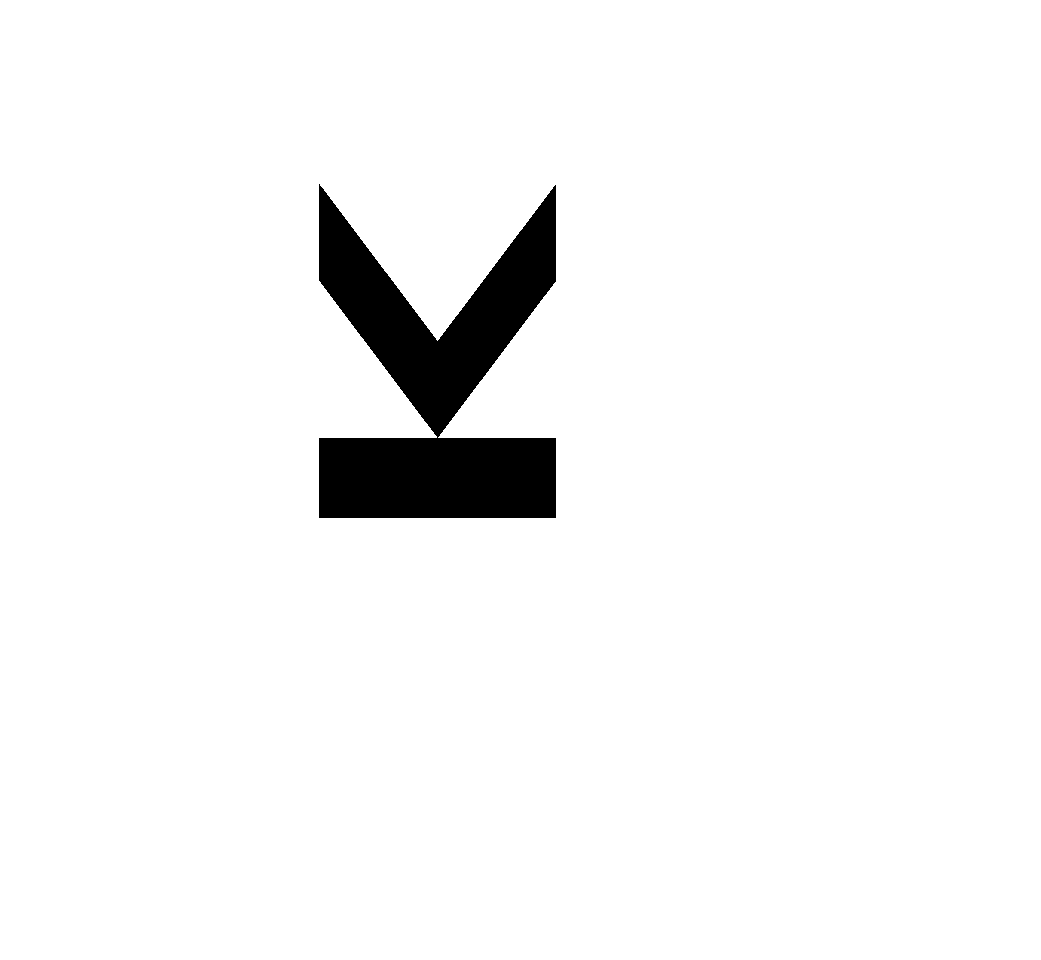
06/2022



Project  
documentation

**Table of Contents**

[1. Introduction 3](#_heading=h.gjdgxs)

[2. Implemented Requirements 3](#_heading=h.30j0zll)

[3. Overview of the System From the User´s Point of View 3](#_heading=h.1fob9te)

[4. Overview of the System From the Developer´s Point of View 3](#_heading=h.3znysh7)

[4.1. Design 3](#_heading=h.2et92p0)

[4.1.1. Overview of the System 3](#_heading=h.tyjcwt)

[4.1.2. Important Design Decisions 3](#_heading=h.3dy6vkm)

[4.2. Implementation 3](#_heading=h.1t3h5sf)

[4.3. Code Quality 3](#_heading=h.4d34og8)

[4.4. Testing 3](#_heading=h.2s8eyo1)

[5. Installation guide 4](#_heading=h.3rdcrjn)

**Version history**

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Date | Creator | Changes |
| 1.1 | 15.06.2022 | Dobretsberger | added Req-Matrix |
| 1.2 | 16.02.2022 | Dobretsberger | Overview User-Int. |
| 1.3 | 17.02.2022 | Dobretsberger | added Reports |
|  |  |  |  |

Table 1: Version history

# Introduction

*Overview of the project and this document*

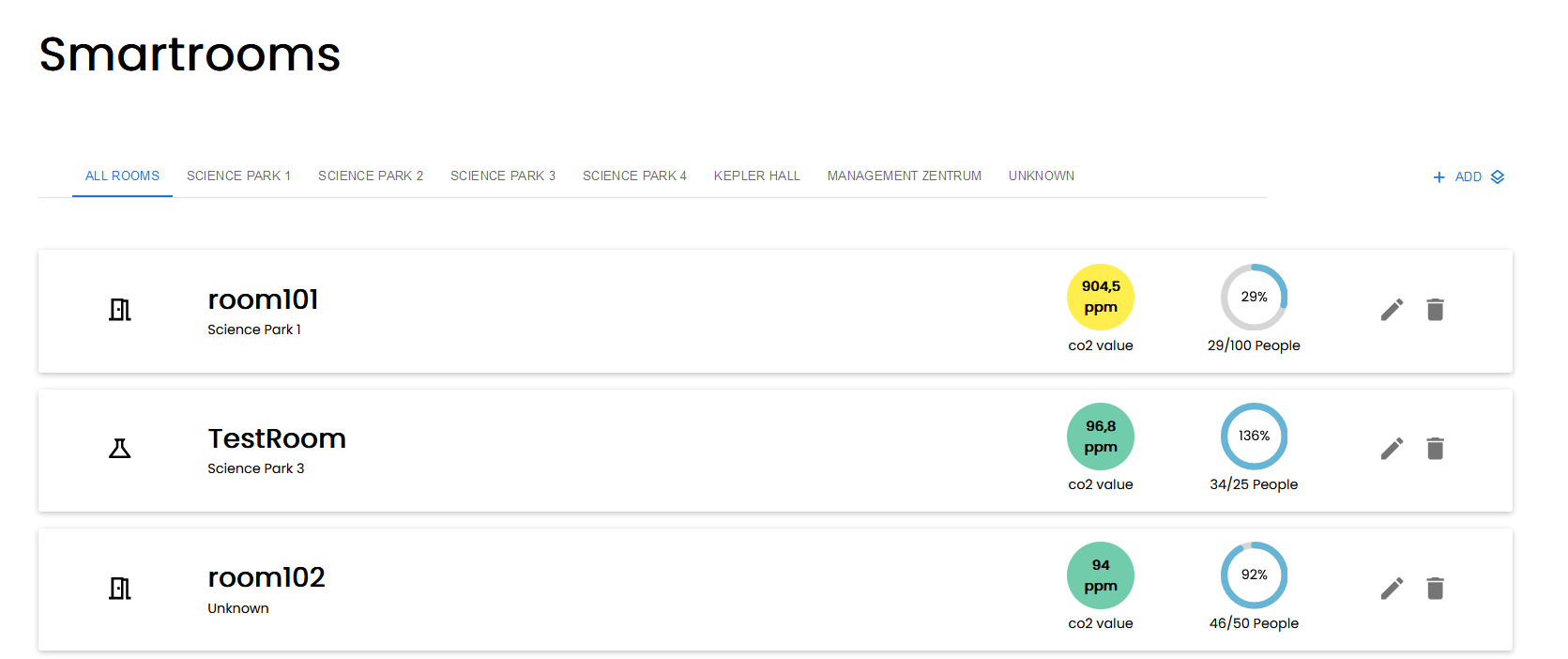
# Implemented Requirements

*Which of the requirements did you implement? Which teammate was responsible? and How many hours he/she expend implementing this? If any requirement has not been implemented, then, the team should provide explanations for each requirement*

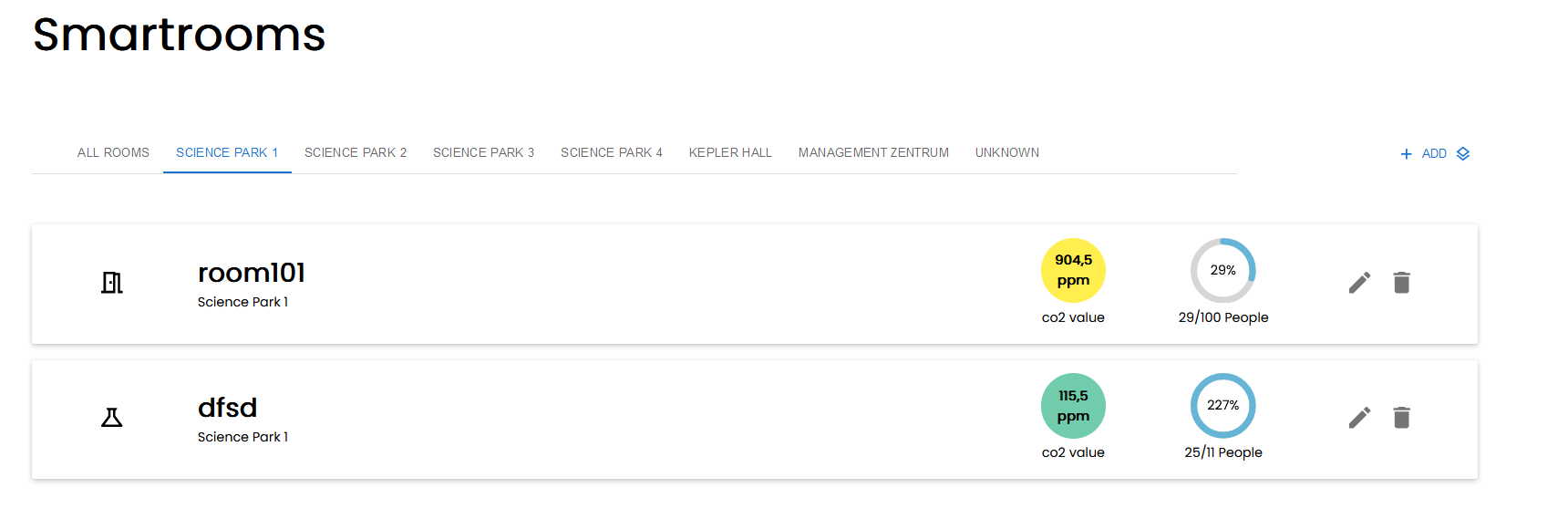
|  |  |  |  |
| --- | --- | --- | --- |
| Nr. | Requirement | Responsible | Effort in SP |
| 1 | Basic: Import data from CSV to an Entity-Relationship database. | Markus Mühleder  Florian Dobretsberger | 21 |
| 2 | Basic: Create/Update Rooms (id, size, available doors, windows, lights and fans). | ALL |  |
| 3 | Basic: Update and remove rooms. |  |  |
| 4 | Basic: Visualize available rooms. | Petra Körper  Matthias Herzog |  |
| 5 | Basic: Visualize static information for each room (id, size, available doors, windows, lights and fans). | Petra Körper  Matthias Herzog |  |
| 6 | Basic: Develop a line chart that shows real-time data regarding light/fan/window/door. | Petra Körper  Matthias Herzog |  |
| 7 | Basic: Develop a line chart that shows the co2/temperature values and the number of people for each room over time. | Petra Körper  Matthias Herzog |  |
| 8 | Basic: Save rooms structure (rooms + static information) in a .csv file. | Florian Dobretsberger |  |
| 9 | Basic: Automatically add random values of co2/temperature/number of people for a specific room. | Markus Mühleder |  |
| 10 | Basic: Live update of visualizations for co2, temperature, and lights/ventilators/windows/doors status for each room. | Petra Körper  Matthias Herzog |  |
| 12 | Remote Control: Allow to lock/unlock doors via the user interface. |  |  |
| 13 | Remote Control: Allow to turn on/off lights via the user interface |  |  |
| 14 | Remote Control: Allow to open/close windows via the user interface. |  |  |
| 15 | Remote Control: Allow to turn on/off fans via the user interface. |  |  |
| 16 | Security: Send alarm if temperature is above 70 degrees celsius. |  |  |
| 17 | Security: Unlock all doors if temperature is above 70 degrees celsius. | Markus Mühleder  Florian Dobretsberger |  |
| 18 | Energy Saving: Turn lights on if there are people in the room. |  |  |
| 19 | Energy Saving: Lights should be turned off if the room is empty. |  |  |
| 20 | Energy Saving: Turn off running devices if the room is empty. | Markus Mühleder  Florian Dobretsberger |  |
| 21 | Air Quality: Open window + activate fan if co2 values are > 1000 parts per million (ppm). | Markus Mühleder  Florian Dobretsberger |  |
| 22 | Air Quality: Change room color in user interface based on co2 values.  - green if c2o values are < 800 ppm  - yellow if co2 values are between 800 and 1000 ppm  - red if co2 values are above 1000 ppm | Petra Körper  Matthias Herzog |  |

# Overview of the System From the User´s Point of View

*How were the requirements implemented in the user interface? (Screenshots of the user interface and description of the functionality based on scenarios) How the users should use the system?*

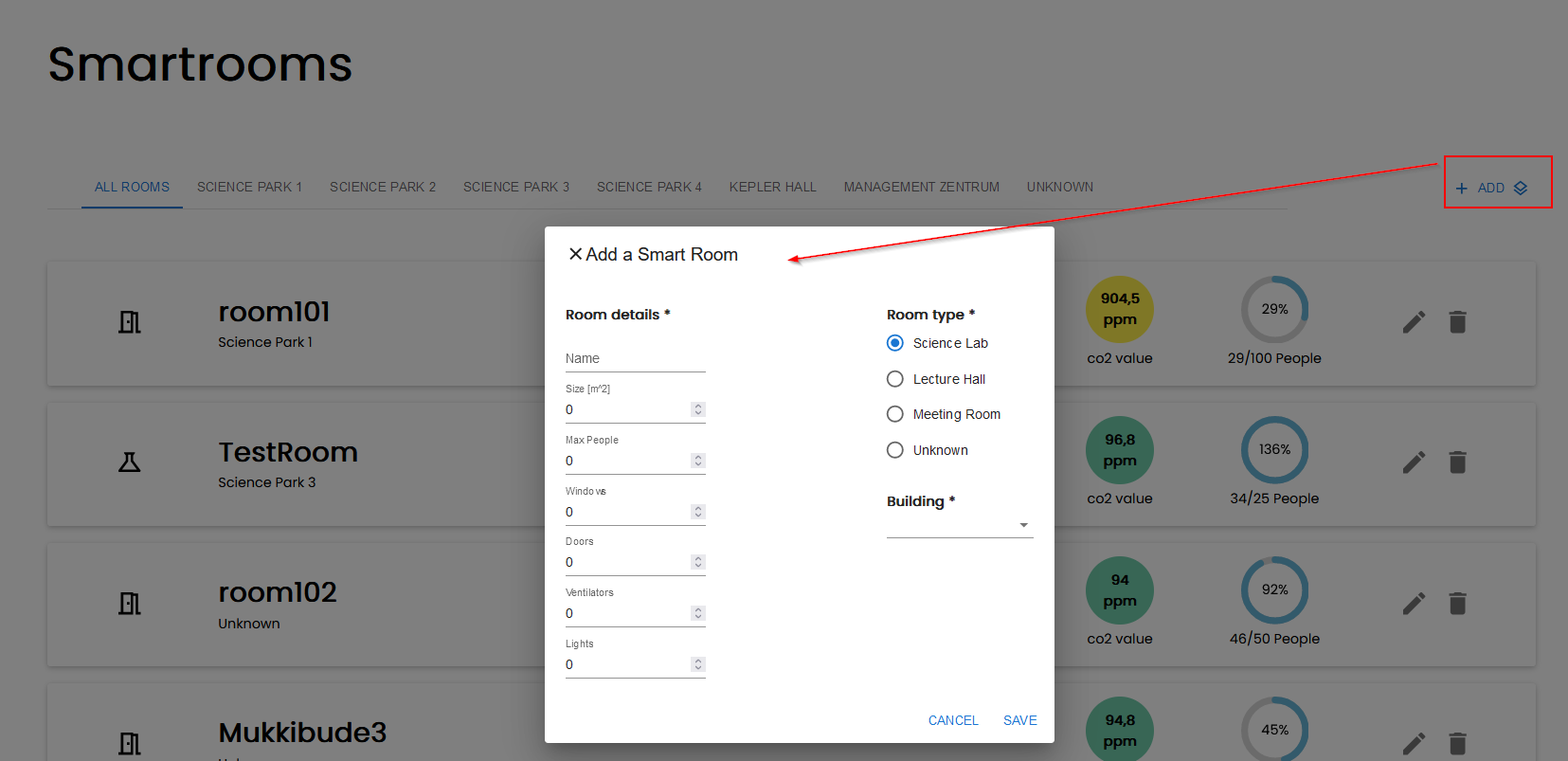
The Mainpage initially displays all rooms which are available within the system. The User is able to switch between the buildings via the menu. Furthermore, the current co2 value and the capacity-state regarding the appropriate room are being displayed automatically.

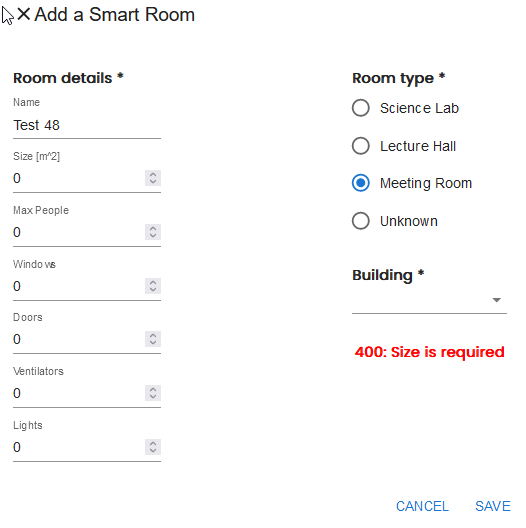
Rooms filtered by building:



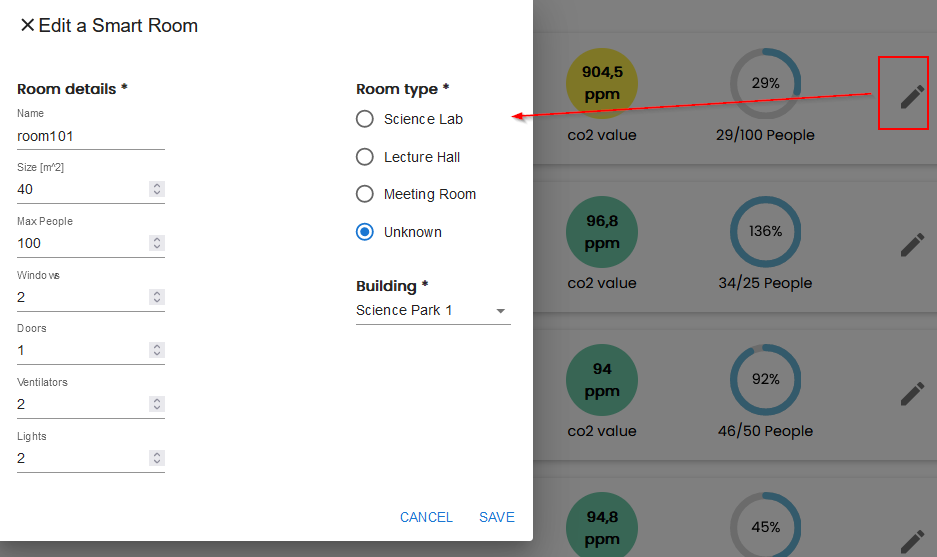
(CUT)

The User is able to create a new room via the ADD Button. A dialog appears and the user needs to fill in all necessary details regarding the room.





In case of wrong input the user gets a notification in terms of the related error.



The user is able to edit an existing room via the edit-button.



**(REPLACE)**

By clicking on a room-section on the overview the user gets a detailed page of the predefined properties. Moreover, the user can control the state of windows, doors, fans and light by clicking on the appropriate button.

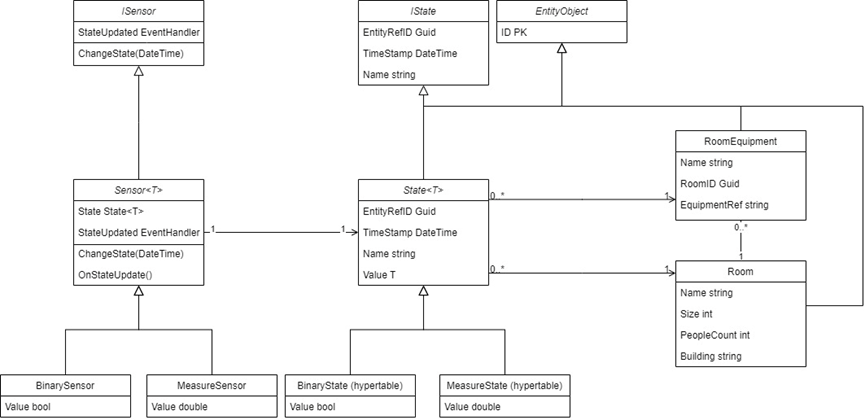
# Overview of the System From the Developer´s Point of View

## Design

### Overview of the System

*UML Diagram with explanations*

*Design patterns used (e.g. Model-View-Controller)*



### Important Design Decisions

*Description of the 3-5 most important design decisions in the following scheme*

*Decision:*

*Reason:*

*Considered Alternatives:*

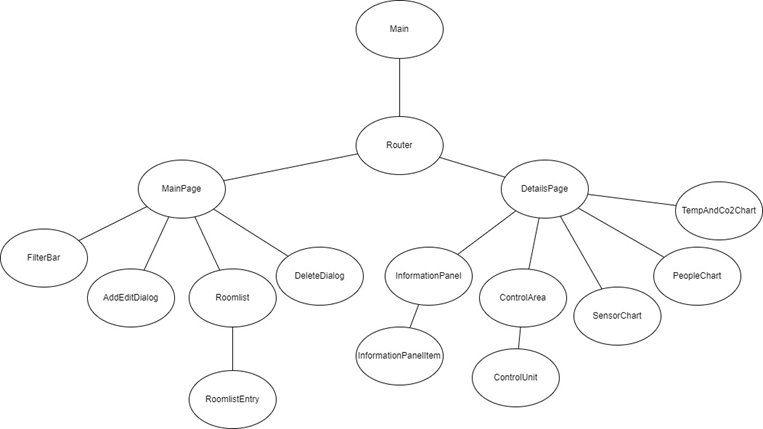
*Assumptions:*

*Effect:*

## Implementation

*Description of the important aspects of the implementation (it can be described a few selected pieces of code), project structure, dependencies, libraries used, etc.*

Frontend Component-Tree:



*ReAct*

*Flexibilität*

*Begründung für Frameworks*

*# Component-Tree (proj-Structure)*

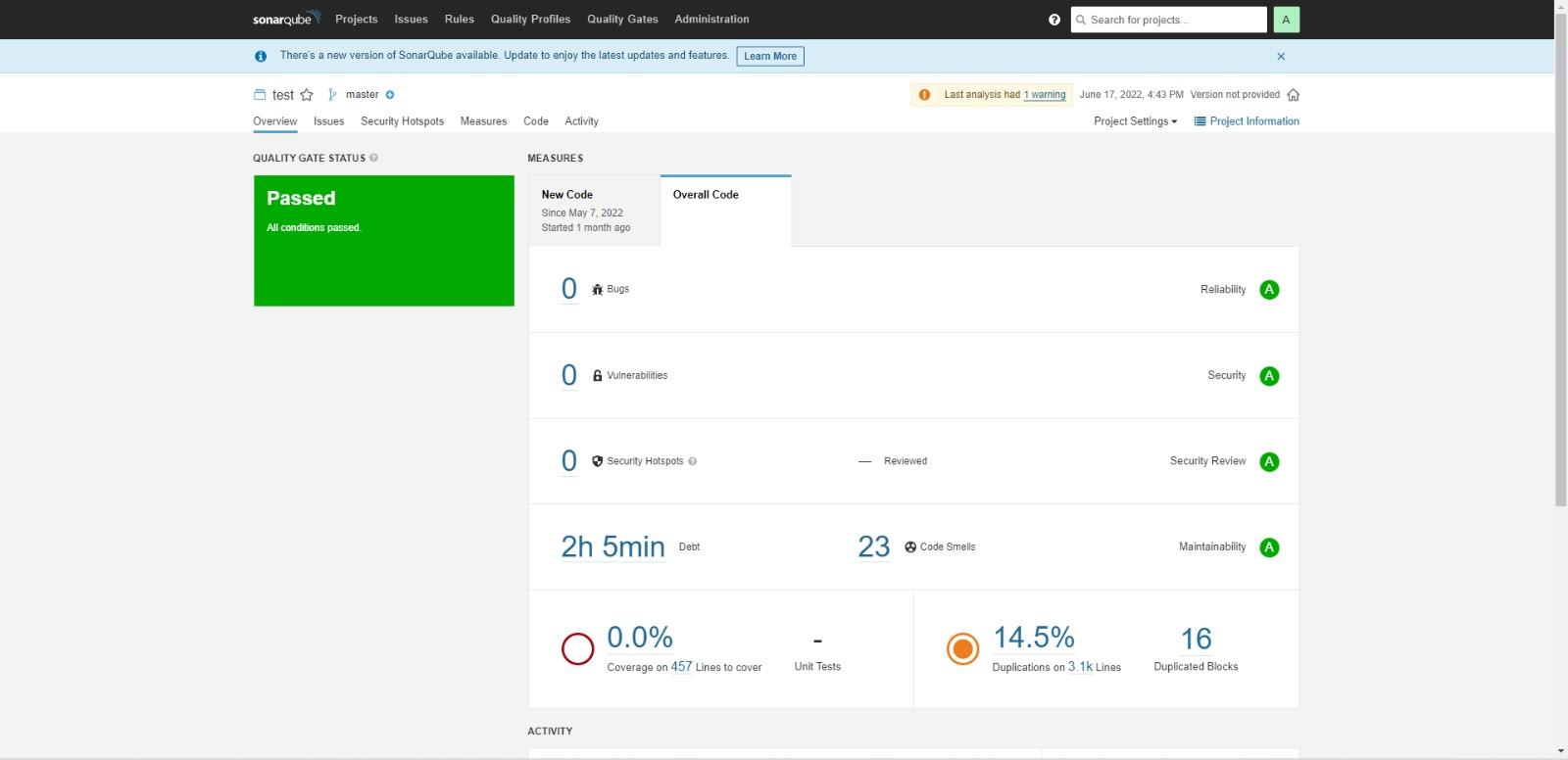
*# Service Arch*

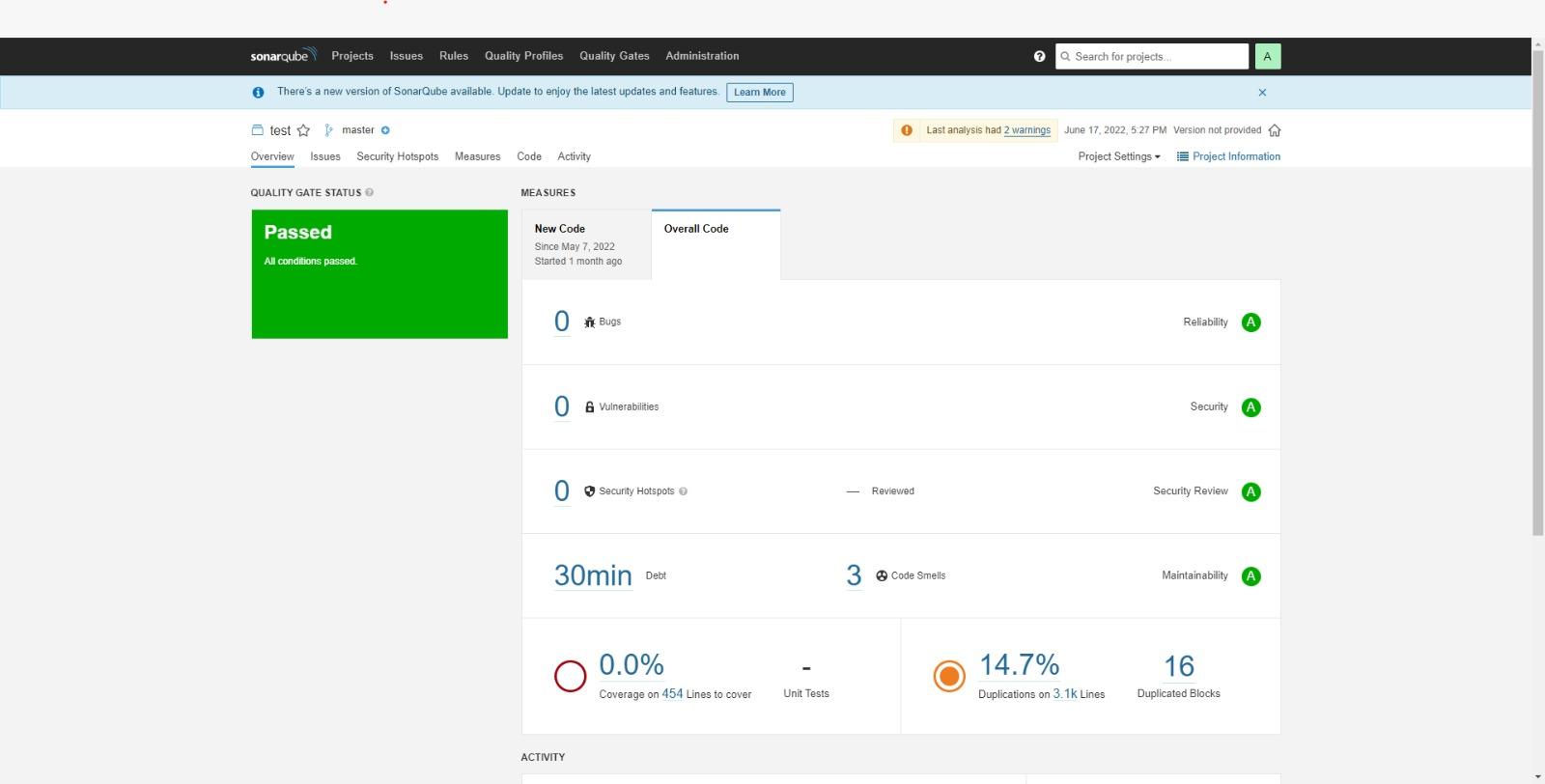
## Code Quality

*Description of the use of PMD, description of the findings and which of them have been fixed.*

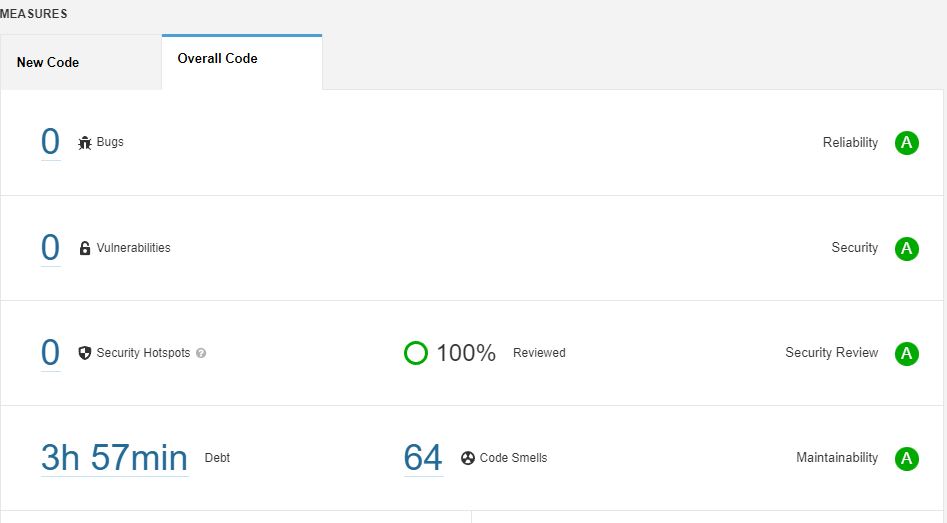
*Pictures (Reports)*

Code Quality Frontend



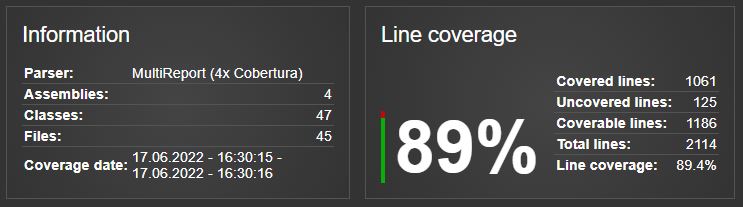


Code Quality Backend



## Testing

*Overview of created JUnit tests (it can be described a few selected tests)*



*Description of the 3 selected tests according to the following pattern:*

|  |  |
| --- | --- |
| Test Case ID |  |
| Designed by |  |
| Execute on |  |
| Carried out by |  |
| Tested Requirement |  |
| Requirement |  |
| Test steps |  |
| Test data |  |
| Expected result |  |
| Post condition |  |
| Status |  |
| Comments |  |

|  |  |
| --- | --- |
| Test Case ID |  |
| Designed by |  |
| Execute on |  |
| Carried out by |  |
| Tested Requirement |  |
| Requirement |  |
| Test steps |  |
| Test data |  |
| Expected result |  |
| Post condition |  |
| Status |  |
| Comments |  |

|  |  |
| --- | --- |
| Test Case ID |  |
| Designed by |  |
| Execute on |  |
| Carried out by |  |
| Tested Requirement |  |
| Requirement |  |
| Test steps |  |
| Test data |  |
| Expected result |  |
| Post condition |  |
| Status |  |
| Comments |  |

# Installation guide

*Description of how to install and start the system*

*(Link to github)*

*Link for Frontend*

*Link for Backend*