



Hochschule **RheinMain**

# **Design, Implementation and Evaluation of a System to Create a Data Set Supporting Research in the EnergieBroker Platform**

**Master Thesis**

for the

**Master of Science**

in Computer Science

at RheinMain University of Applied Sciences

by

**Niklas Sauer**

January 11<sup>th</sup>, 2022

**Project Duration**

6 Months

**Student ID**

1209772

**First Reviewer**

Prof. Dr. Heinz Werntges

**Second Reviewer**

Prof. Dr. Robert Kaiser

**Supervisor**

Johannes Kaepfel

## **Abstract**

# Contents

|  |            |
|--|------------|
| <b>Acronyms</b>  | <b>IV</b>  |
| <b>List of Figures</b>                                   | <b>V</b>   |
| <b>List of Tables</b>                                    | <b>VI</b>  |
| <b>Listings</b>  | <b>VII</b> |
| <b>1. Introduction</b>                                   | <b>1</b>   |
| 1.1. Motivation . . . . .                                | 1          |
| 1.2. Goals and Scope . . . . .                           | 1          |
| 1.3. Thesis Overview . . . . .                           | 1          |
| <b>2. Theoretical Framework</b>                          | <b>2</b>   |
| 2.1. Microservice Architecture . . . . .                 | 3          |
| 2.2. Virtualization . . . . .                            | 3          |
| 2.2.1. Hypervisor-based . . . . .                        | 3          |
| 2.2.2. Container-based . . . . .                         | 3          |
| 2.3. Container . . . . .                                 | 3          |
| 2.3.1. LXC . . . . .                                     | 3          |
| 2.3.2. Docker . . . . .                                  | 3          |
| 2.4. Container Orchestration . . . . .                   | 3          |
| 2.5. Kubernetes . . . . .                                | 3          |
| 2.5.1. Overview . . . . .                                | 3          |
| 2.5.2. Cluster Architecture . . . . .                    | 3          |
| 2.5.3. Containers . . . . .                              | 3          |
| 2.5.4. Workloads . . . . .                               | 3          |
| 2.5.5. Services, Load Balancing and Networking . . . . . | 3          |
| 2.5.6. Storage . . . . .                                 | 3          |
| 2.5.7. Configuration . . . . .                           | 3          |
| 2.5.8. Security . . . . .                                | 3          |
| 2.5.9. Policies . . . . .                                | 3          |
| 2.5.10. Scheduling, Preemption and Eviction . . . . .    | 3          |
| 2.6. Agile Software Development . . . . .                | 3          |
| 2.6.1. CI/CD . . . . .                                   | 3          |
| 2.6.2. DevOps . . . . .                                  | 3          |
| 2.6.3. GitOps . . . . .                                  | 3          |
| <b>3. Concept</b>  | <b>4</b>   |
| 3.1. Overview . . . . .                                  | 4          |
| 3.2. Functional Requirements . . . . .                   | 4          |
| 3.3. Non-Functional Requirements . . . . .               | 4          |

|   |           |
|---|-----------|
| <b>4. Architecture</b>                                    | <b>5</b>  |
| 4.1. Component Design . . . . .                           | 5         |
| 4.1.1. Hardware . . . . .                                 | 5         |
| 4.1.2. Backend . . . . .                                  | 5         |
| 4.1.3. Maintenance and Support Plan . . . . .             | 5         |
| 4.2. Component Interaction . . . . .                      | 5         |
| 4.3. Component Mapping . . . . .                          | 5         |
| 4.4. Design Rationale . . . . .                           | 5         |
| 4.4.1. Modeling as Cloud-Edge Computing Problem . . . . . | 5         |
| <b>5. Implementation</b>                                  | <b>6</b>  |
| <b>6. Testing</b>   | <b>7</b>  |
| <b>7. Conclusion</b>                                      | <b>8</b>  |
| <b>8. Summary</b>   | <b>9</b>  |
| <b>9. Outlook</b>   | <b>10</b> |
| <b>Appendices</b>   | <b>11</b> |
| <b>A. Concept</b>   | <b>12</b> |
| <b>B. Architecture</b>                                    | <b>13</b> |
| B.1. Representational State Transfer (REST) . . . . .     | 13        |
| B.2. Certificate-based Authentication . . . . .           | 13        |
| <b>C. Conclusion</b>                                      | <b>14</b> |

# Acronyms

# List of Figures

# List of Tables

# Listings



# **1. Introduction**

## **1.1. Motivation**

## **1.2. Goals and Scope**

## **1.3. Thesis Overview**



## **2. Theoretical Framework**

### **2.1. Microservice Architecture**

### **2.2. Virtualization**

#### **2.2.1. Hypervisor-based**

#### **2.2.2. Container-based**

### **2.3. Container**

#### **2.3.1. LXC**

#### **2.3.2. Docker**

### **2.4. Container Orchestration**

### **2.5. Kubernetes**

#### **2.5.1. Overview**

#### **2.5.2. Cluster Architecture**

#### **2.5.3. Containers**

#### **2.5.4. Workloads**

#### **2.5.5. Services, Load Balancing and Networking**

#### **2.5.6. Storage**

#### **2.5.7. Configuration**

#### **2.5.8. Security**

#### **2.5.9. Policies**

## **3. Concept**

### **3.1. Overview**

### **3.2. Functional Requirements**

### **3.3. Non-Functional Requirements**

## **4. Architecture**

### **4.1. Component Design**

#### **4.1.1. Hardware**

#### **4.1.2. Backend**

#### **4.1.3. Maintenance and Support Plan**

### **4.2. Component Interaction**

### **4.3. Component Mapping**

### **4.4. Design Rationale**

#### **4.4.1. Modeling as Cloud-Edge Computing Problem**

## 5. Implementation

## 6. Testing

## 7. Conclusion



## 8. Summary

## 9. Outlook

# **Appendices**

## **A. Concept**

## **B. Architecture**

### **B.1. Representational State Transfer (REST)**

### **B.2. Certificate-based Authentication**

## C. Conclusion