        return metricText;
    }

    public void setMetricText(String metricText) {
        this.metricText = metricText;
    }

    @ManyToMany
    public List<SmartMeter> getSmartmeter() {
        return smartmeter;
    }

    public void setSmartmeter(List<SmartMeter> smartmeter) {
        this.smartmeter = smartmeter;
    }

    public void addToSmartMeterList(SmartMeter smartmeter) {
        this.smartmeter.add(smartmeter);
    }

    @Override
    public String toString() {
        return this.metricId;
    }

}
```

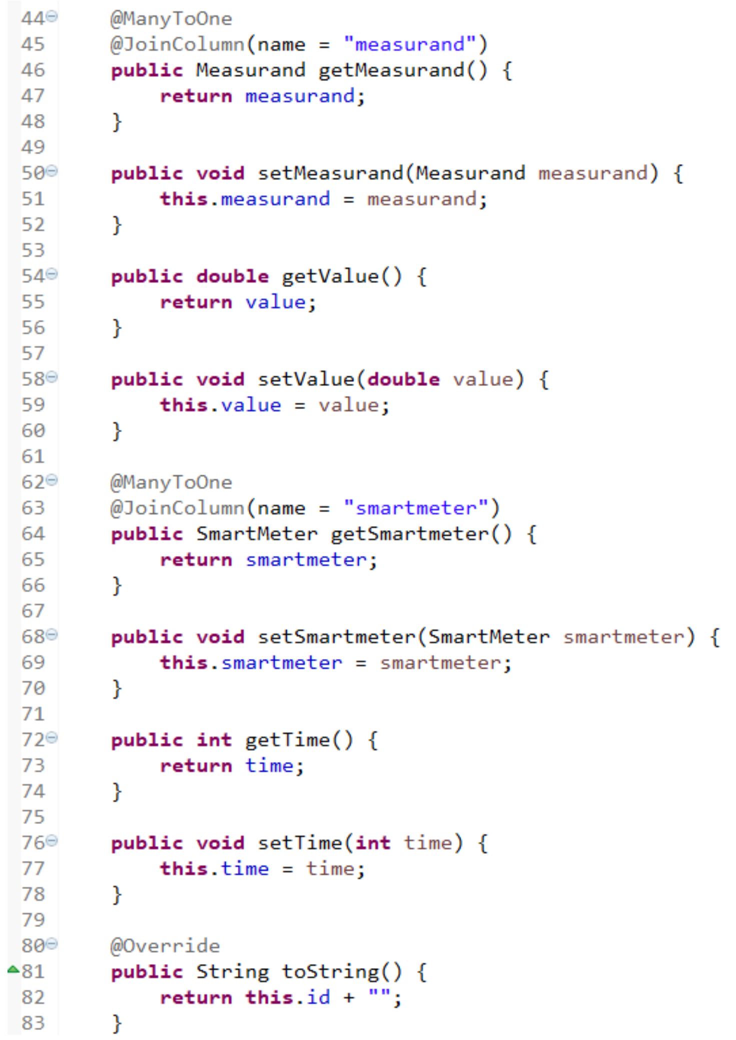
Group 46



# Record



```java
@Entity
public class Record implements Serializable {

    private static final long serialVersionUID = 2831438593938521629L;

    private int id;
    private Measurand measurand;
    private double value;
    private SmartMeter smartmeter;
    private int time;

    protected Record() {
        // empty constructor required by JPA
    }

    public Record(Measurand measurand, double value, SmartMeter smartmeter, int time) {
        this.measurand = measurand;
        this.value = value;
        this.smartmeter = smartmeter;
        this.time = time;
    }

    @Id
    @GeneratedValue(strategy = GenerationType.IDENTITY)
    public int getId() {
        return this.id;
    }

    public void setId(int id) {
        this.id = id;
    }
```

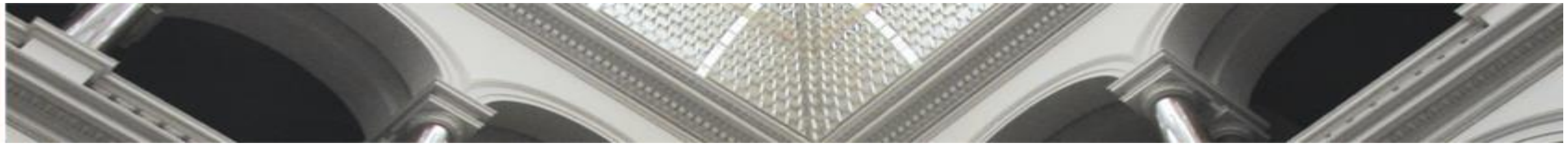


```java
    @ManyToOne
    @JoinColumn(name = "measurand")
    public Measurand getMeasurand() {
        return measurand;
    }

    public void setMeasurand(Measurand measurand) {
        this.measurand = measurand;
    }

    public double getValue() {
        return value;
    }

    public void setValue(double value) {
        this.value = value;
    }

    @ManyToOne
    @JoinColumn(name = "smartmeter")
    public SmartMeter getSmartmeter() {
        return smartmeter;
    }

    public void setSmartmeter(SmartMeter smartmeter) {
        this.smartmeter = smartmeter;
    }

    public int getTime() {
        return time;
    }

    public void setTime(int time) {
        this.time = time;
    }

    @Override
    public String toString() {
        return this.id + "";
    }
```

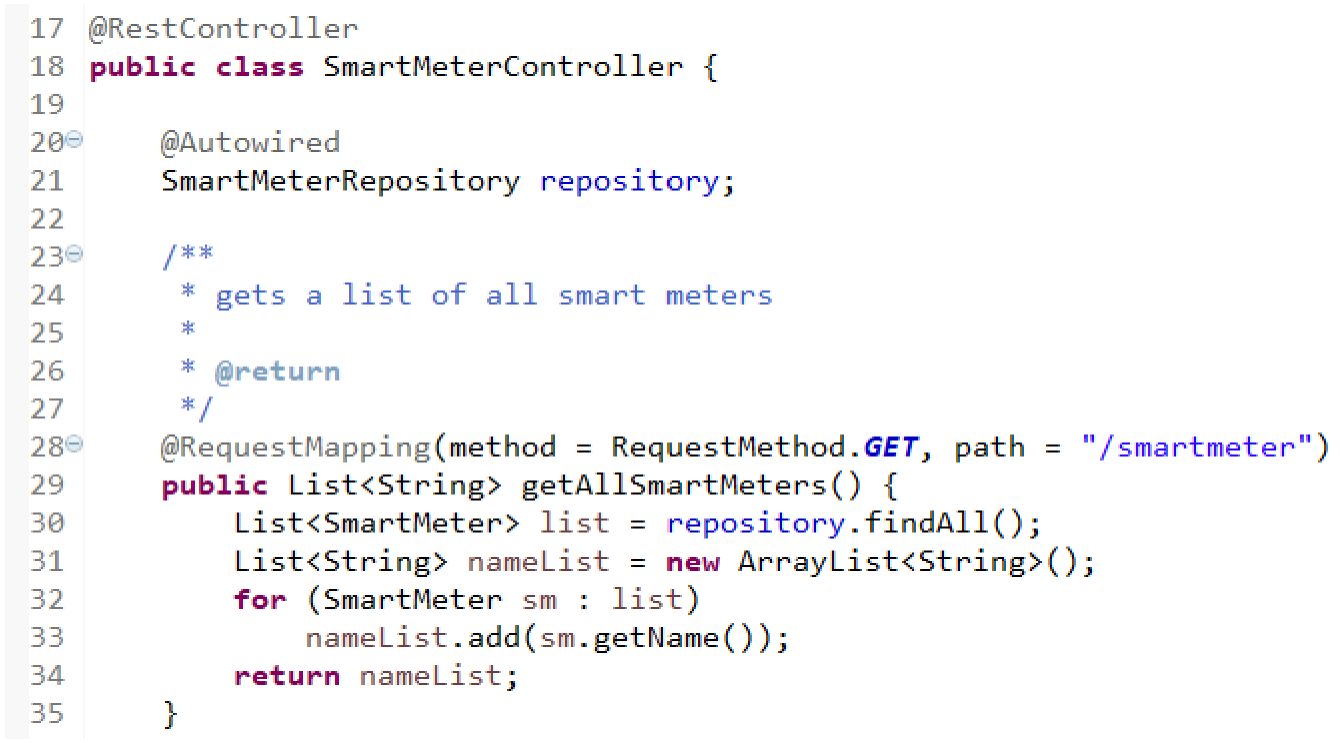
Group 46

## Controllers

| | |
|---|---|
| SmartmeterController: | Bietet eine GET-Methode("/smartmeter") zur Ausgabe einer Liste aller Smartmeter an und eine POST-Methode("/smartmeter"), um neue Smartmeter zu erstellen. |
| MeasurandController: | Bietet eine GET-Methode("/smartmeter/{smartmeter}") zur Ausgabe einer Map an, die die Zuordung von Measurands zu Smartmetern angibt und eine POST-Methode("/smartmeter/{smartmeter}"), die neue Measurands erstellt oder zu einem Smartmeter hinzufügt. |
| RecordController: | Bietet eine GET-Methode("/smartmeter/{smartmeter}/record" ) an, die eine Map zurückgibt, welche die Records eines Smartmeters nach Zeit gruppiert zurückgibt.<br>Zudem eine POST-Methode("/smartmeter/{smartmeter}/record"), welche neue Records erstellt.<br>Eine weitere GET-Methode("/smartmeter/{smartmeter}/record/{metric}") gibt eine Map der Records eines Smartmeters für einen bestimmten Measurand gruppiert nach Zeit zurück. |

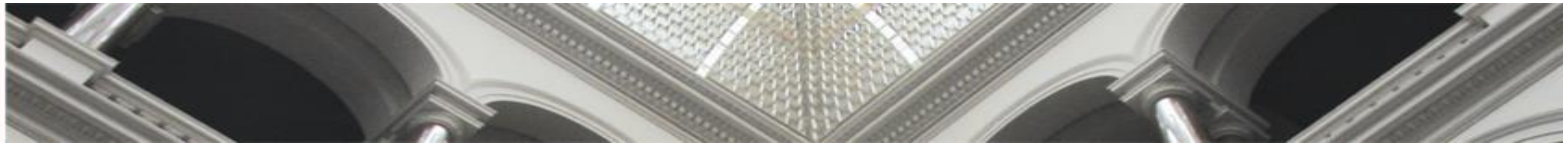


# SmartmeterController(1)

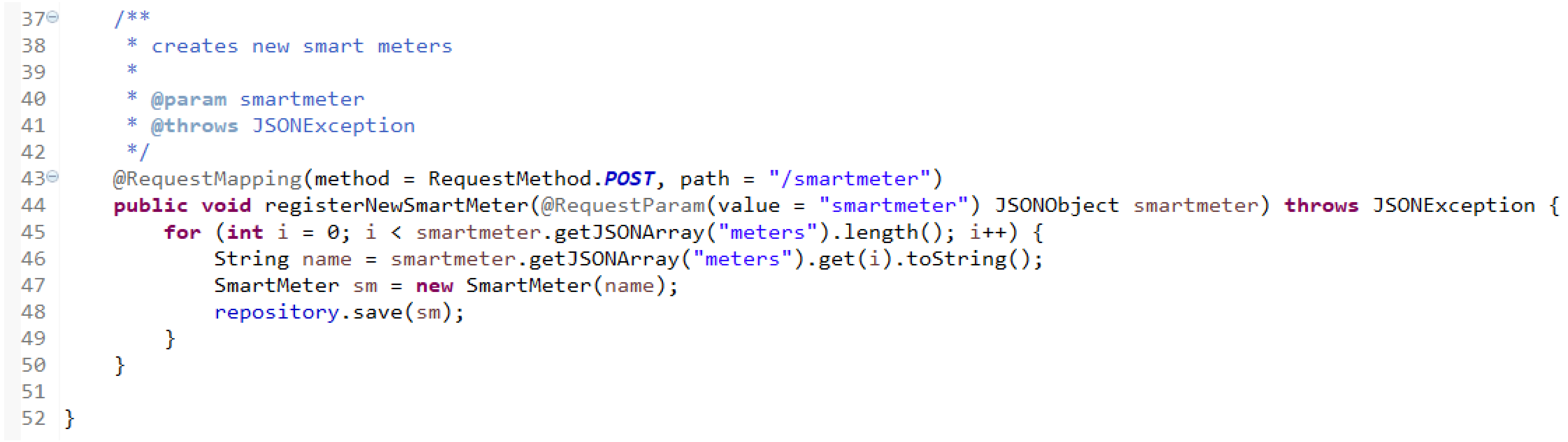


```java
@RestController
public class SmartMeterController {

    @Autowired
    SmartMeterRepository repository;

    /**
     * gets a list of all smart meters
     *
     * @return
     */
    @RequestMapping(method = RequestMethod.GET, path = "/smartmeter")
    public List<String> getAllSmartMeters() {
        List<SmartMeter> list = repository.findAll();
        List<String> nameList = new ArrayList<String>();
        for (SmartMeter sm : list)
            nameList.add(sm.getName());
        return nameList;
    }
```

Group 46



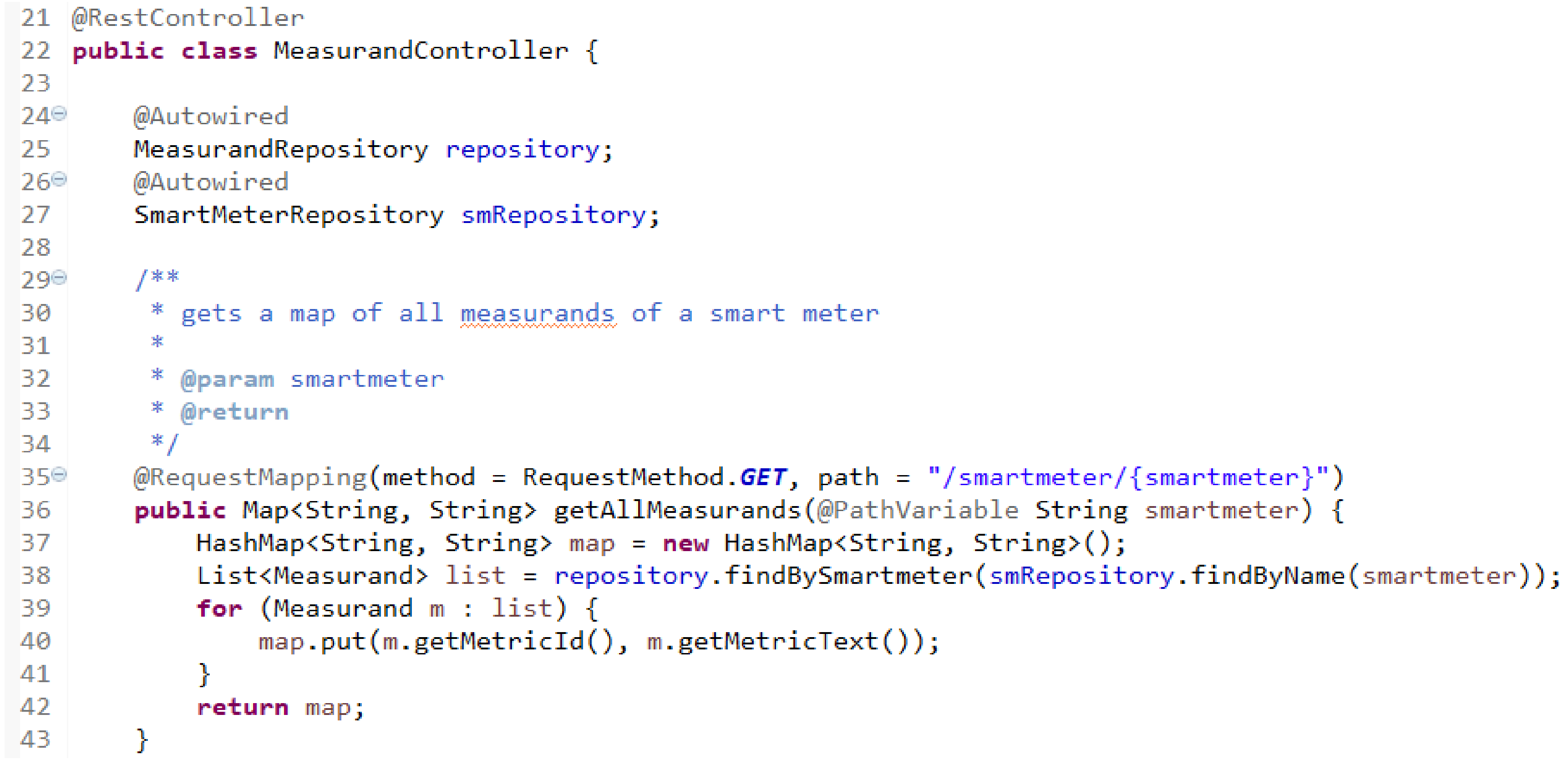
## SmartmeterController(2)



```java
    /**
     * creates new smart meters
     *
     * @param smartmeter
     * @throws JSONException
     */
    @RequestMapping(method = RequestMethod.POST, path = "/smartmeter")
    public void registerNewSmartMeter(@RequestParam(value = "smartmeter") JSONObject smartmeter) throws JSONException {
        for (int i = 0; i < smartmeter.getJSONArray("meters").length(); i++) {
            String name = smartmeter.getJSONArray("meters").get(i).toString();
            SmartMeter sm = new SmartMeter(name);
            repository.save(sm);
        }
    }

}
```

Group 46

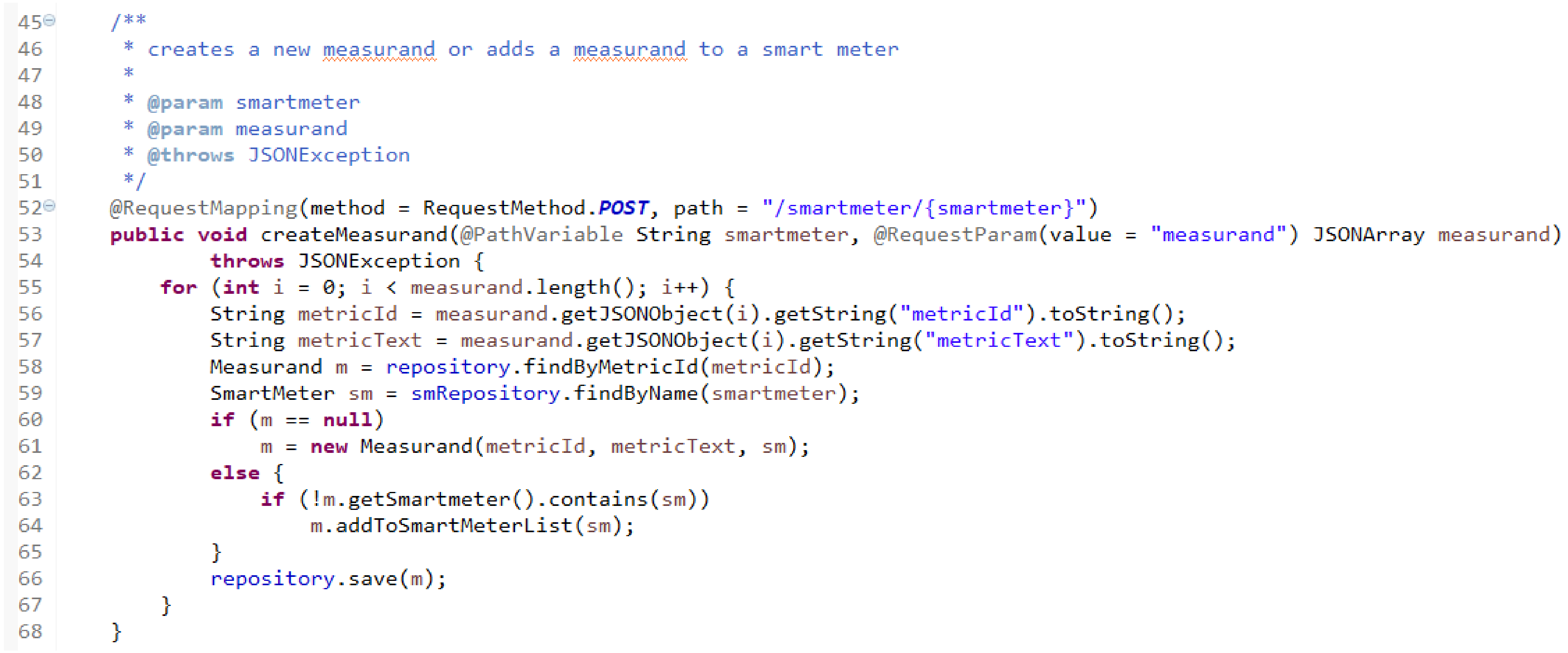
## MeasurandController(1)



```java
@RestController
public class MeasurandController {

    @Autowired
    MeasurandRepository repository;
    @Autowired
    SmartMeterRepository smRepository;

    /**
     * gets a map of all measurands of a smart meter
     *
     * @param smartmeter
     * @return
     */
    @RequestMapping(method = RequestMethod.GET, path = "/smartmeter/{smartmeter}")
    public Map<String, String> getAllMeasurands(@PathVariable String smartmeter) {
        HashMap<String, String> map = new HashMap<String, String>();
        List<Measurand> list = repository.findBySmartmeter(smRepository.findByName(smartmeter));
        for (Measurand m : list) {
            map.put(m.getMetricId(), m.getMetricText());
        }
        return map;
    }
```

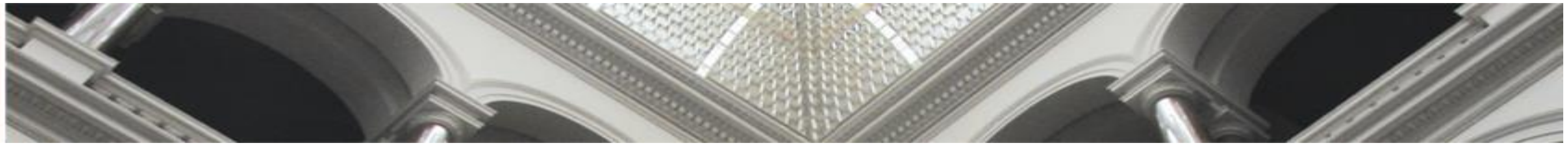
Group 46

# MeasurandController(2)

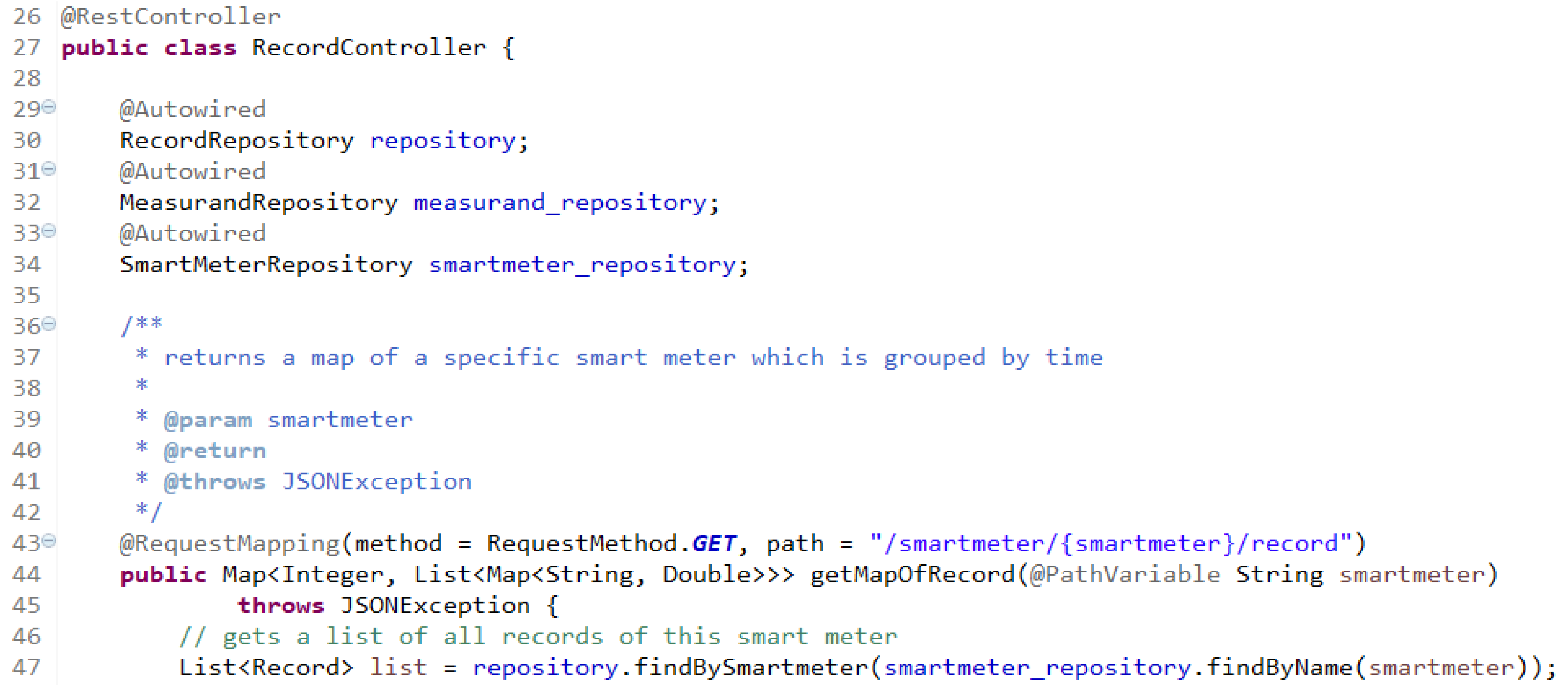


```java
    /**
     * creates a new measurand or adds a measurand to a smart meter
     *
     * @param smartmeter
     * @param measurand
     * @throws JSONException
     */
    @RequestMapping(method = RequestMethod.POST, path = "/smartmeter/{smartmeter}")
    public void createMeasurand(@PathVariable String smartmeter, @RequestParam(value = "measurand") JSONArray measurand)
            throws JSONException {
        for (int i = 0; i < measurand.length(); i++) {
            String metricId = measurand.getJSONObject(i).getString("metricId").toString();
            String metricText = measurand.getJSONObject(i).getString("metricText").toString();
            Measurand m = repository.findByMetricId(metricId);
            SmartMeter sm = smRepository.findByName(smartmeter);
            if (m == null)
                m = new Measurand(metricId, metricText, sm);
            else {
                if (!m.getSmartmeter().contains(sm))
                    m.addToSmartMeterList(sm);
            }
            repository.save(m);
        }
    }
```

Group 46

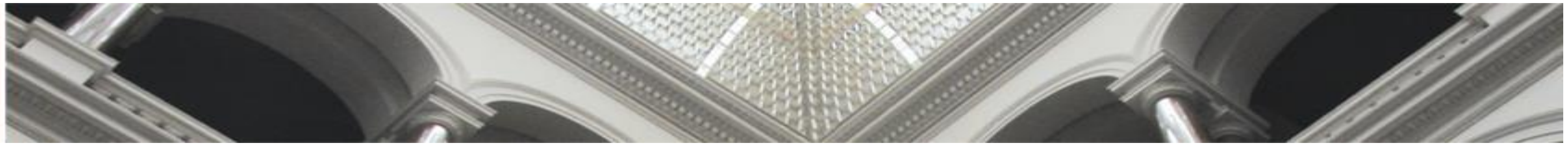


## RecordController(1)

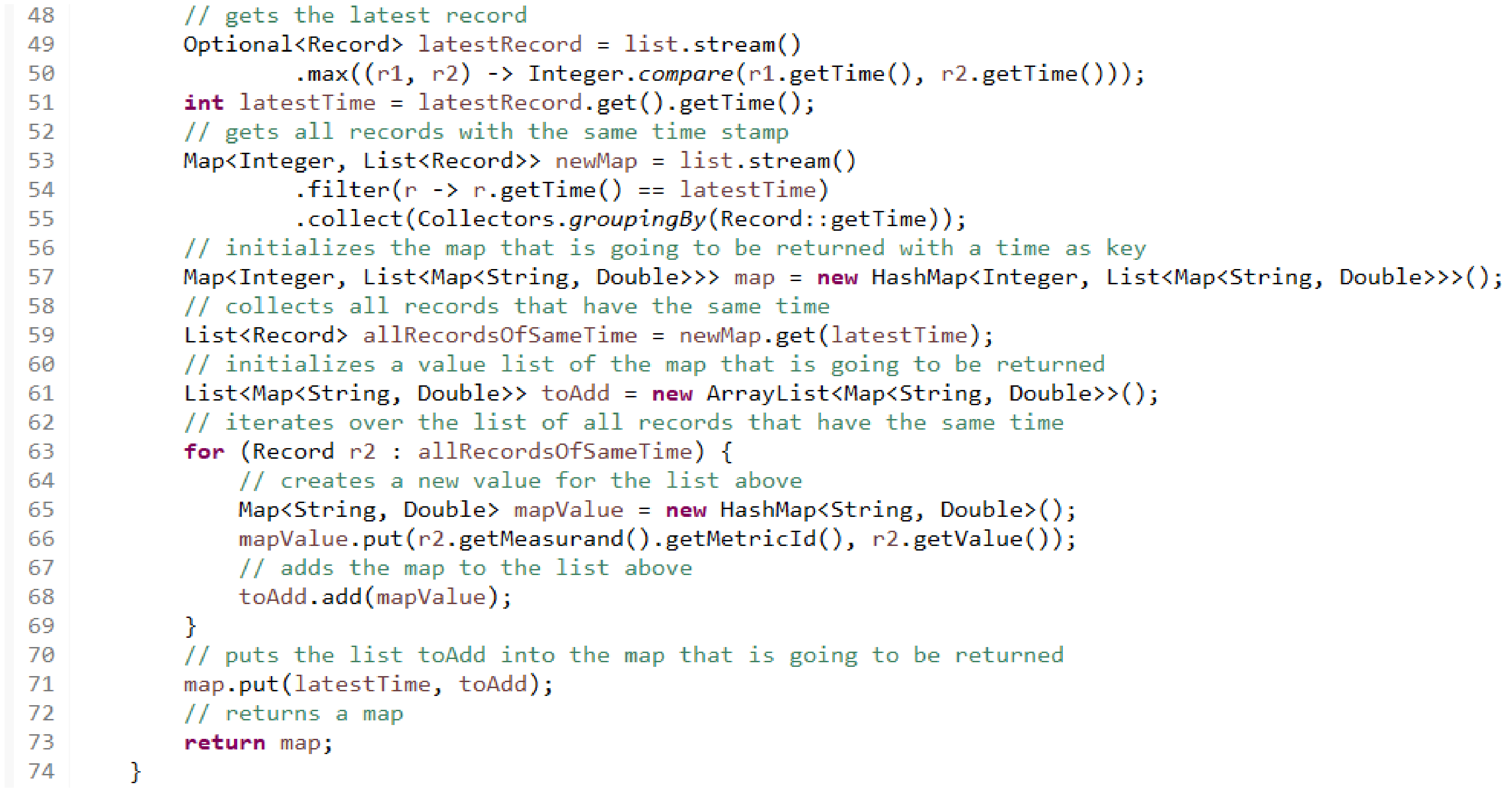


```java
@RestController
public class RecordController {

    @Autowired
    RecordRepository repository;
    @Autowired
    MeasurandRepository measurand_repository;
    @Autowired
    SmartMeterRepository smartmeter_repository;

    /**
     * returns a map of a specific smart meter which is grouped by time
     *
     * @param smartmeter
     * @return
     * @throws JSONException
     */
    @RequestMapping(method = RequestMethod.GET, path = "/smartmeter/{smartmeter}/record")
    public Map<Integer, List<Map<String, Double>>> getMapOfRecord(@PathVariable String smartmeter)
            throws JSONException {
        // gets a list of all records of this smart meter
        List<Record> list = repository.findBySmartmeter(smartmeter_repository.findByName(smartmeter));
```

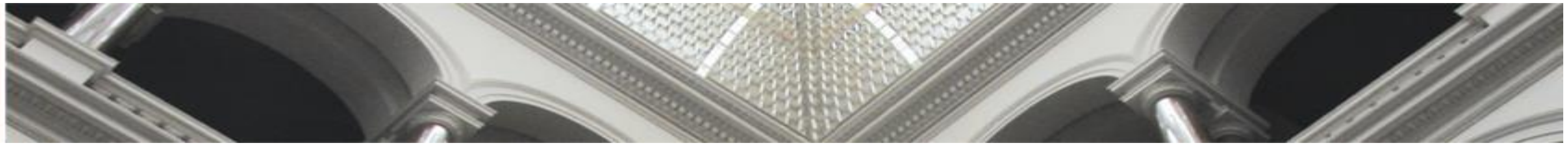
Group 46



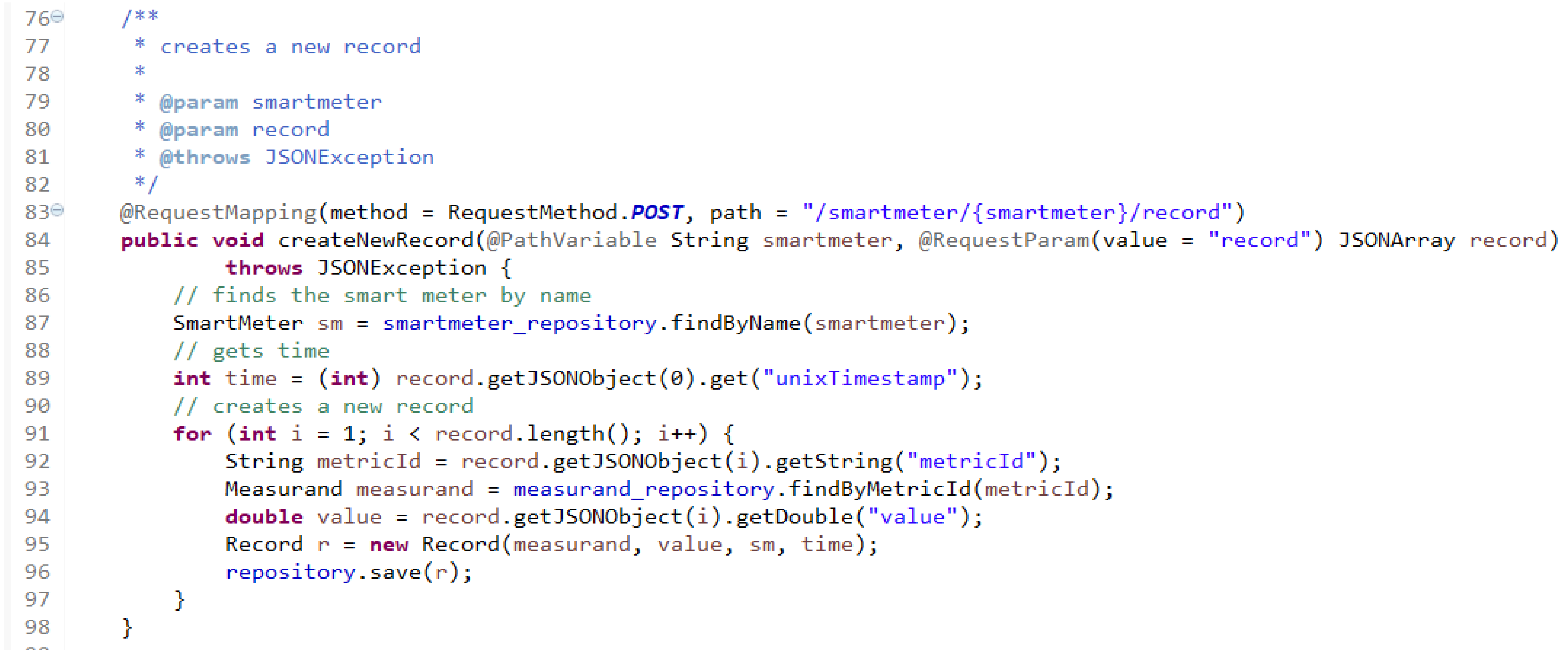
## RecordController(2)



```java
        // gets the latest record
        Optional<Record> latestRecord = list.stream()
                .max((r1, r2) -> Integer.compare(r1.getTime(), r2.getTime()));
        int latestTime = latestRecord.get().getTime();
        // gets all records with the same time stamp
        Map<Integer, List<Record>> newMap = list.stream()
                .filter(r -> r.getTime() == latestTime)
                .collect(Collectors.groupingBy(Record::getTime));
        // initializes the map that is going to be returned with a time as key
        Map<Integer, List<Map<String, Double>>> map = new HashMap<Integer, List<Map<String, Double>>>();
        // collects all records that have the same time
        List<Record> allRecordsOfSameTime = newMap.get(latestTime);
        // initializes a value list of the map that is going to be returned
        List<Map<String, Double>> toAdd = new ArrayList<Map<String, Double>>();
        // iterates over the list of all records that have the same time
        for (Record r2 : allRecordsOfSameTime) {
            // creates a new value for the list above
            Map<String, Double> mapValue = new HashMap<String, Double>();
            mapValue.put(r2.getMeasurand().getMetricId(), r2.getValue());
            // adds the map to the list above
            toAdd.add(mapValue);
        }
        // puts the list toAdd into the map that is going to be returned
        map.put(latestTime, toAdd);
        // returns a map
        return map;
    }
```

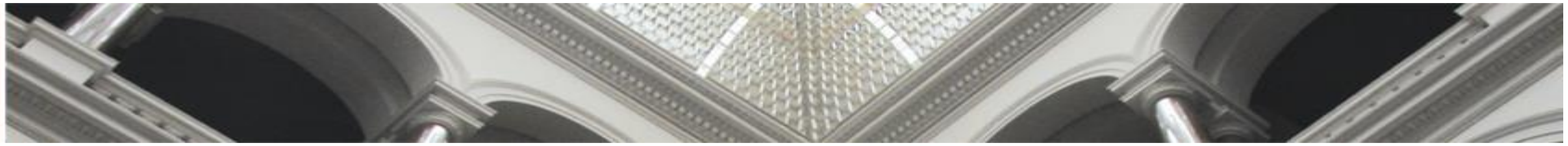


## RecordController(3)

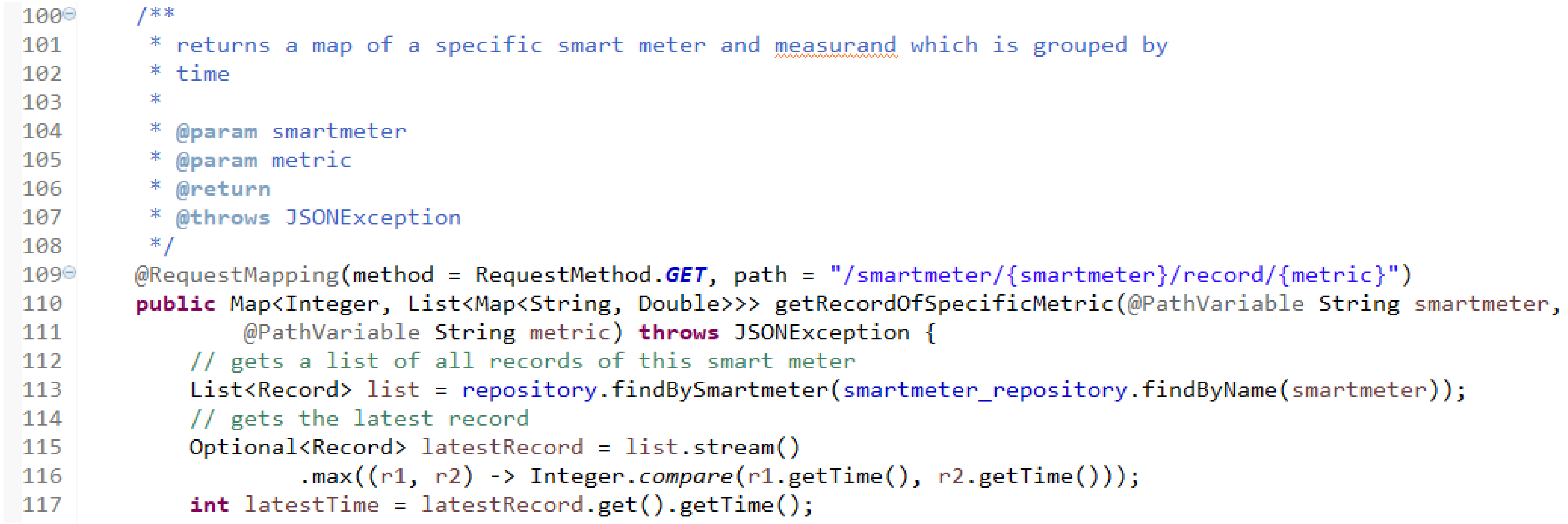


```java
    /**
     * creates a new record
     *
     * @param smartmeter
     * @param record
     * @throws JSONException
     */
    @RequestMapping(method = RequestMethod.POST, path = "/smartmeter/{smartmeter}/record")
    public void createNewRecord(@PathVariable String smartmeter, @RequestParam(value = "record") JSONArray record)
            throws JSONException {
        // finds the smart meter by name
        SmartMeter sm = smartmeter_repository.findByName(smartmeter);
        // gets time
        int time = (int) record.getJSONObject(0).get("unixTimestamp");
        // creates a new record
        for (int i = 1; i < record.length(); i++) {
            String metricId = record.getJSONObject(i).getString("metricId");
            Measurand measurand = measurand_repository.findByMetricId(metricId);
            double value = record.getJSONObject(i).getDouble("value");
            Record r = new Record(measurand, value, sm, time);
            repository.save(r);
        }
    }
```

Group 46

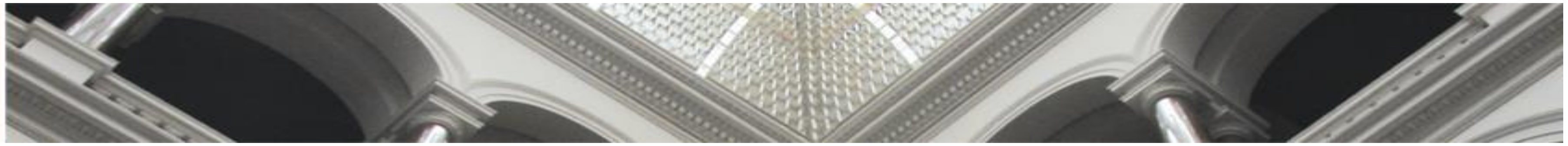


# RecordController(4)

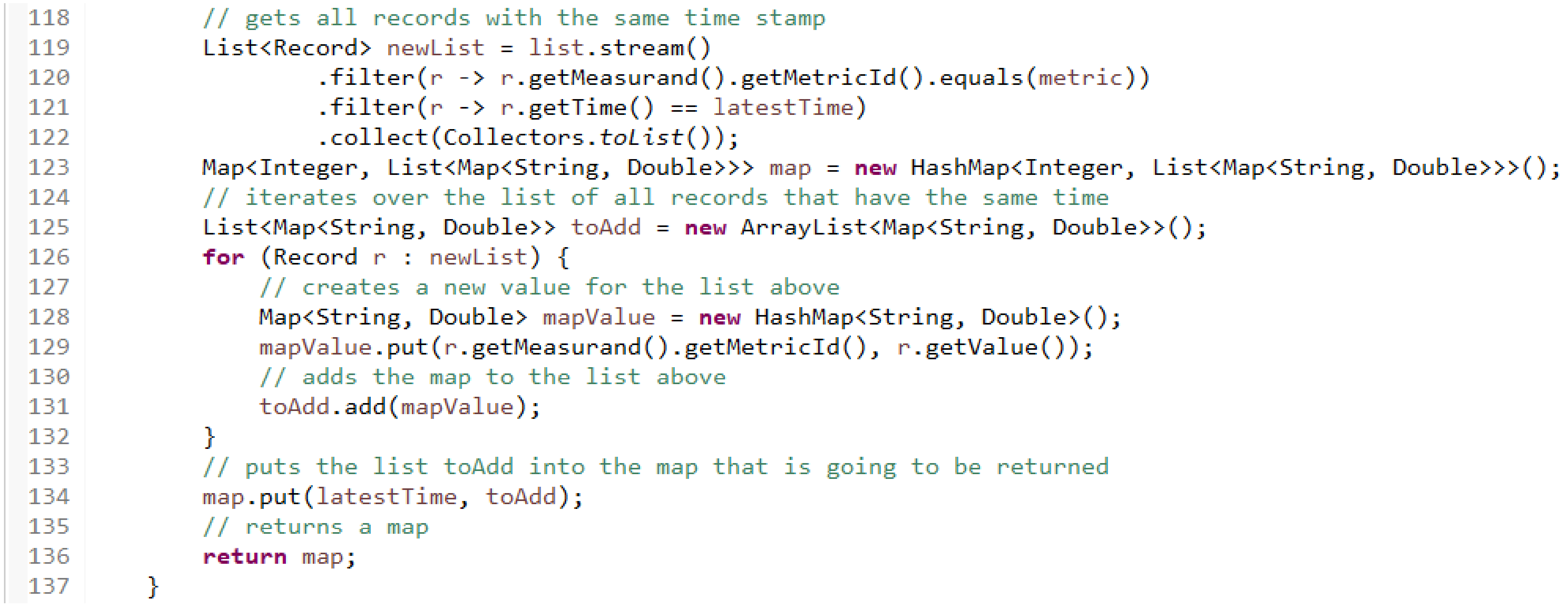


```
    /**
     * returns a map of a specific smart meter and measurand which is grouped by
     * time
     *
     * @param smartmeter
     * @param metric
     * @return
     * @throws JSONException
     */
    @RequestMapping(method = RequestMethod.GET, path = "/smartmeter/{smartmeter}/record/{metric}")
    public Map<Integer, List<Map<String, Double>>> getRecordOfSpecificMetric(@PathVariable String smartmeter,
            @PathVariable String metric) throws JSONException {
        // gets a list of all records of this smart meter
        List<Record> list = repository.findBySmartmeter(smartmeter_repository.findByName(smartmeter));
        // gets the latest record
        Optional<Record> latestRecord = list.stream()
                .max((r1, r2) -> Integer.compare(r1.getTime(), r2.getTime()));
        int latestTime = latestRecord.get().getTime();
```

Group 46

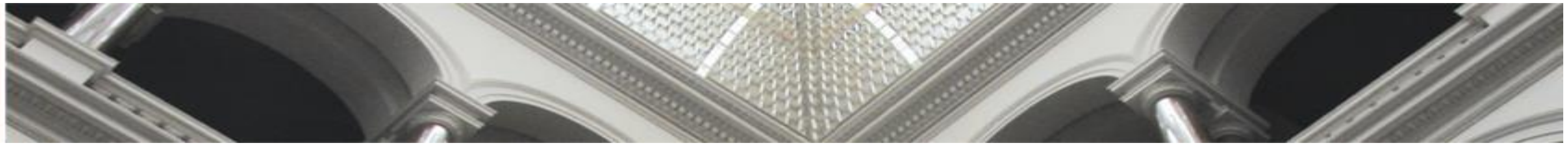


## RecordController(5)



```java
        // gets all records with the same time stamp
        List<Record> newList = list.stream()
                .filter(r -> r.getMeasurand().getMetricId().equals(metric))
                .filter(r -> r.getTime() == latestTime)
                .collect(Collectors.toList());
        Map<Integer, List<Map<String, Double>>> map = new HashMap<Integer, List<Map<String, Double>>>();
        // iterates over the list of all records that have the same time
        List<Map<String, Double>> toAdd = new ArrayList<Map<String, Double>>();
        for (Record r : newList) {
            // creates a new value for the list above
            Map<String, Double> mapValue = new HashMap<String, Double>();
            mapValue.put(r.getMeasurand().getMetricId(), r.getValue());
            // adds the map to the list above
            toAdd.add(mapValue);
        }
        // puts the list toAdd into the map that is going to be returned
        map.put(latestTime, toAdd);
        // returns a map
        return map;
    }
```

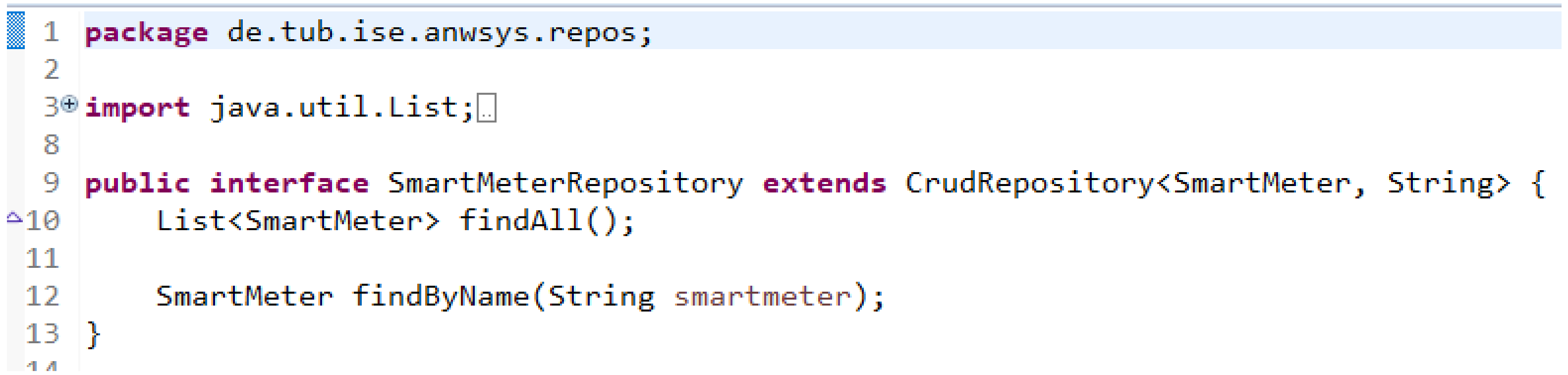
Group 46



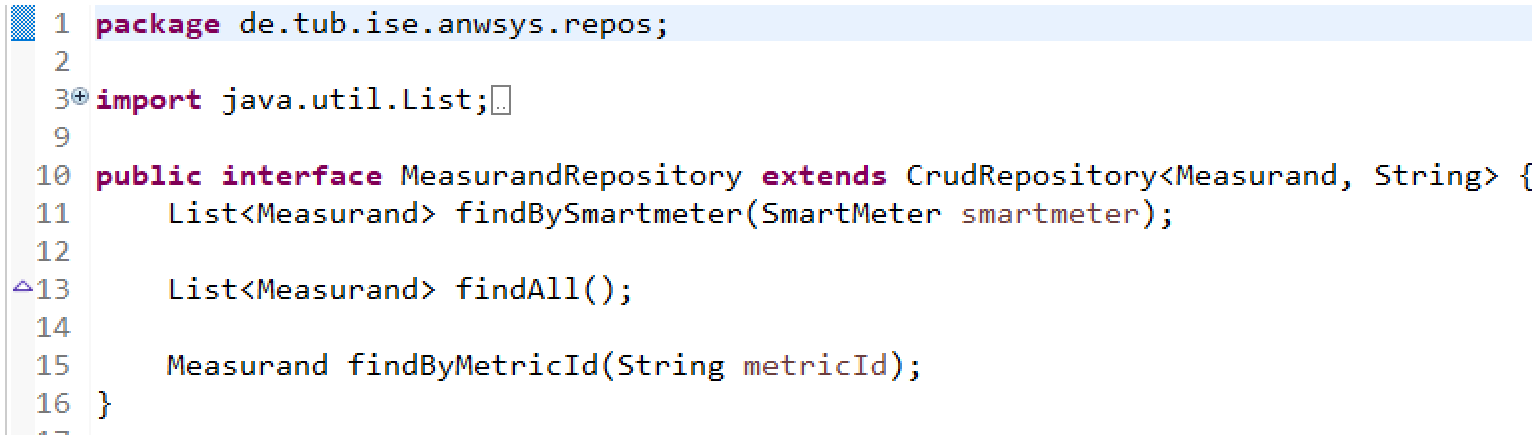
## Repositories

| | |
|---|---|
| SmartmeterRepository: | Bietet Methoden zur Rückgabe aller Smartmeter oder eines spezifischen Smartmeters an. |
| MeasurandRepository: | Bietet Methoden an um alle Measurands, einen bestimmten Measurand, oder alle einem spezifischen Smartmeter zugeordneten Measurands zurückzugeben. |
| RecordRepository: | Bietet Methoden an um alle Records oder alle einem spezifischen Smartmeter zugeordneten Records zurückzugeben. |

## SmartmeterRepository

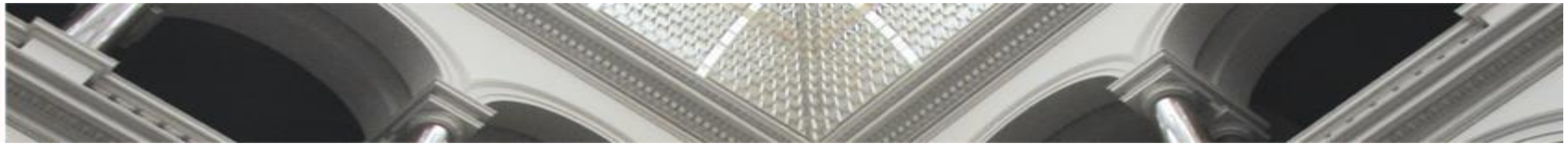

```
package de.tub.ise.anwsys.repos;

import java.util.List;

public interface SmartMeterRepository extends CrudRepository<SmartMeter, String> {
    List<SmartMeter> findAll();

    SmartMeter findByName(String smartmeter);
}
```

# MeasurandRepository

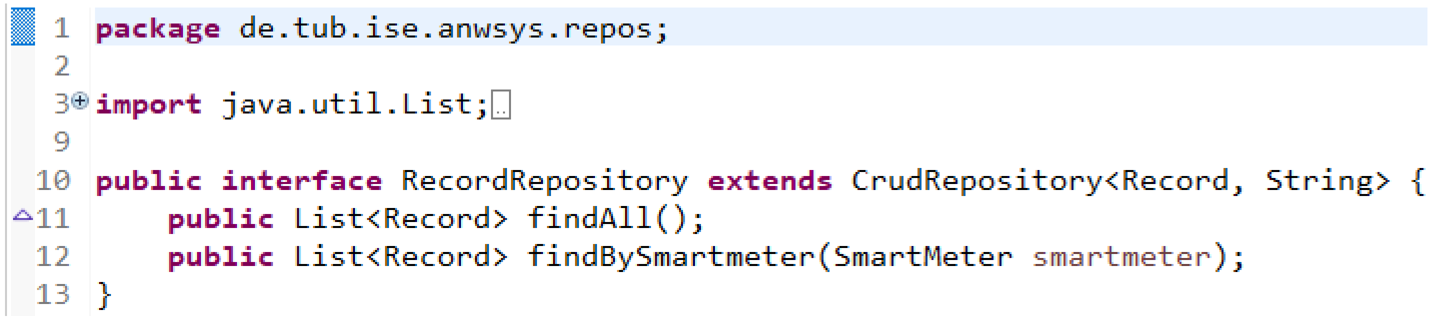


```java
package de.tub.ise.anwsys.repos;

import java.util.List;

public interface MeasurandRepository extends CrudRepository<Measurand, String> {
    List<Measurand> findBySmartmeter(SmartMeter smartmeter);

    List<Measurand> findAll();

    Measurand findByMetricId(String metricId);
}
```

Group 46

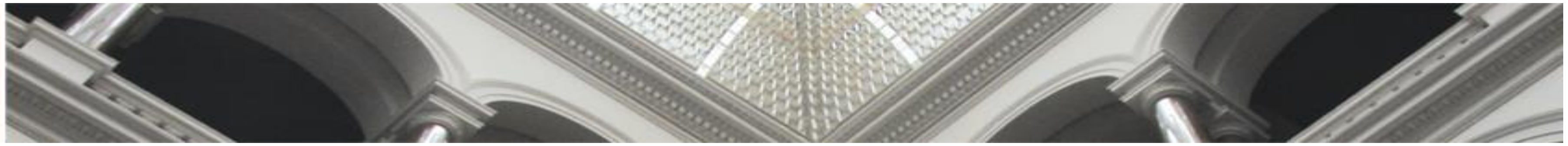


## RecordRepository

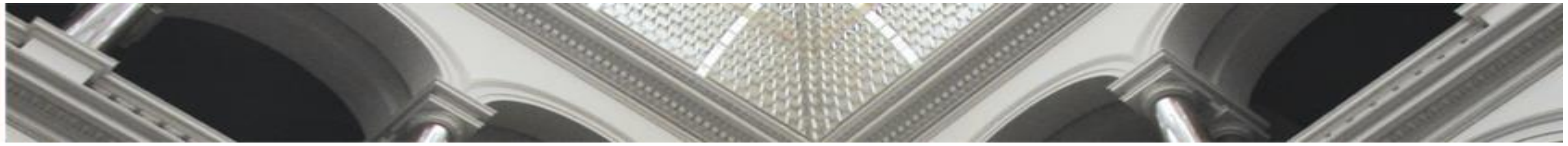


```
package de.tub.ise.anwsys.repos;

import java.util.List;

public interface RecordRepository extends CrudRepository<Record, String> {
    public List<Record> findAll();
    public List<Record> findBySmartmeter(SmartMeter smartmeter);
}
```

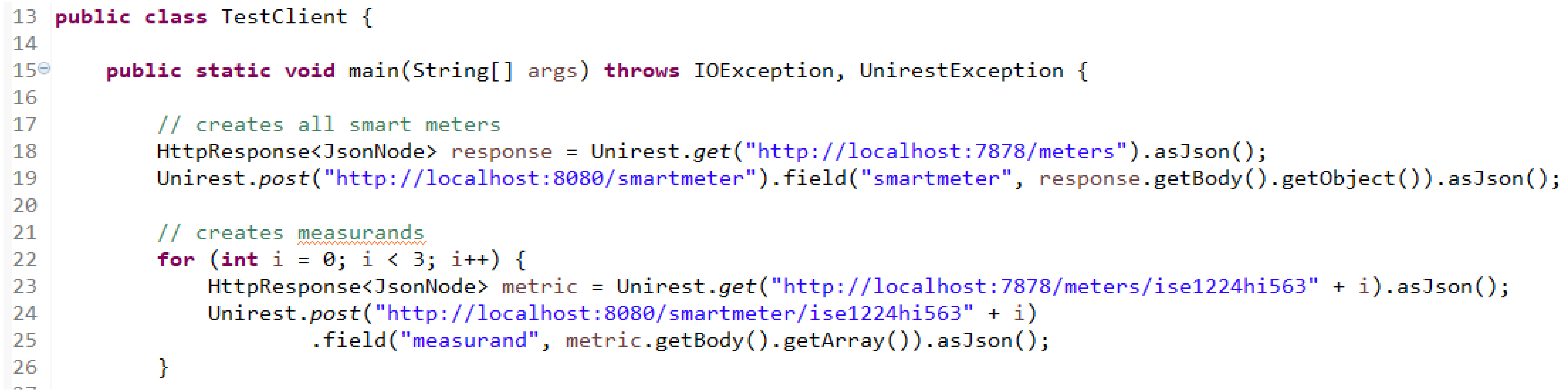
Group 46



## Client



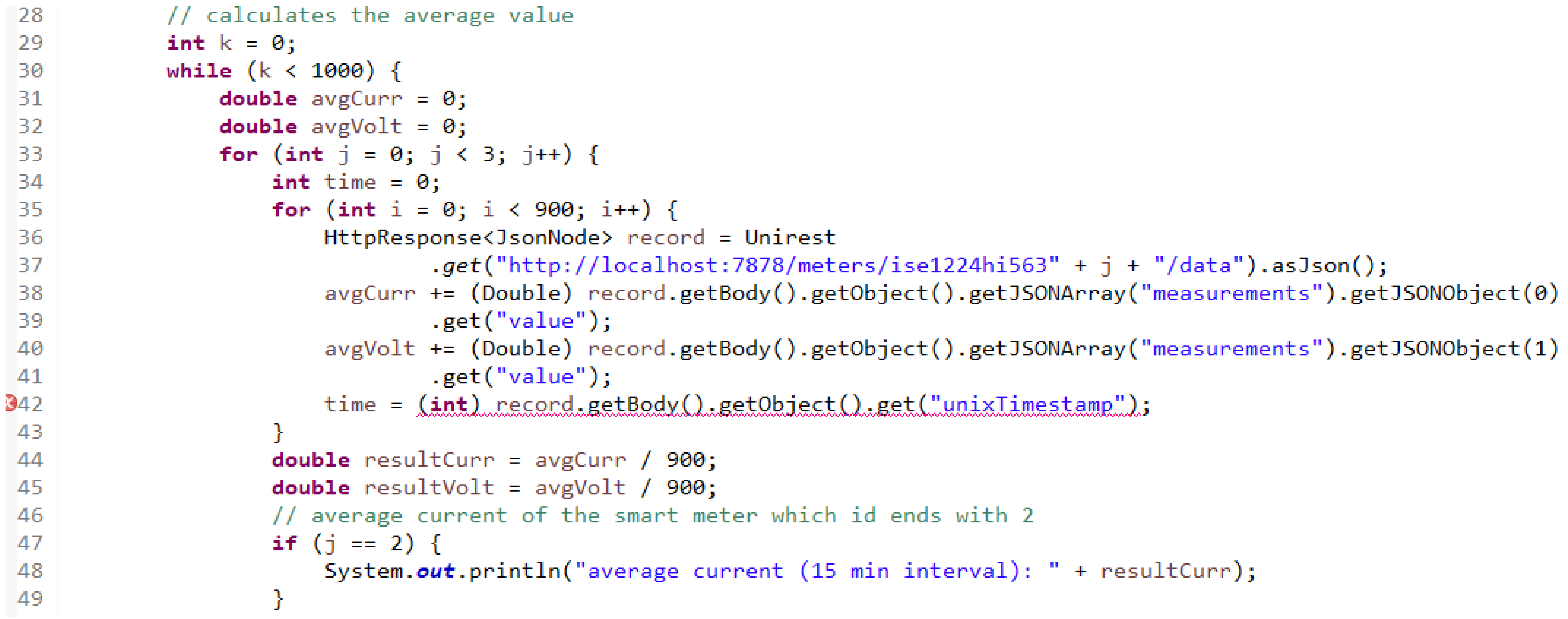
## Client Implementierung(1)



```java
public class TestClient {

    public static void main(String[] args) throws IOException, UnirestException {

        // creates all smart meters
        HttpResponse<JsonNode> response = Unirest.get("http://localhost:7878/meters").asJson();
        Unirest.post("http://localhost:8080/smartmeter").field("smartmeter", response.getBody().getObject()).asJson();

        // creates measurands
        for (int i = 0; i < 3; i++) {
            HttpResponse<JsonNode> metric = Unirest.get("http://localhost:7878/meters/ise1224hi563" + i).asJson();
            Unirest.post("http://localhost:8080/smartmeter/ise1224hi563" + i)
                    .field("measurand", metric.getBody().getArray()).asJson();
        }
```

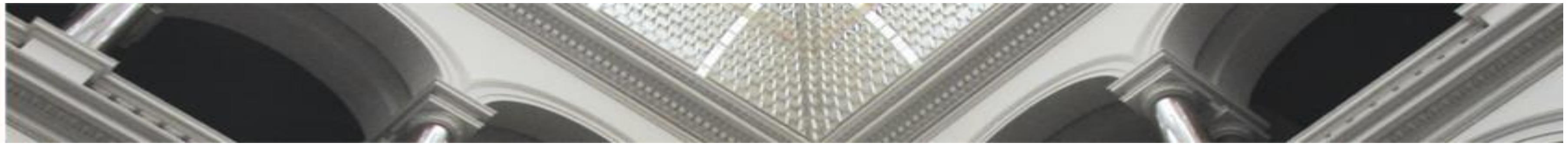
Group 46

# Client Implementierung(2)

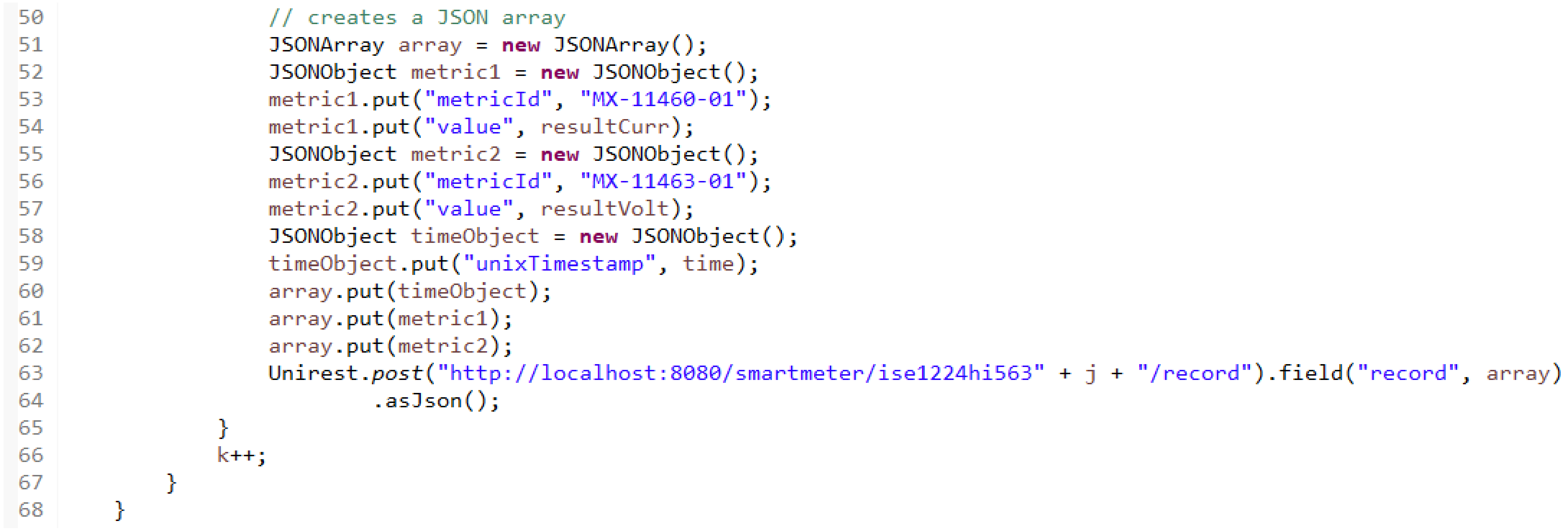


```java
        // calculates the average value
        int k = 0;
        while (k < 1000) {
            double avgCurr = 0;
            double avgVolt = 0;
            for (int j = 0; j < 3; j++) {
                int time = 0;
                for (int i = 0; i < 900; i++) {
                    HttpResponse<JsonNode> record = Unirest
                            .get("http://localhost:7878/meters/ise1224hi563" + j + "/data").asJson();
                    avgCurr += (Double) record.getBody().getObject().getJSONArray("measurements").getJSONObject(0)
                            .get("value");
                    avgVolt += (Double) record.getBody().getObject().getJSONArray("measurements").getJSONObject(1)
                            .get("value");
                    time = (int) record.getBody().getObject().get("unixTimestamp");
                }
                double resultCurr = avgCurr / 900;
                double resultVolt = avgVolt / 900;
                // average current of the smart meter which id ends with 2
                if (j == 2) {
                    System.out.println("average current (15 min interval): " + resultCurr);
                }
```

Group 46



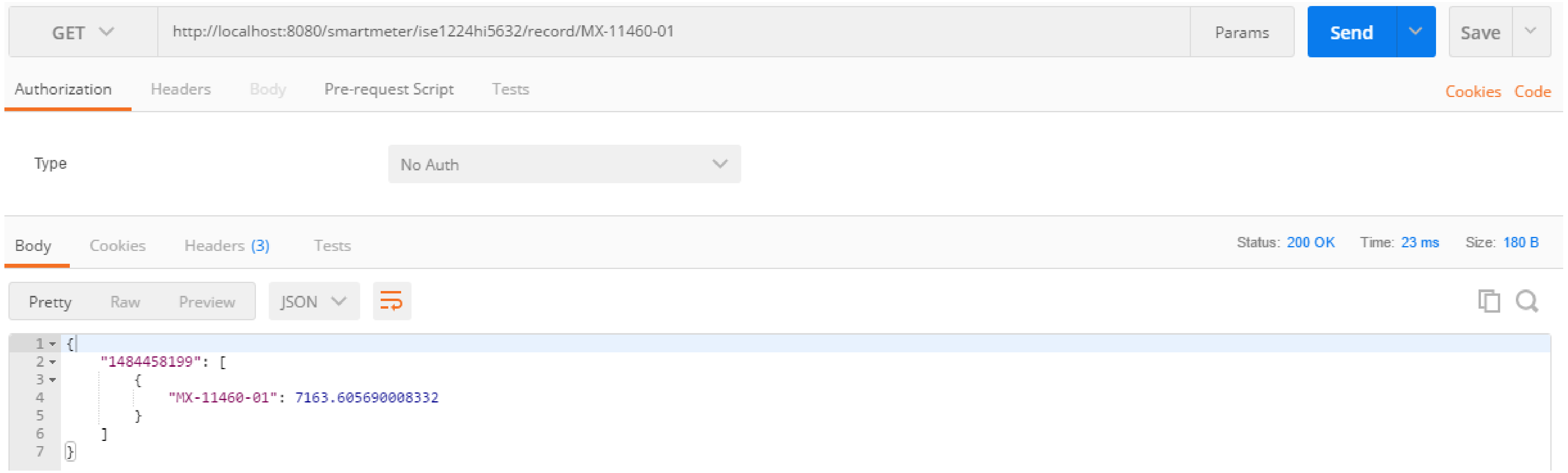
## Client Implementierung(3)



```java
                    // creates a JSON array
                    JSONArray array = new JSONArray();
                    JSONObject metric1 = new JSONObject();
                    metric1.put("metricId", "MX-11460-01");
                    metric1.put("value", resultCurr);
                    JSONObject metric2 = new JSONObject();
                    metric2.put("metricId", "MX-11463-01");
                    metric2.put("value", resultVolt);
                    JSONObject timeObject = new JSONObject();
                    timeObject.put("unixTimestamp", time);
                    array.put(timeObject);
                    array.put(metric1);
                    array.put(metric2);
                    Unirest.post("http://localhost:8080/smartmeter/ise1224hi563" + j + "/record").field("record", array)
                            .asJson();
                }
                k++;
            }
        }
```

Group 46

## Stromstärke Smartmeter mit ID=…2



http://localhost:8080/smartmeter/ise1224hi5632/record/MX-11460-01

Type: No Auth

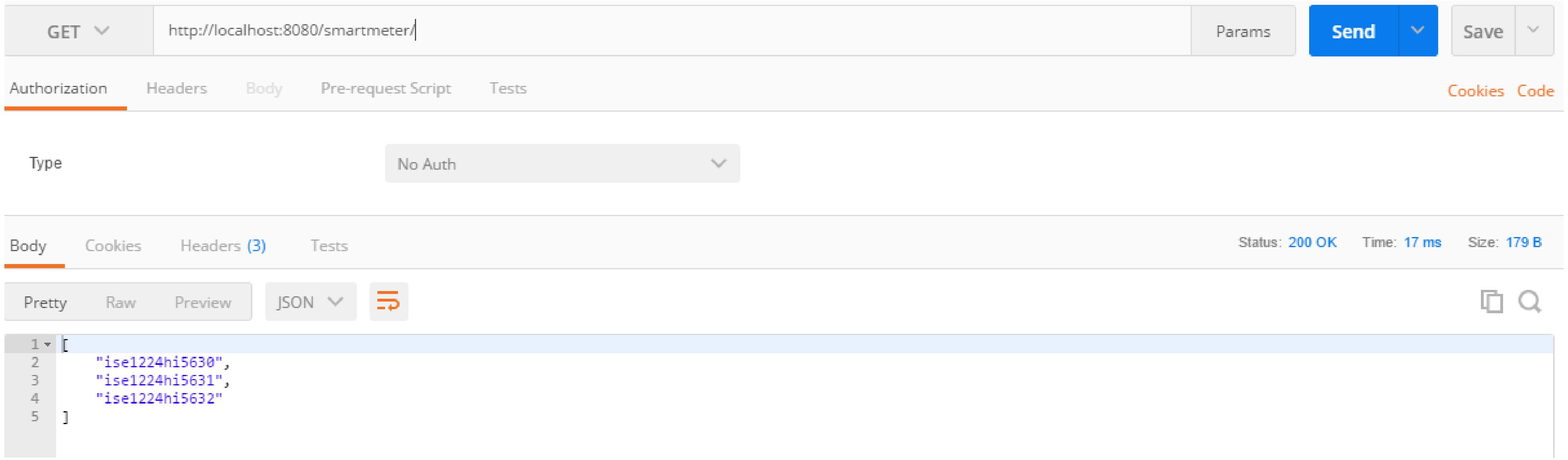
Status: 200 OK Time: 23 ms Size: 180 B

```
{
    "1484458199": [
        {
            "MX-11460-01": 7163.605690008332
        }
    ]
}
```

Group 46

Technische Universität Berlin

## /smartmeter



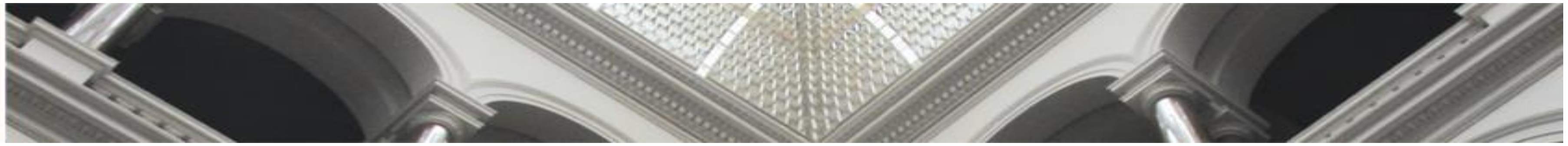
GET http://localhost:8080/smartmeter/

Type: No Auth

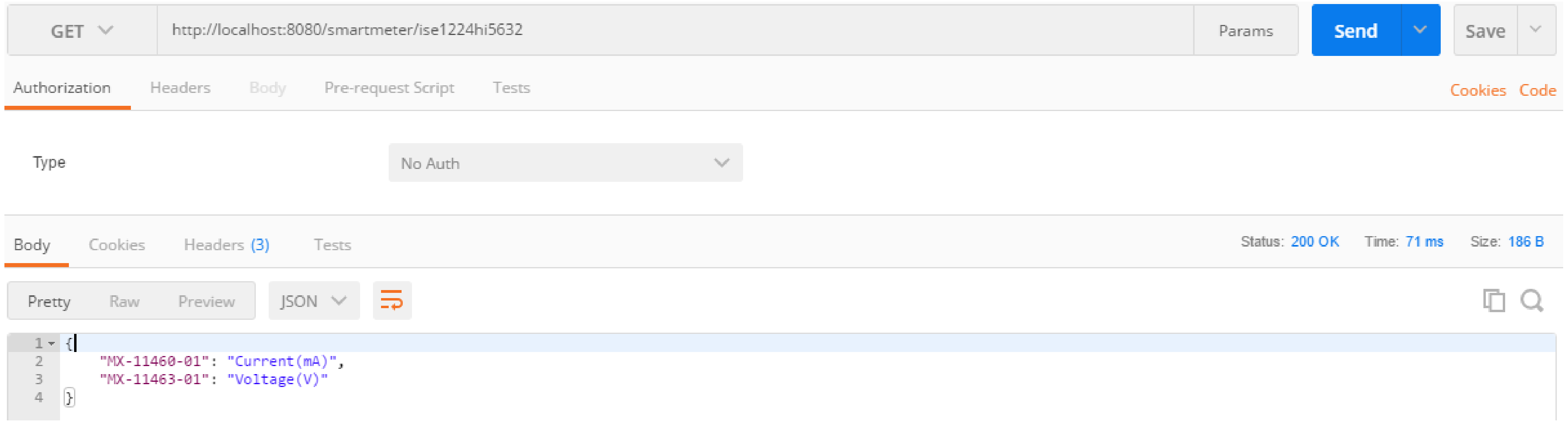
Status: 200 OK Time: 17 ms Size: 179 B

```
[
    "ise1224hi5630",
    "ise1224hi5631",
    "ise1224hi5632"
]
```

Group 46



## /smartmeter/{smartmeter}



http://localhost:8080/smartmeter/ise1224hi5632

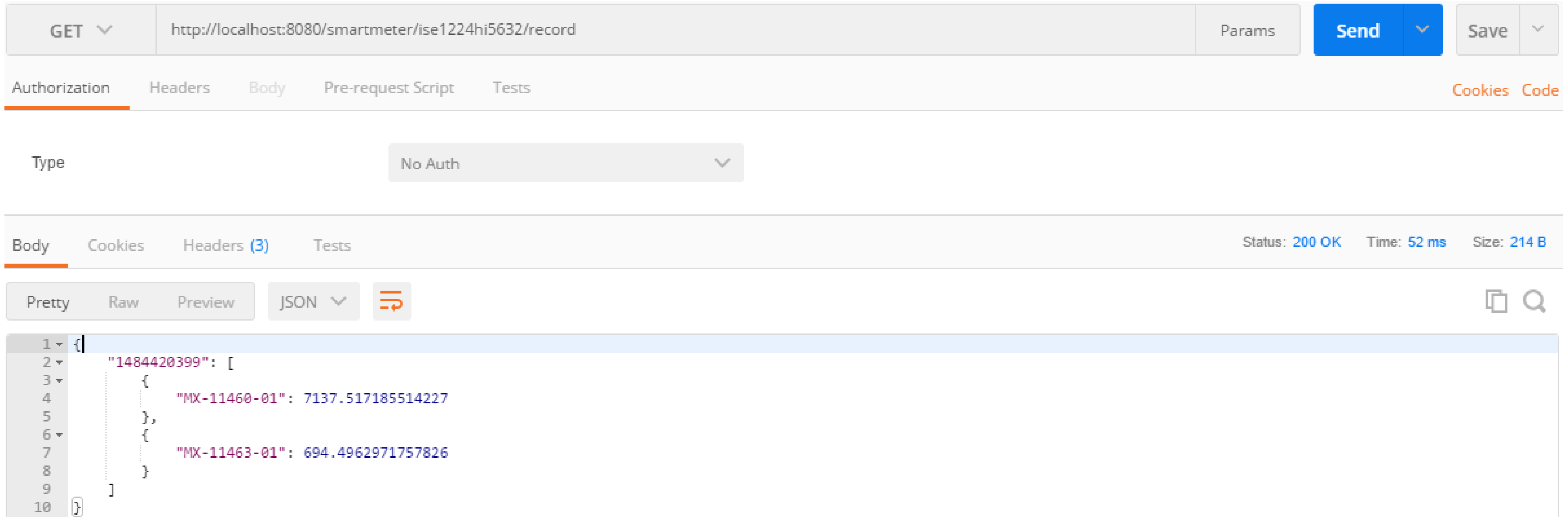
Type: No Auth

Status: 200 OK Time: 71 ms Size: 186 B

```
{
    "MX-11460-01": "Current(mA)",
    "MX-11463-01": "Voltage(V)"
}
```

Group 46

# /smartmeter/{smartmeter}/record



http://localhost:8080/smartmeter/ise1224hi5632/record

Type: No Auth

Status: 200 OK Time: 52 ms Size: 214 B

```
{
    "1484420399": [
        {
            "MX-11460-01": 7137.517185514227
        },
        {
            "MX-11463-01": 694.4962971757826
        }
    ]
}
```

Group 46