

**Masaccio – Monitoring for urbAn SAfety with the IoT**Deliverable D2 Template

|  |  |
| --- | --- |
| **Date** | 29/11/2017 |
| **Team ID** | SF |

|  |  |  |
| --- | --- | --- |
| **Team Members** | | |
| **Name and Surname** | **Matriculation number** | **E-mail address** |
| Fabio Giancristofaro | *255146* | [*Fabio.giancristofaro@student.univaq.it*](mailto:Fabio.giancristofaro@student.univaq.it) |
| Stefan Ilic | *252212* | *Stefan.ilic@student.univaq.it* |

Table of Contents

[Challenges/Risk Analysis 3](#_Toc497175124)

[Major differences do D2 and D3 4](#_Toc497175125)

[List of Assumptions](#_Toc497175125) 5

[State of the art](#_Toc497175126) 6

[Informal Description of your system and its Software/System Architecture](#_Toc497175127) 8

[User Stories](#_Toc497175128) 12

[Functional requirements 1](#_Toc497175129)2

[Extra-Functional requirements 1](#_Toc497175130)3

[Views and Viewpoints 1](#_Toc497175131)4

[UML Static and Dynamic Architecture View 1](#_Toc497175129)5

[Design Decisions 2](#_Toc497175132)6

[From Architecture to Code](file:///C:\Users\bargn\Downloads\l'aquila\sw%20architecture\project\DeliverableTemplate-SA2017-2018_D2.rtf#_Toc497175401) 31

[CAPS Architecture view](#_Toc497175128) 38

[CAPS SAML 38](#_Toc497175129)

[CAPS HWML 43](#_Toc497175129)

CAPS SPML44

[Summary](file:///C:\Users\bargn\Downloads\l'aquila\sw%20architecture\project\DeliverableTemplate-SA2017-2018_D2.rtf#_Toc497175402) 47

# Challenges/Risk Analysis

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Date the risk is identified** | **Date the risk is resolved** | **Explanation on how the risk has been managed** |
| Ensure fault tolerance | 28.10.2017 | 28.10.2017 | System will be distributed with each school having local server and being able to function autonomously in a degraded mode. This will ensure that even in case communication breaks they will be able to continue to function. |
| Ensure 5 sec delivery time for critical messages | 28.10.2017 | 07.11.2017 | Critical messages are sent with  push notifications and SMS service. Messages are sent on these two channels, ensuring a high chance of message delivery. |
| Ensure system can handle up to 40.000 messages per hour | 28.10.2017 | 28.10.2017 | Since schools have local servers, sensor load will be distributed across them, ensuring they don’t get overwhelmed. |
| Each critical message has to be delivered | 07.11.2017 | 07.11.2017 | Use message queue with ACK from server side to ensure no messages are lost. |
| Ensure privacy of student data | 12.12.2017 | 12.12.2017 | * Parents have to be notified about Italian privacy law. * School servers will not be exposed to open internet thus reducing chance of data breaches. * Ensure that we have proper permission for storing video recording for more than 24/h. * Access to video recording is just for authorized operators. * Parents can only receive notifications about their children and check their arrival/departure time. |

# 

# Application Domain

# Major differences with respect do D2 and D3

# The biggest difference between D2 and D3/D4 is on the dashboard. A new dashboard has been implemented and is available for parents. A parent will see a login page at first, where he has to enter his codice fiscale and password. If data are found in database, he will see all the entrance related to his children, since each child’s account are connected to parents’ account.

The old dashboard, implemented in D2, has been completely changed. Now all the content is available only after a login done by operators with their username and password. A login session will be created and will allow to check:

* Children log entrance;
* Current rooms capacities;
* Add entry/exit manually;
* Add a new guest;
* Add/Modify/Delete user data;
* Add/Modify/Delete permission’s room.

Now passwords are protected using PBKDF2. This method uses multiple iterations of password hashing with salt value to reduce vulnerability of brute force attacks on stored passwords. In database passwords are stored in following format:

***pbkdf2:sha256:50000$salt$hash***

Number of iteration is set to 50000 and it is constant.

Another change from D2 is also that access right service now considers room occupancy when granting access permission and also tracks room occupancy status.

Sequence diagrams:

3. Check child arrival and departure time

4. Gain access to authorized areas

have been slightly updated to reflect control flow in code.

# List of Assumptions

|  |  |
| --- | --- |
| **Assumption** | **Description** |
| We assume that WiFi will be already present in school |  |
| We assume that power supply for sensors,cameras and actuators is already taken care off |  |
| Sensors are appropriately located | Since we instal sensors we assume they are going to be properly positioned |
| Emergency exit will be present | Emergency exits don’t have RFID readers. |
| Each student has a card | A student has to insert the card at the entrace. In case that card is forgotten or lost student can be recognized by name and surname (using photo as reference). |
| Students will use their card at the entry and at the exit | Operators at entry point will check if students are using properly their cards. |
| We have legal permission for storing camera data for more then a day | Since Italian law gives 24h limit on storing camera data without certain legal permissions, we assume that school are authorized to store data for 3 days. |
| We have parents legal permission to store child data | In Italy to store children data schools need to obtain parents permission. |
| We assume that school will hire personel to operate system | Staff is required at the entries, to ensure proper entry/exit procedures are followed and to assist children in case of need. Also operator is needed to manage access rights and monitor camera feeds. |
| We assume each school has designated release point | Release point is a point in which children have to gather in place of emergency and it will be used as point where parents pick up their children. |
| Rooms for special events have a fixed capacity | People counters ensure that this capacity will not be exceeded. |
| Only people authorized by parents can interact with children | Authorized people (e.g. uncles, brothers etc.) that can interact with a specific child and with its data are communicated to school operator and registrated. |

# 

# State of the art

***From Google search:***

* Campus Safety: Five Advances in Physical Security for Schools

*https://www.cisco.com/c/en/us/solutions/industries/education/secure-campus.html#~stickynav=3*

<https://www.cisco.com/c/dam/en_us/solutions/industries/docs/education/safetyBroch013108.pdf>

This document explain how is important to protect students and staff from threats including armed intruders, abductions, fights, natural disasters, bullying, vandalism, and thefts.

Safe and secure campuses are better learning environments. They help to attract and retain teachers and to prevent theft and vandalism frees up money for next generation learning programs.

Input: from this document we got a general view about security inside schools and some possible services to implement.

* *Safe schools campus management*

<https://www.safeschoolssecurity.com/>

SafeSchools Campus Management is a modular and scalable student safety and campus management system. It provides services like student monitoring, event management, emergency management and visitor security.

Input: from this document we got a lot of useful information about students monitoring and visitor management system, information that we are going to use to implement Access right management service.

* LA VIDEOSORVEGLIANZA NELLE SCUOLE: TRATTAMENTO DEI DATI PERSONALI E RISPETTO DELLA PRIVACY

<http://www.dirittoscolastico.it/wordpress/wp-content/uploads/La-videosorveglianza-nelle-scuole.pdf>

This document contains a lot of information about cameras inside school.

Input: From here we got information about Italian laws that authorize use of cameras inside schools: rules that operators must respect, days of video storage, privacy, the aim and what cameras can avoid.

* Ways to improve school entrance security

<https://www.campussafetymagazine.com/hospital/8_ways_to_improve_school_entrance_security/>

## Schools must recognize the importance of good maintenance, good construction, good design, and a fair and equal management style of school operation. Many campus leaders are looking to enhance how people, goods and services enter their facilities. A secure entrance can prevent unauthorized entry and delay their access until law enforcement can be summoned.

Input: Here we found information about entrance security that will be integrated in our Access right management service.

* School safety and security toolkit

<http://archive.ncpc.org/resources/files/pdf/school-safety/bsstoolkit_complete.pdf>

This toolkit is an easy-to-use guide that will assist parents and administrators in implementing the Be

Safe and Sound model in their schools. It includes a step-by-step procedure for assessing school safety

and security, forming an action team, identifying the problems, holding a forum with stakeholders to

brainstorm solutions, developing an action plan and building support for it and evaluating the

results.

Input: This document helps us to manage the project step by step understanding the kinds of safety and security threats today’s students face at school and what makes a school safe.

* The appropriate and effective use of security technologies in U.S. schools

<https://www.ncjrs.gov/school/178265.pdf>

This guide describes existing commercially available technologies and urges thoughtful consideration of not only the potential safety benefits that may accrue from their use. In particular there is a discussion for each potential service pointing out its negative aspects.

Input: Many information about camera, sensors and metal detector. From here we took data for alarm management.

# 

# Informal Description of your system and its Software/System Architecture

Our system is focused on increasing safety of students in schools, it consists of multiple services each focused on this goal. Primarly system is focused on checking authentication of persons entering school, this is done trough RFID cards. Card will be given to students and to all other school employees. Operators will be placed on school entrances who will enforce that proper procedures will be followed, also in case of forgotten RFID cards they will be able to manualy check in students to school. Parents are allowed to come inside school as guest with a temporary card.

As a optional measure to increase security metal detectors can be placed on entry points. The same cards will be used to enforce permissions of entering specific rooms for example labaratories or teacher offices.

This authorization system will be tied will people counters enabling more control, for example forbidding entry in rooms over set capacty.

To help parents, system will also enable them to see when their children enter and leave school premisses. This option can also be extended to school busses by adding appropriate RFID readers.

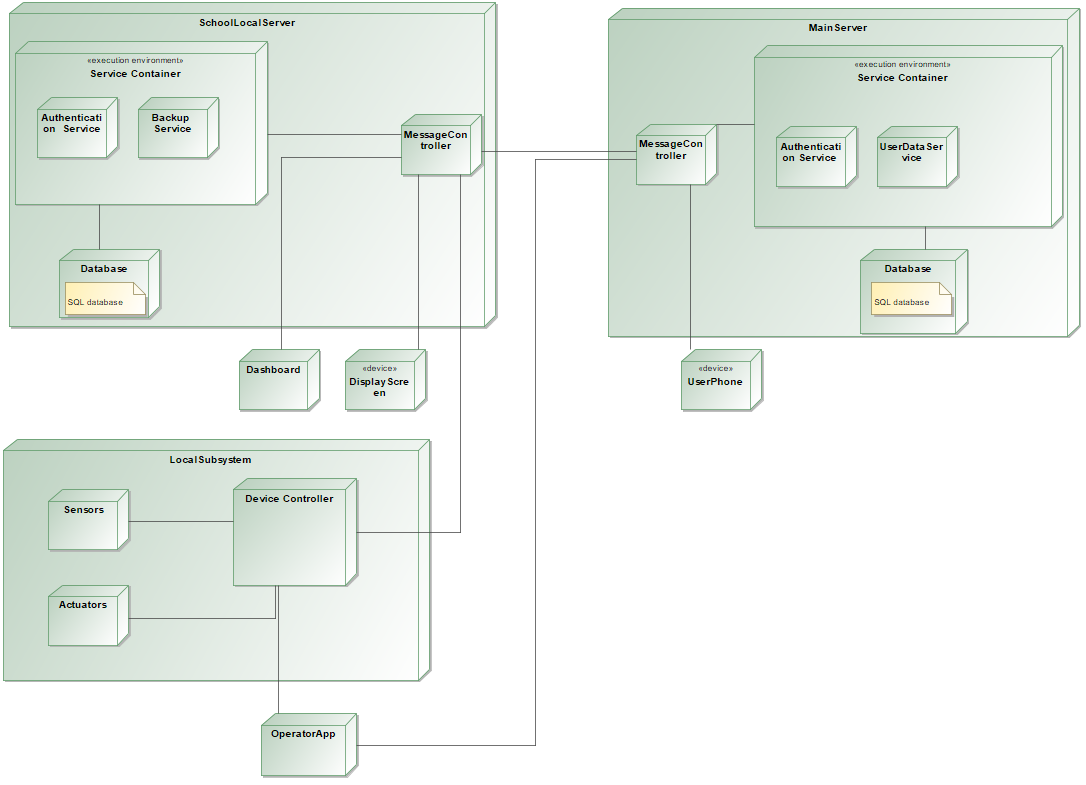
An additional service of the system will consist of number of cameras covering key areas of interest. Cameras won’t be activly monitored due to cost, but will be used to get more information about incidents (bullying, property damage, fights etc).

Additionaly system will consist of number of fire, earthquake sensors which will be able to automatically raise alarms. This tied to option of automatically calling appropriate emergency services will ensure that each situation get handled as quickly as possible. In case situation is not detected by sensors authorized staff will be also able to raise alarms if needed.

|  |  |
| --- | --- |
| **Service** | **Features** |
| Access right management | * only authorized people can enter school * unauthorized people can enter with a guest card after security check (manual) * students can only enter student areas (classrooms and common areas) * record entrance and departure time |
| Parents information service | * parents can check time of student arrival and departure from school * parents receive notifications in case of emergency or unexpected events |
| People counters | * information about occupancy of specific rooms * prevent entry of authorized people in case capacity is exceeded |
| CCTV video surveillance | * automatically record videos and store them for 3 days * enable reviewing of stored content |
| Alarm system | * fire detection alarm * earthquake detection alarm * manual alarm from operator app * operators can stop alarms * emergency exit management |
| First responders communication service | * automatic calls to appropriate emergency services in case of alarm |

Optional security features can include:

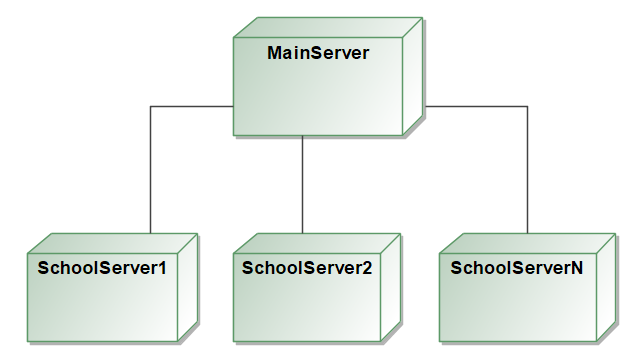
* metal detector (at the entry point)
* RFID card reader can be put also on the bus to give more information to parents



System will have 3 main parts.

The schoolLocalServer module that represents schools, the mainServer which is the central server connected to every node (schoolLocalServer) and the localSubSystem, a subsystem of a single school. Local school servers will not be connected to open internet to prevent potential security problems.

The architecture is combination between a client-server architecture and distributed system. Clients in our case are school nodes which are mostly independent and self-sufficient. Since each node doesn’t share anything with others, this kind of architecture will eliminate any [single point of failure](https://en.wikipedia.org/wiki/Single_point_of_failure). Nodes depend on main server only for communication with parents and periodic backups while for all security services they can function autonomously.



The schoolLocalServer is composed by a service container and will be able to run local services autonomously using data stored in local SQL database. This database contains only data related to that school and its students. Spliting functionality in services will enable us to deploy system in multiple steps. Information saved in the local DB will be sent periodically to the main server that will be used as backup in case of problems or data loss. Camera data will be stored on file system. This data will not be copied to main server due to high network usage instead RAID will be used to ensure data safety.

The localSubsystem module include sensors, actuators and device controller which will be responsible for message passing.

Operators will interatct with system trough dashboard and operator app.

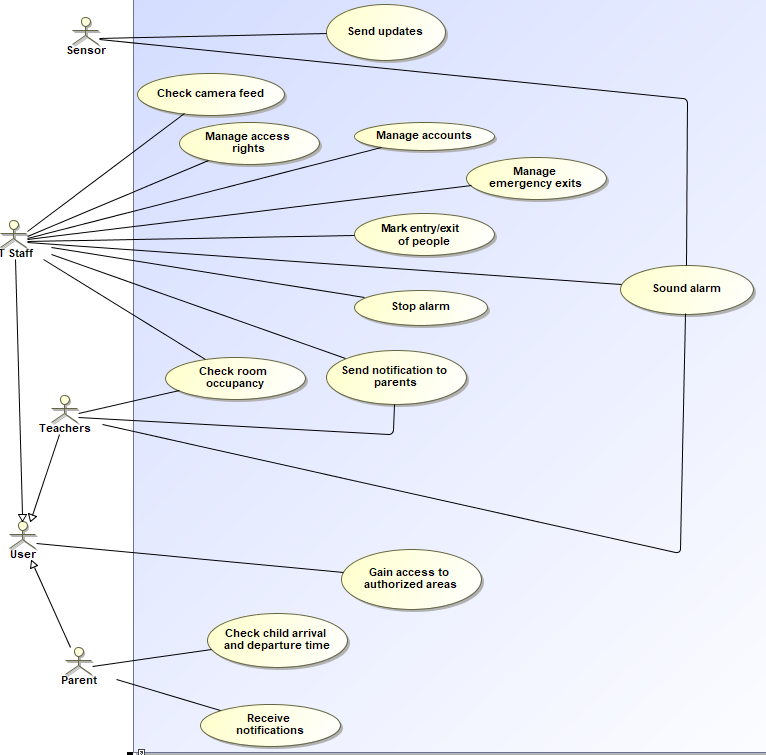
Operator app will provide options to raise alarms, stop alarms, give basic informations about room occupancy, provide ability to check in users with missing card and notify parents. Operator app will communicate directly with device controller of the localSubsystem which will enable communication to the local server, or in case local system is down directly with a main server using device internet (only alarm operations are available in this mode since they are of highest priority).

Dashboard will provide more detailed overview related to the building where they are working. It will contain all functionality like operator app while adding options to monitor camera feed, manage emergency exits and accounts. These operation would be unpractical for operator app given limitation of screen size.

Main server is used by all schools for communications with parents and as a backup. To avoid main server becoming single point of failure we will deploy backup main server which will take over in case of failure. This will help us achieve higher availability and increase fault tolerance of system.

# 

# User Stories



## Functional requirements

Parent,

* check arrival and departure time of their children
* receive notifications about security issues
* gain access to areas they are authorized

Users (students),

* gain access to areas they are authorized

Teachers,

* gain access to areas they are authorized
* gain information on room occupancy
* raise alarms

IT Staff,

* gain access to areas they are authorized
* gain information on room occupancy
* manage alarms (raise new alarms and stop false ones)
* manage access permissions
* check camera feed
* mark entry/exit of people in case of missing RFID card
* create/edit user accounts
* manage emergency exits

Sensors,

* raise alarms
* send periodic updates

## Extra-Functional requirements

|  |  |  |
| --- | --- | --- |
| *Requirement nr* | *Requirement* | *Priority* |
| *1* | *Critical message should be delivered in 5 sec* | *High* |
| *2* | *System should be able to handle 40000 messages/hours* | *Medium* |
| *3* | *System shall be operative in a degraded mode in case of disaster* | *High* |
| *4* | *System shall be able to handle up to 2000 sensors* | *Medium* |
| *5* | *System will be able to function in degraded mode in case of power loss for 4h* | *High* |
| *6* | *Emergency exits must unlock automatically in case of alarm* | *High* |
| *7* | *Video recording has to be saved for exactly 3 days* | *Medium* |
| *8* | *Ensure privacy of student data* | *Medium* |
| *9* | *Ensure backup of student data* | *Medium* |
| *10* | *Information about child entrance/exit are automatically updated once card is used* | *Medium* |
| *11* | *Special notifications to parents are sent automatically in case of emergency* | *High* |
| *12* | *Forbid access to unauthorized people at the entry* | *High* |
| *13* | *Forbid access to children in particular rooms* | *Medium* |
| *14* | *Forbid access in special rooms once the maximum capacity is reached* | *Medium* |

# Views and Viewpoints

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
|  | Service Provider (NoSQL) | User (parents) | Users (students) | School staff | Mobile App Developer | Emergency services | Sensor app developer |
| Dependability | X | X |  | X | X | X | X |
| Energy Consumption |  |  |  |  |  |  | X |
| Networking & Communication |  |  |  |  | X |  | X |
| Usability | X | X | X | X | X | X |  |
| Performance | X |  |  | X | X | X | X |
| Security | X | X | X | X |  |  | X |
| Cost |  |  |  | X |  |  |  |
| Accountability |  | X |  | X |  | X |  |

***UML Static and Dynamic Architecture View***

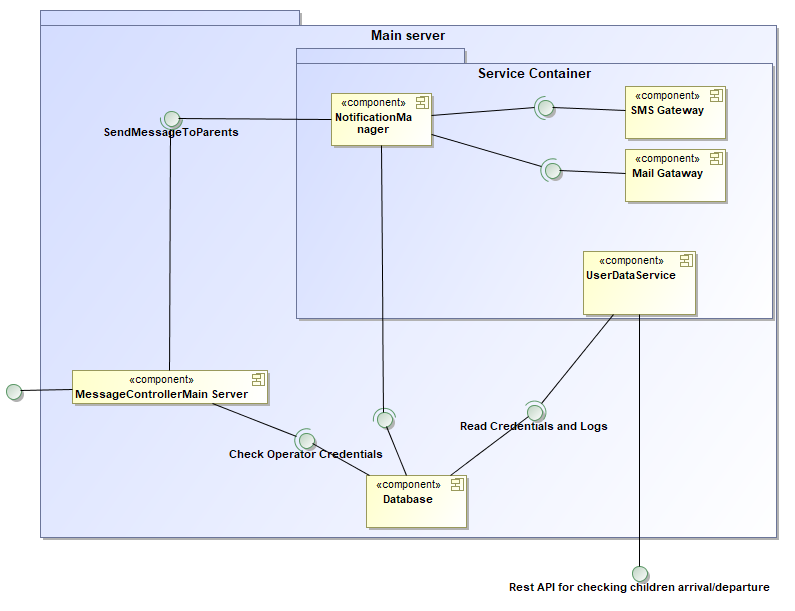
**Models**

Architecture of system is explained with following component diagrams:

1. Parent information and notification subsystem
2. Backup service
3. Access right service
4. Management service
5. CCTV service
6. Alarm service

**Component Diagrams**

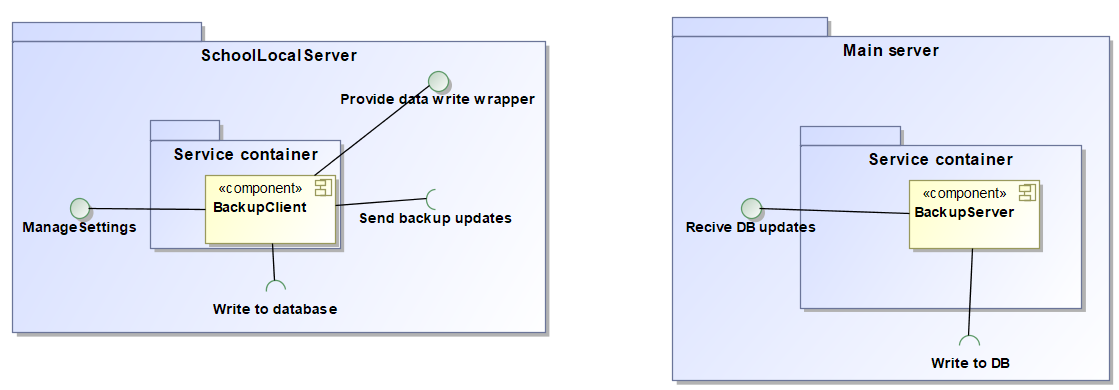
**1. Parent information and notification subsystem**



This service is divided in two parts user notifications and user data service.

* UserDataService provides informations to parents about children arrival departure times. Communication with parents is done trough REST API. Communication is encrypted using SSL to provide protection from eavsdropping and username, password are used to authorize parents. Service responds with JSON object containing array of children with their arrival/departure times.
* Notification service send messages to parent using sms gateway and email gateway in case of unexpected events and emergencies. It recieves data trough MessageControllerMain which checks authorization of requests before passing them to Notification service in case they are sent by operators. Notification service reads required contact information from DB given user id or school id in case of school wide notification.

**2. Backup service**



This service is divided in client and server side

* Backup client sends periodically DB changes to main server. To do this in most efficent manner it provides wrapper for DB writes. Periodically, after set time period, all changes written in that time are sent to server part for storage. This component has following interfaces:

1. DB write wrapper
2. Manage settings interface for changing backup interval time
3. Interface for sending backup updates

* Backup server recieves updates sent from backup client and applies them to DB on main server.

Message passing between client and server is done trough MessageControllers.

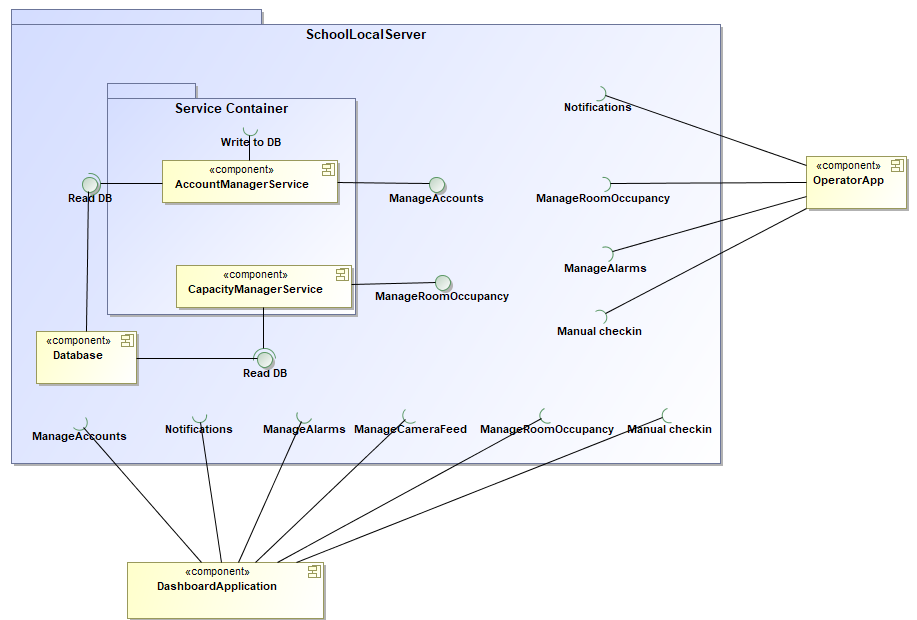
1. **Access right service**

# 

Access right service checks permissions of people to enter given areas. It recieves messages containing RFID id of person asking for permission and responds by granting or refusing permission to enter. Also there is interface for manual check-in by operators in case they need to grant access to users who forgot their RFID card or users without one. In this case MessageController check operator permission before passing message to Access right service. This interface is connected to Dashboard and Operator application.

Access right service uses interface provided by Backup service to write access logs and room occupation statistics to DB. Messages coming from RFID readers go trough RabbitMQ since they can be in large numbers and also replies from Access right service go trough MQ.

1. **Management service**

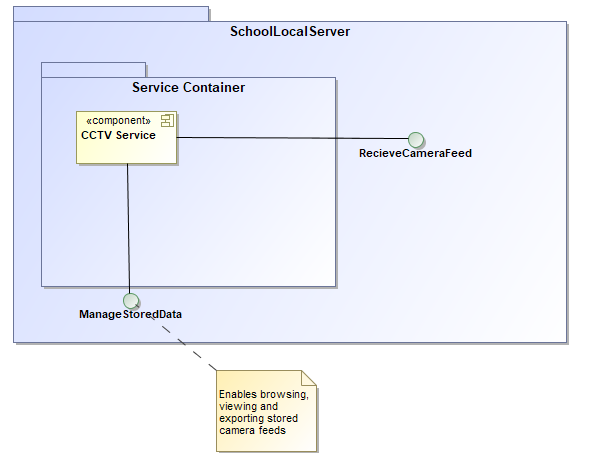


Management service connects Dashboard and operator application with interfaces provided by other services (Alarm service, Notification service, CCTV service, Access right service and Backup service). Besides these existing services it contains two sub services:

* CapacityManager services, this service provides insight into current room occupancy and enables managing (setting) room capacity limit
* AccountManager service this service enables creating, viewing and modifying user accounts.

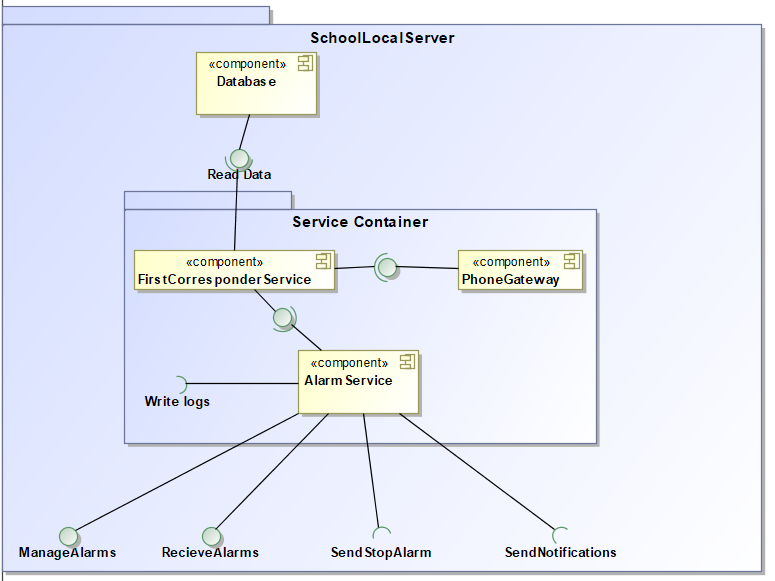
Authorization of request toward other services is done on MessageControllers so services itself can be only focused on their primary duty. Mechanism used is based on permanent device key assigned to each of operator devices and temporary token value assigned each time user logs in.

1. **CCTV Service**



CCTV service is tasked with storing camera feeds arriving trough MessageController. Video feed is stored on HDD which uses RAID for redundancy. Service also provides API to browse, view and export stored video feeds. This API is used by Dashboard application.

1. **Alarm service**



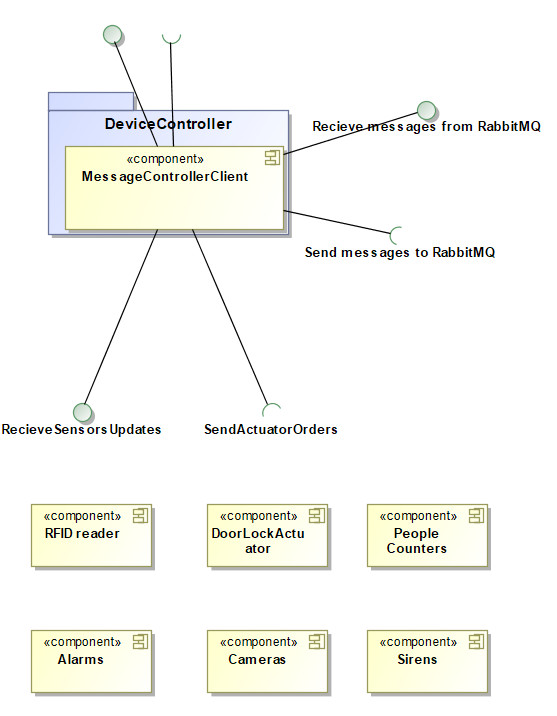
Alarm service is tasked with alarm handling. It provides following interfaces:

* Recieve alarm – recieve notification from MessageController orignating from sensors or operators. All alarms are logged in database trough BackupClient interface.
* Manage alarms – enables operators to stop false alarms, this interface is connected with Dashboard and Operator application.
* Send notifaction – sends messages to parents of school students trough MessageController using Notification service.
* SendStopAlarm – sends message to siren actuators trough school to stop alarm sound. This interface is connected to MessageController and messages are delivered to siren actuators.

Unlike other services Alarm service runs both in local school server and main server, this is done to increase availability since this is one of systems key features.

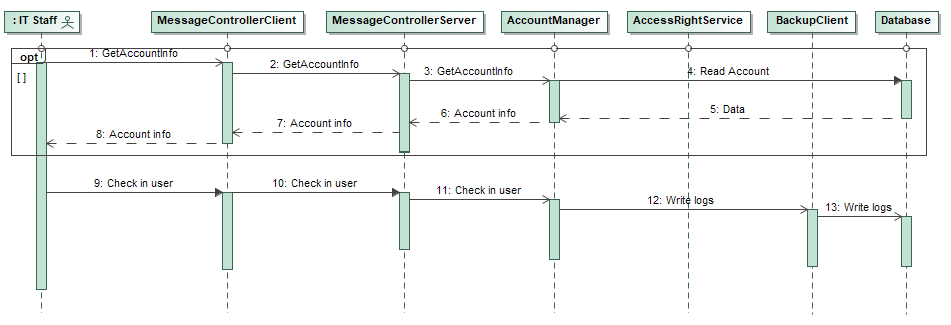
Another subservice is First corresponder service which provides automatic emergency calls to appropriate emergency services depending on alarm type.

Another part of system is Device controller this component is used to connect sensors and acctuators in building with services running on SchoolLocalServer. This part is only used for passing messages to appriopriate location.



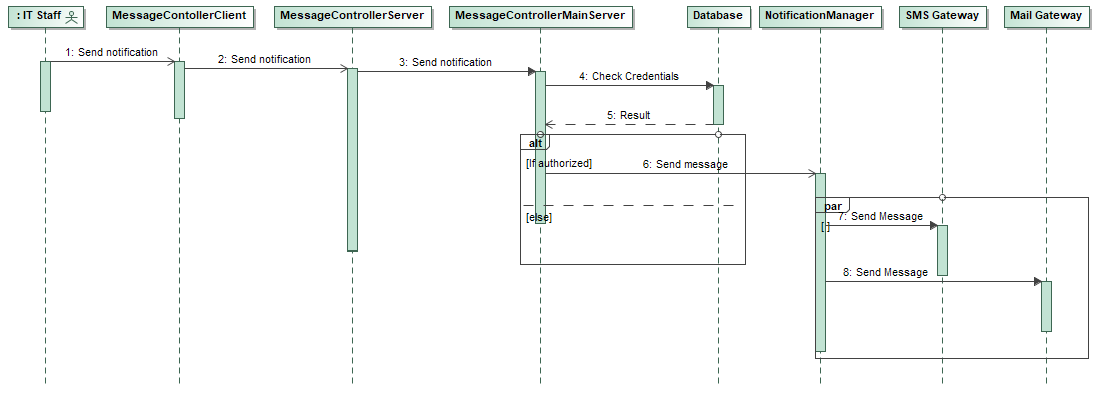
**Sequence Diagrams**

1. **Mark entry/exit of people**

****

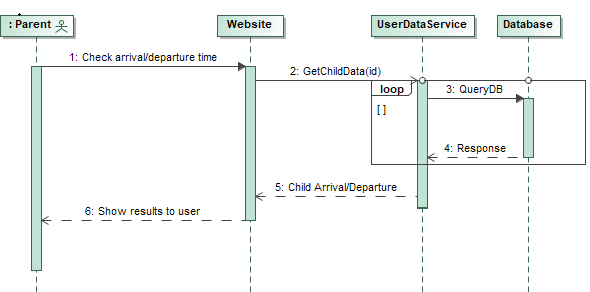
* An entry/exit can be marked manually for a child that forgot his card. First, the MessageControllerClient query the MessageControllerServer and then the AccountManager. Now check in DB if the child has an account and if he has enough permissions for the room where he’s trying to entry.
* Once the operator knows that the child has permissions he can add manually the entry and save the log for this event.

1. **Send notification to parents**

****

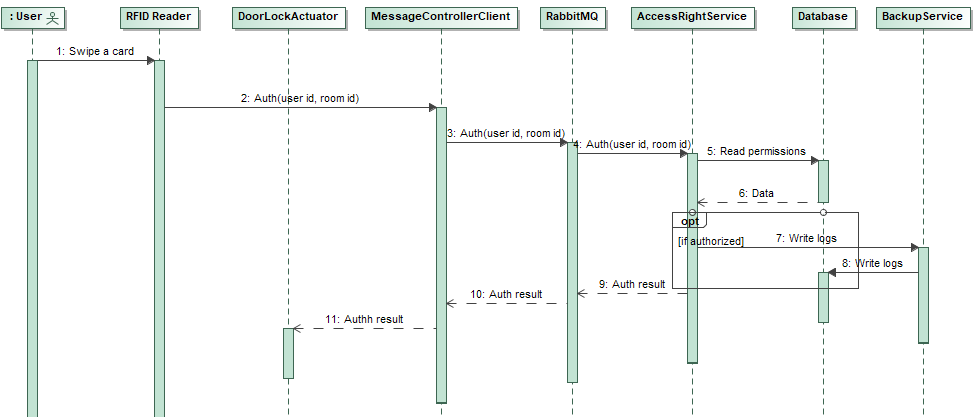
* Notifications are sent only from authorized people, operators, passing through the MessageControllerClient, MessageControllerServer and MessageControllerMainServer.
* The MessageControllerServer check the credentials of the operator and forwards the message if he has permission.
* The message is sent to the NotificationManager that send a mail and an sms to the user.

1. **Check child arrival and departure time**



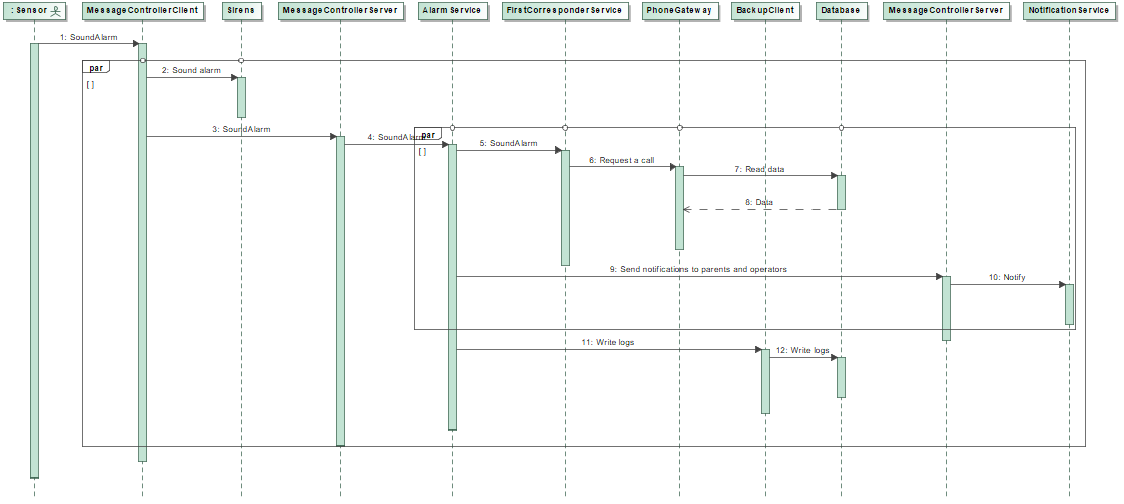
* Parent or an authorized person connected with a child, can check the arrival or departure time of the child through the website.
* Website sends a request to UserDataService passing username, password of user in additional request header.
* UserDataService queries DB and replies (if user is authorized) with list of the entries/exits of the child with timestamps for last 30 days. Result is returned for all children connected with user account.

1. **Gain access to authorized areas**



* User swipes his card on the RFID Reader located in a specific room or in the main entry of the school.
* The request is sent to the MessageControllerClient and fowarded trough RabbitMQ queue.
* Access right service, checks user permissions and room capacity by querying DB. If all checks pass it returns message back trough message controllers and RabbitMQ to apporopriate gate which will unlock it.

1. **Sound alarm**

****

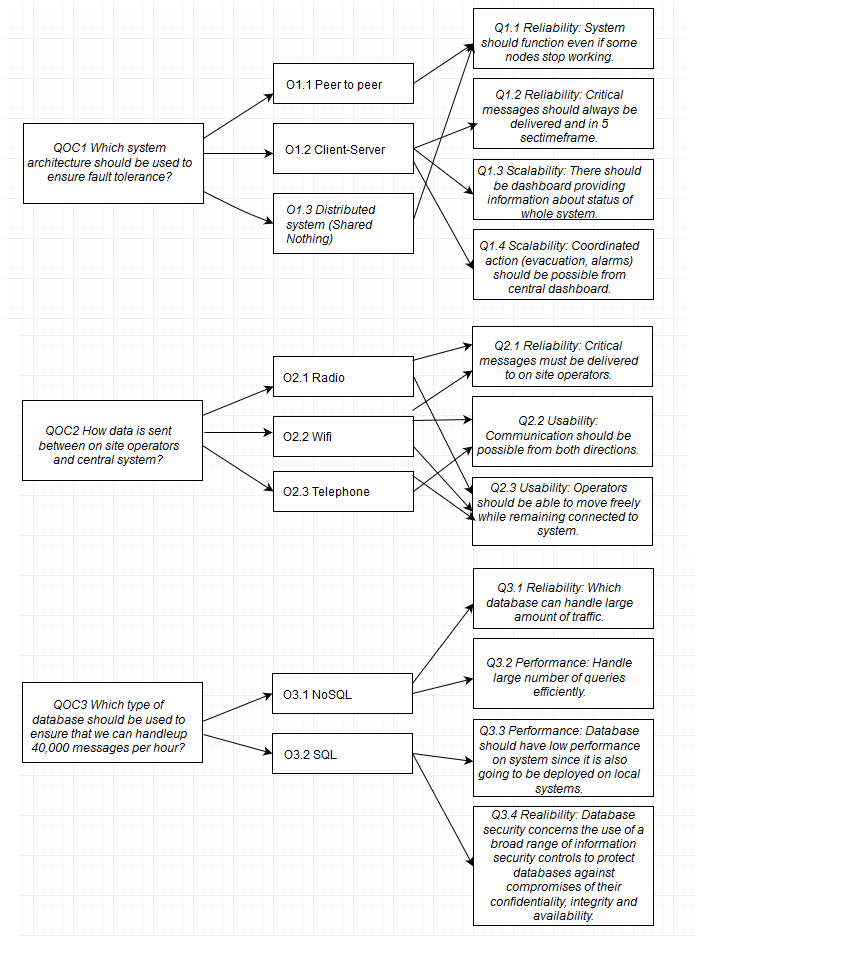
* In case of emergency, sensors notify MessageControllerClient which sound sirens automatically.
* At the same time the alarm is sent to the MessageControllerService and then to the AlarmService that will check in the DB the number to call associated to the current emergency do an automatic call.
* The staff of the school and children parents are notified.
* Logs of the event are saved in the DB.

# Design Decisions

|  |  |  |
| --- | --- | --- |
| **Concern (Identifier: Description)** | | *Con#1: Which system architecture should be used to ensure fault tolerance?* |
| **Ranking criteria (Identifier: Name)** | | *Cr#1.1 Reliability*  *Cr#1.2 Scalability* |
| **Options** | **Identifier: Name** | *Con#1-Opt#1: Peer to peer* |
| **Description** | *Is a distributed application architecture that partitions tasks or workloads between peers. Peers are equally privileged, equipotent participants in the application. They are said to form a peer-to-peer network of nodes.* |
| **Status** | *This option is rejected.* |
| **Relationship(s)** | *-* |
| **Evaluation** | *Cr#1.1: The decentralized nature of P2P networks increases robustness because it removes the single point of failure. If one peer on the network fails to function properly, the whole network is not compromised or damaged.*  *Cr#1.2: In P2P networks, clients both provide and use resources. This means that the content-serving capacity of peer-to-peer networks can increase as more users begin to access the content.* |
| **Rationale of decision** | *This option is rejected because in our systems each node is different and we only the main server need a big load of data.* |
| **Identifier: Name** | *Con#1-Opt#2: Client-server* |
| **Description** | *Is a distributed application structure that partitions tasks or workloads between the providers of a resource or service, called servers, and service requesters, called clients. A server host runs one or more server programs which share their resources with clients. A client does not share any of its resources, but requests a server's content or service function.* |
| **Status** | *This option is decided.* |
| **Relationship(s)** | *-* |
| **Evaluation** | *Cr#1.1: In our case to ensure fault tolerance client will have certain degree of autonomy being able to function in degraded mode without central server. Also in case client stops working central server can take over its role ensuring system continues to function.*  *Cr#1.2: Client will have local storage which will enable them to function autonomously, but they will only store data relevant to their school. Main server will store all data and offer some shared services (e.g. SMS gateway, parent checking arrival/departure etc.)* |
| **Rationale of decision** | *We choose this architecture since the main server is designed to operate as a centralized system that serves many clients (all the schools connected). Each school has its own and independent system that doesn’t affects others.* |
| **Identifier: Name** | *Con#1-Opt#3: Distributed system (Shared Nothing)* |
| **Description** | *A shared-nothing architecture (SN) is a* [*distributed-computing*](https://en.wikipedia.org/wiki/Distributed_computing) *architecture in which each node is independent and self-sufficient, and there is no single point of contention across the system. More specifically, none of the nodes share memory or disk storage. Distributed architecture offers very high level of fault tolerance by making duplication of services on every node (school).* |
| **Status** | *This option is rejected.* |
| **Relationship(s)** | *-* |
| **Evaluation** | *Cr#1.1: The advantages of SN architecture versus a central entity that controls the network, include eliminating any single point of failure.*  *Cr#1.2: Every service, memory or resource are totally independent.* |
| **Rationale of decision** | *This option was a very good alternative since each school is independent, but we can’t say the same speaking about interaction between the local system of a school and the main server.* |

|  |  |  |
| --- | --- | --- |
| **Concern (Identifier: Description)** | | *Con#2: How data is sent between on site operators and central system?* |
| **Ranking criteria (Identifier: Name)** | | *Cr#2.1 Reliability*  *Cr#2.2 Usability* |
| **Options** | **Identifier: Name** | *Con#2-Opt#1: Radio* |
| **Description** | *Is the technology to carry information through the transmission of signals by modulation of electromagnetic waves with frequencies below those of visible light.* |
| **Status** | *This option is rejected.* |
| **Relationship(s)** | *-* |
| **Evaluation** | *Cr#2.1: Data are sent safely but it’s not easy to track them.*  *Cr#2.2: It’s not a method that everyone can use.* |
| **Rationale of decision** | *Radio communication requires specific devices that usually are not common inside a school. Wifi, instead, is a service that actually we can find anywhere.* |
| **Identifier: Name** | *Con#2-Opt#2: Wifi* |
| **Description** | *WiFi is a technology for wireless local area networking with devices based on the IEEE 802.11 standards. Many different devices can use Wifi and connect to the Internet via a WLAN and a wireless access point.* |
| **Status** | *This option is decided.* |
| **Relationship(s)** | *-* |
| **Evaluation** | *Cr#2.1: It’s possible to make the system safe with a VPN.*  *Cr#2.2: Operators should be able to move freely while remaining connected to system and this is possible with some access point.* |
| **Rationale of decision** | *Wifi is widely available technology which enables duplex communication. Operator must be able to send responses to our system while moving, this makes wifi as best choice.* |
| **Identifier: Name** | *Con#2-Opt#3: Telephone* |
| **Description** | *Is a* [*telecommunications*](https://en.wikipedia.org/wiki/Telecommunication) *device that permits two or more users to conduct a conversation when they are too far apart to be heard directly. A telephone send sound (typically and most efficiently the human voice) or sms text into electronic signals that are transmitted via cables and other communication channels to another telephone which reproduces the data to the receiving user.* |
| ***Status*** | This option is rejected. |
| **Relationship(s)** | - |
| **Evaluation** | *Cr#2.1: Telephone allow duplex communication and gives reply that assure that an important message has been sent.*  *Cr#2.2: The development of digital data communications method, such as the protocols used for the Internet, it became possible to digitize voice and transmit it as real-time data across computer networks, giving rise to the field of Internet Protocol (IP) telephony, also known as voice over Internet Protocol (VoIP).* |
| **Rationale of decision** | *Telephone is also one of alternatives but recognizing operators voice input is problematic especially in crowded places, but nevertheless phone (SMS or mobile internet) can be used as backup way to send alarms to operators in case wifi network is down.* |

|  |  |  |
| --- | --- | --- |
| **Concern (Identifier: Description)** | | *Con#3: Which type of database should be used to ensure that we can handle up 40,000 messages per hour?* |
| **Ranking criteria (Identifier: Name)** | | *Cr#3.1 Reliability*  *Cr#3.2 Performance* |
| **Options** | **Identifier: Name** | *Con#3-Opt#1: No-SQL* |
| **Description** | *A NoSQL or non-relational database provides a mechanism for storage and retrieval of data that is modeled in means other than the tabular relations used in relational databases.* |
| **Status** | *This option is rejected.* |
| **Relationship(s)** | *-* |
| **Evaluation** | *Cr#3.1: NoSQL databases do not offer as much security as provided by a relational database.*  *Cr#3.2: NoSQL is database created to handle large amounts of unstructured data.* |
| **Rationale of decision** | *This option is rejected because the security features of the relational database are far better than those offered by NoSQL. Furthermore, each school has its own database that doesn’t have to manage a large amount of data.* |
| **Identifier: Name** | *Con#3-Opt#2: SQL* |
| **Description** | *SQL (Structured Query Language) is a standardized programming language used for managing relational databases and performing various operations on the data in them.* |
| **Status** | *This option is decided.* |
| **Relationship(s)** | *-* |
| **Evaluation** | *Cr#3.1: Relational database security includes integrated features such as role-based security, encrypted communications and support for row and field access control, as well as access control through user-level permissions on stored procedures.*  *Cr#3.2: SQL can handle in simpler and faster way structured data of lower volumes.* |
| **Rationale of decision** | *We will have SQL database because the amount of data is not going to be large and they are less resource intensive and simpler to use. Also, data will be divided among schools, with each school saving only their data.* |

**

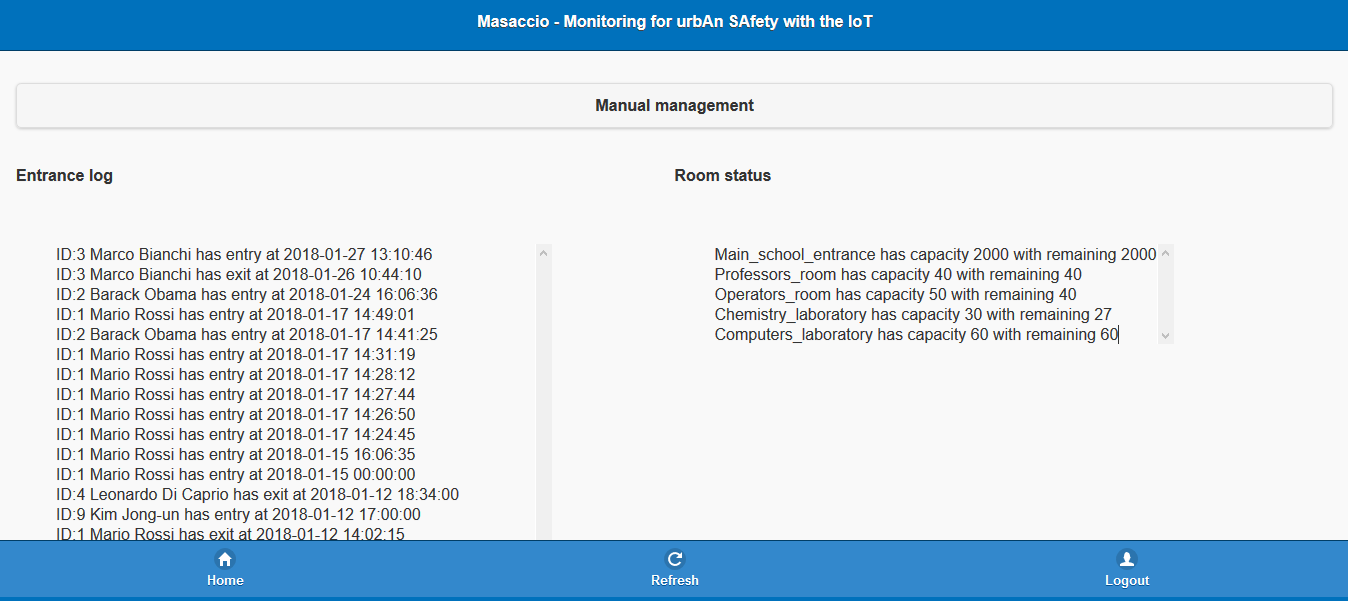
# From Architecture to Code

**Operators dashboard**

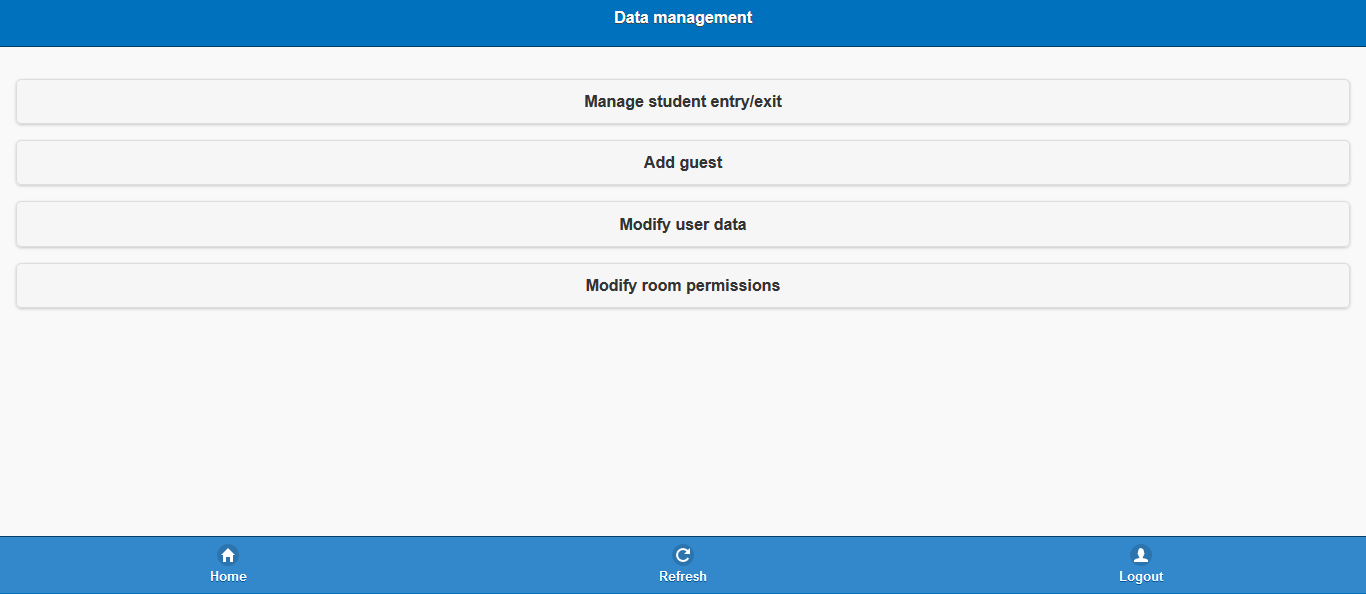
We realized a dashboard that allows operators to check and manage all the security issues about the school where they are working.

The dashboard is realized with HTML code, PHP that create a login session and allows direct connection with the local db getting and sending continuously information from and to it and Javascript that allows to implement function.

Logs and other data are taken from db with Json call and are sent with a POST request.

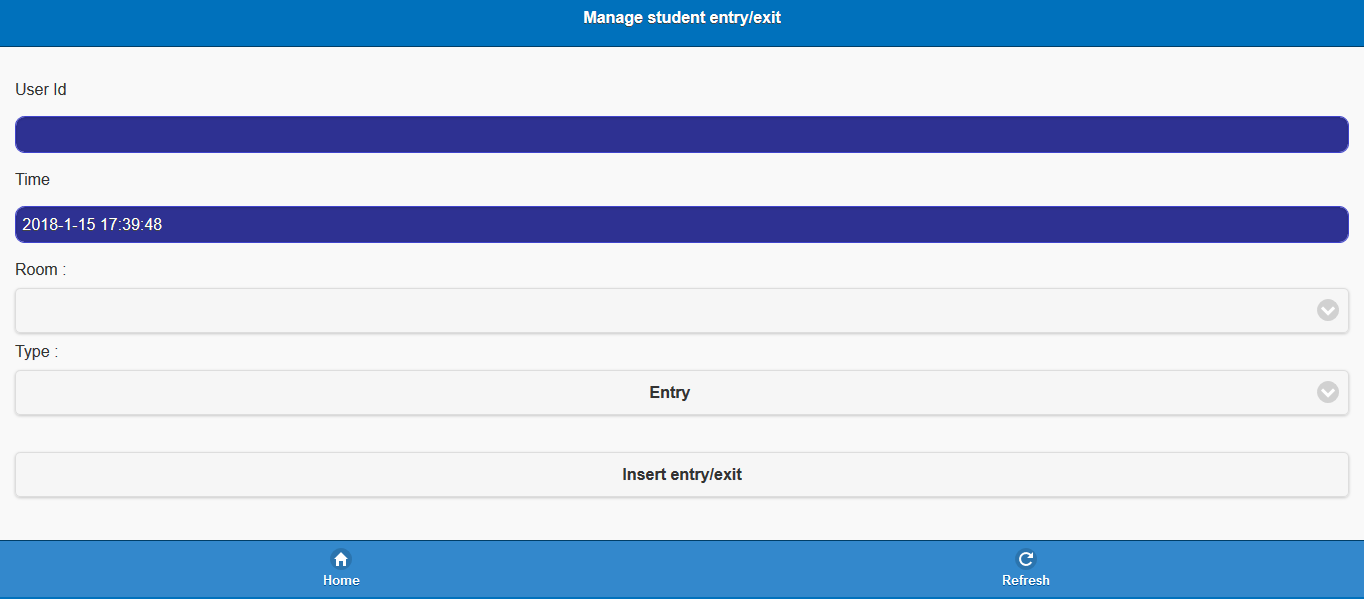


The main page shows the Entrance log of all the RFID card reader and tells the operator who is entering with the relative timestamp. In the same page there is also a list that shows the current capacity of all the rooms inside school. Furthermore, the operator can decide to logout in every moment.

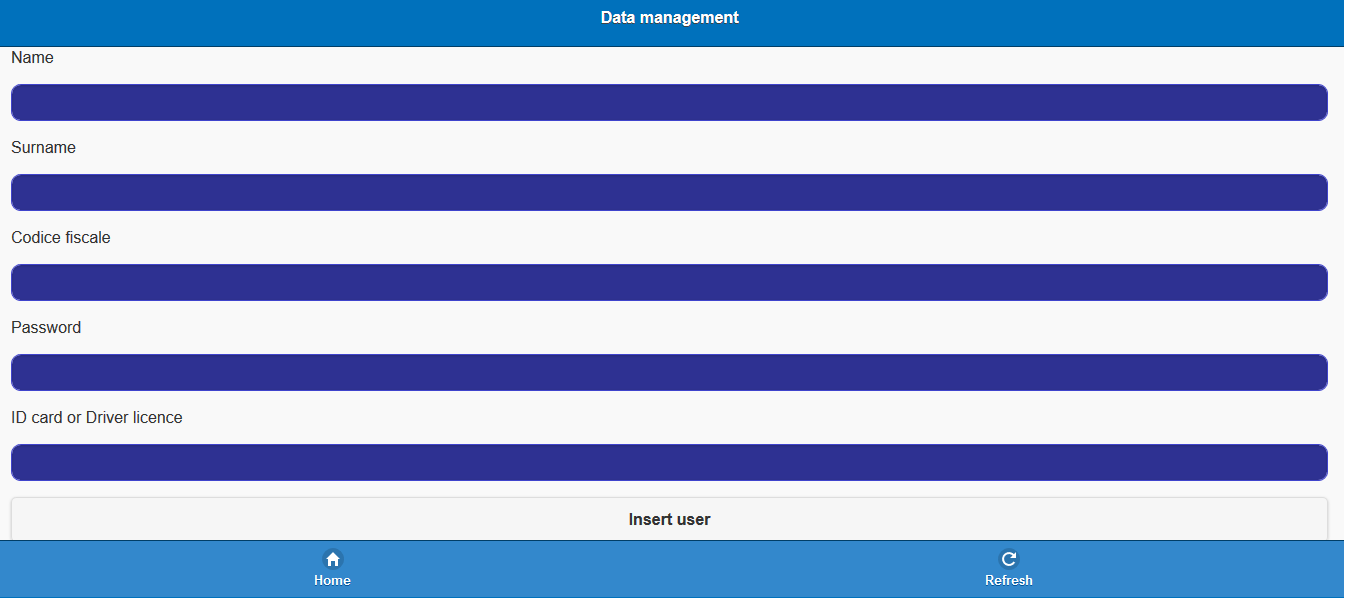


From the index, clicking on Manual management, an operator can insert manually some data:

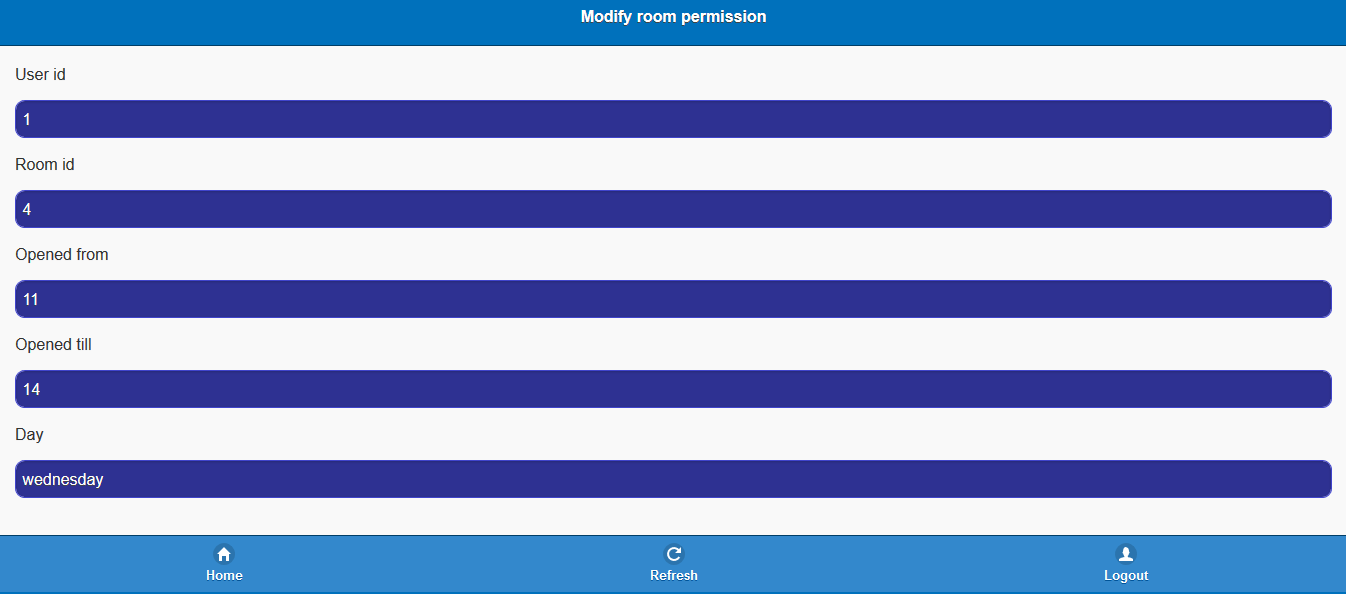
* Manage student entry/exit - in case a student forgot his RFID card;
* Add guest - in case a relative that want to entry doesn’t have an account;
* Modify user data - modify data about user (like contact info or other data), add a new user or delete an existing one.
* Modify room permissions – give, remove or add permission to someone in a specific room;



Is possible to add an entry/exit simply inserting the user id (a unique code that identify a user), the room where is trying to entry, the type (entry or exit) and the time (current time is given automatically).



Guest and user management are similar, the only difference is that guests don’t need password, since they will not have an account.



This page allows to change user’s permission for a specific room. The operator must enter the user id and choose the permission that he wants to change (or also delete). In this page all data related to the specific permission is pick from database and shown to the operator that can do changes.

**Parent information service (UserDataService)**

**Parents dashboard**

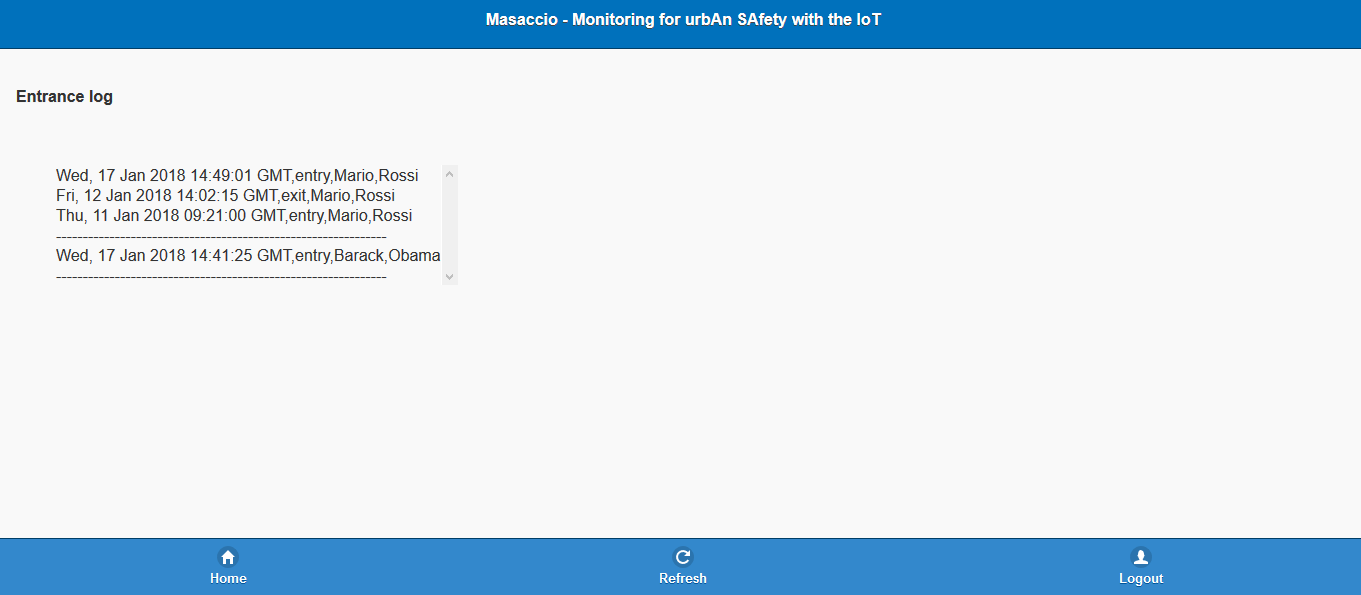
This dashboard allows parents to check all the entries of their children, restricted to the main entrance (parents will not see if a child is inside the computer’s room).

The dashboard has a login session that is connected with a local server created with Python code.

We have an HTTP basic authentication that receives credentials through a GET request.



After the login, parent will see all the logs related to his children about the last 30 days, with each child’s entrance separated with a line.



**Parents REST server**

Python flask is used on server side to implement REST API. Currently we provide only one service returning children logs. HTTP Basic auth is used to authorize parents, this method consist of sending additional header in request. Header must be in following format (according to specification of HTTP Basic auth):

Authorization: Basic Base64(username:password)

Since REST is stateless API requires that in each request we send this header.

**Access right service**

Beside dashboard we also implemented access right service and message passing code to enable its function. Implemented pipline consists of message passing from RFID reader trough message controllers and RabbitMQ to access right service back, fowarding response to gate actuators. Following files implement this service:

* **rfid\_reader.py** reader implementation, in current iteration it returns hard coded data.
* **gate.py** implementation of gate logic, it consists of server listening for messages from controller. Currently same as with rfid\_reader logic for controlling hardware is not implemented.
* **message\_controller\_client.py** implementation of local building controller. Message controller is currently taking messages obtained from reader and fowarding them to appropriate RabbitMQ queue. At the same time, it listens for new messages about gates and forwards those messages from RabbitMQ to gate in question.
* **message\_controller\_server.py** implementation of local school message controller. It listens to new messages from RabbitMQ in RFID queue and forwards them to acess right service and also routes messages from access\_right service in appropriate queue.
* **access\_service.py** contains main logic for this service. Given id of rfid reader and user id it checks user permissions. It takes in consideration also time of request and room occupancy. Response to *gate.py* is returned trough *message\_controller\_server.py*. In case that user has sufficient rights and room has enough free space, it also logs time of access into database and updates room occupancy informations for other services to use.



Beside standard python libraries, in this implementation we used RabbitMQ to achieve message passing through queues and pika library as client for RabbitMQ. Standard RabbitMQ message flow is shown on picture above.

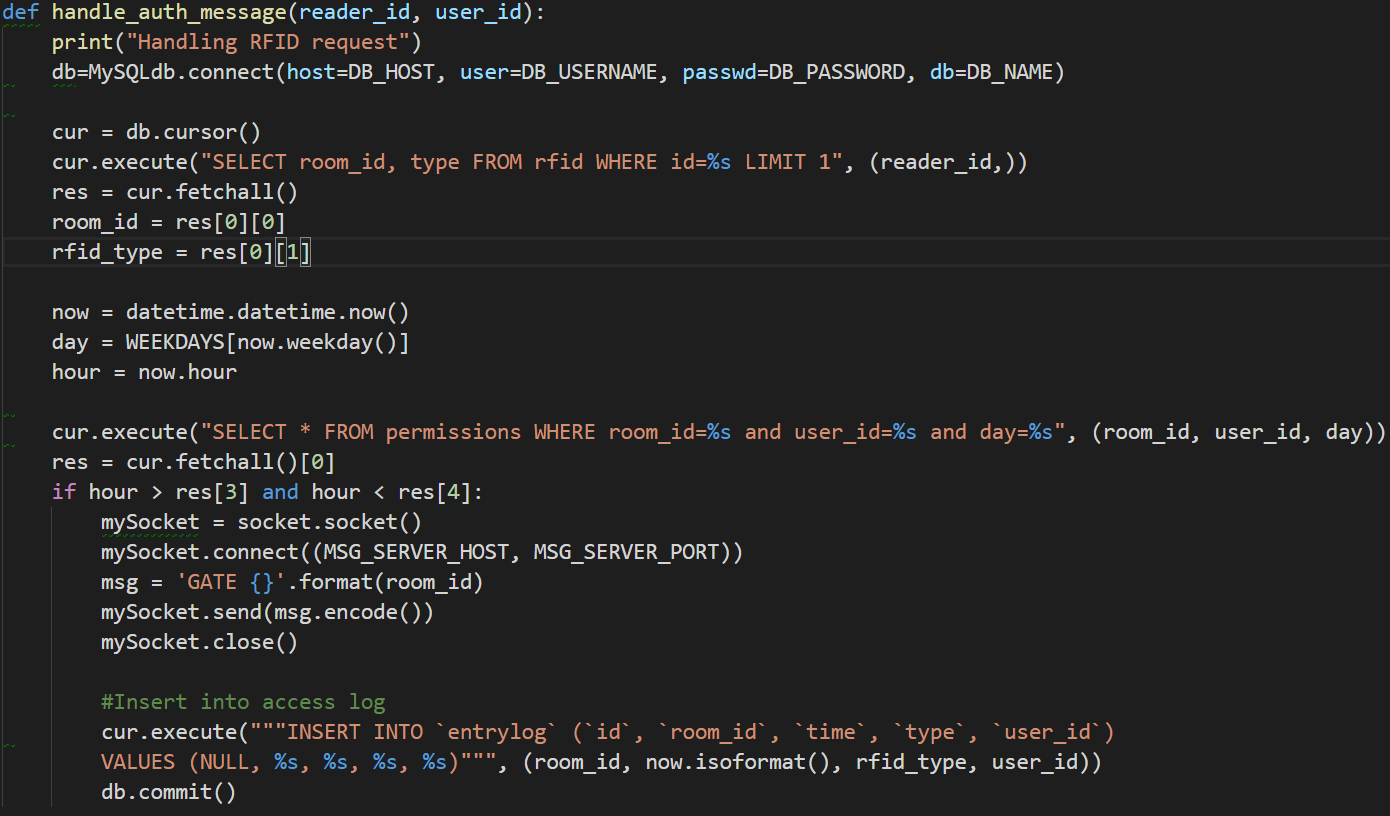


Figure 1 Main logic of access right service

**Database**

# 

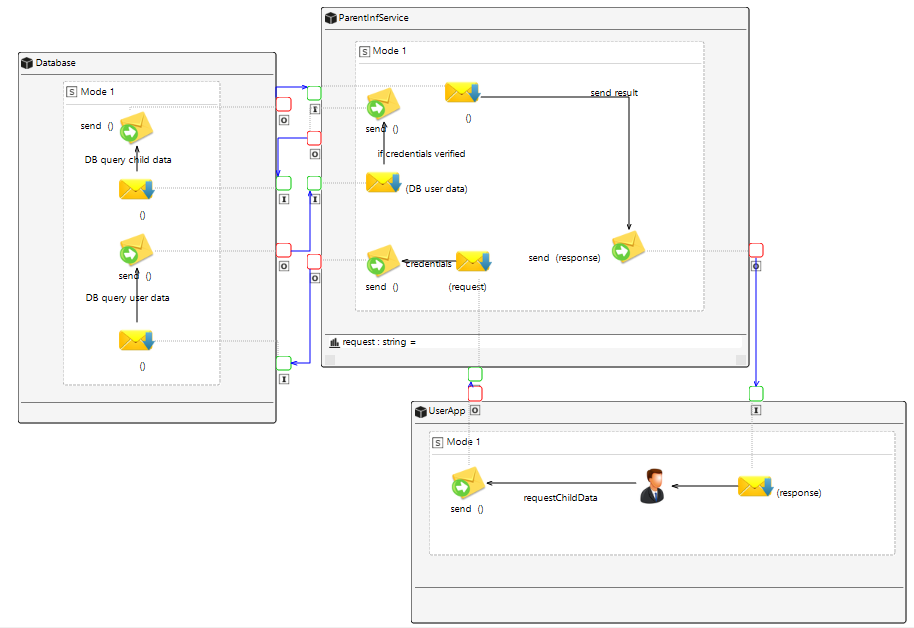
* **alarm**: the type of alarm that can happen (fire,earthquake…) with the relative number that will be automatically called.
* **alarmlog**: alarm event is saved here with a timestamp and its type. There is also one data between user\_id or sensor\_id, since alarm can be raised up also manually from operators.
* **contact\_info**: contacts that are associated to users, extremely necessary for parents.
* device: operators have device that are authorized for accesses.
* **door\_lock\_actuator**: give information about the room where actuator is placed.
* **entrylog**: save all the entry/exit events.
* **parents**: one or more parents or relatives are connected to one child.
* **permissions**: each user can gain permission to entry in a specific a room in a specific range of time.
* **rfid**: an rfid read is positioned in the main entrance. Also other particular room can have a rfid with a type that identify the entry and the exit.
* **room**: a room has a name, a capacity and a current capacity that must be monitored.
* **school**: since all this data are sent to the main server, we have to identify a specific school with its name and location.
* **sensors**: a sensor is positioned in a room and has a type (fire, earthquake).
* **status**: we need this table to distinguish users (professors, guests, students, operators).
* **user**: basic and needed information that identifies people.

CAPS Architecture View

CAPS SAML

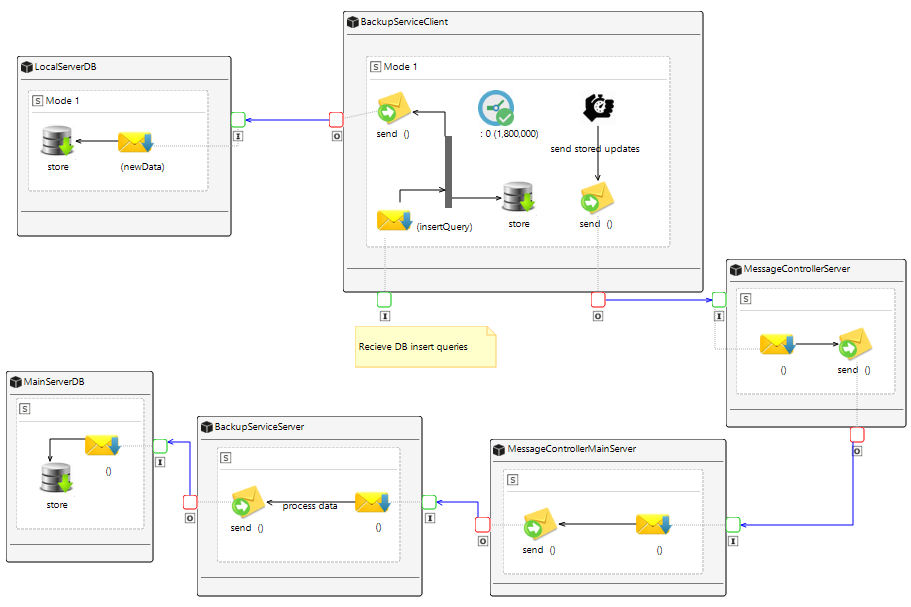
SAML shows combination of static and dynamic view of system. We provide SAML diagrams for following services:

1. Parent information service
2. Backup service
3. Access right service
4. Alarm service
5. **Parent information service**



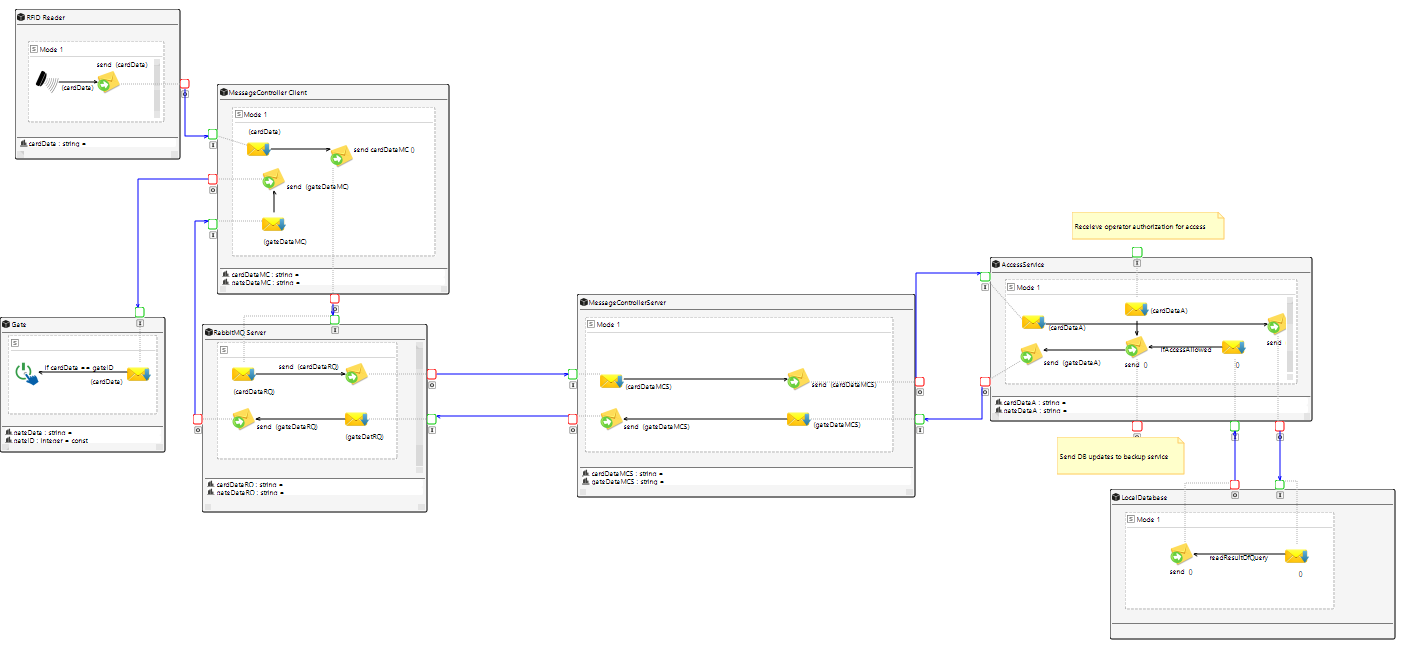
User sends request for children arrival/departure logs to ParentInformationService. Service checks user credentials, and if user is authorized returns logs regarding children associated with that user.

1. **Backup service**



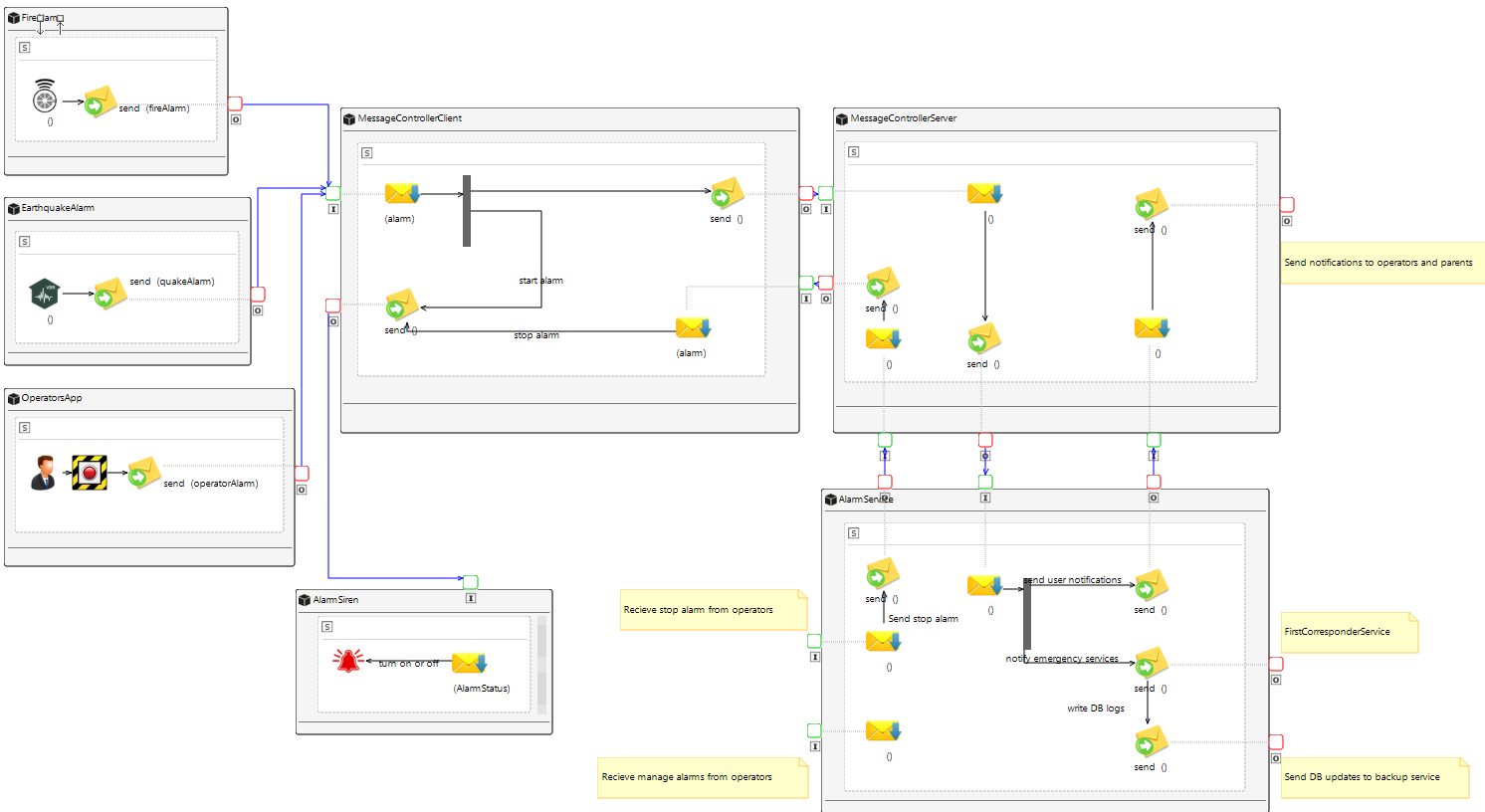
Backup service client recieves database writes from other services and stores them in DB local to that server. After timer expires (30 min) it takes all insert queries since last update and sends them to main server. On main server Backup service server waits for insert logs and applies them to main DB for backup.

1. **Access right service**



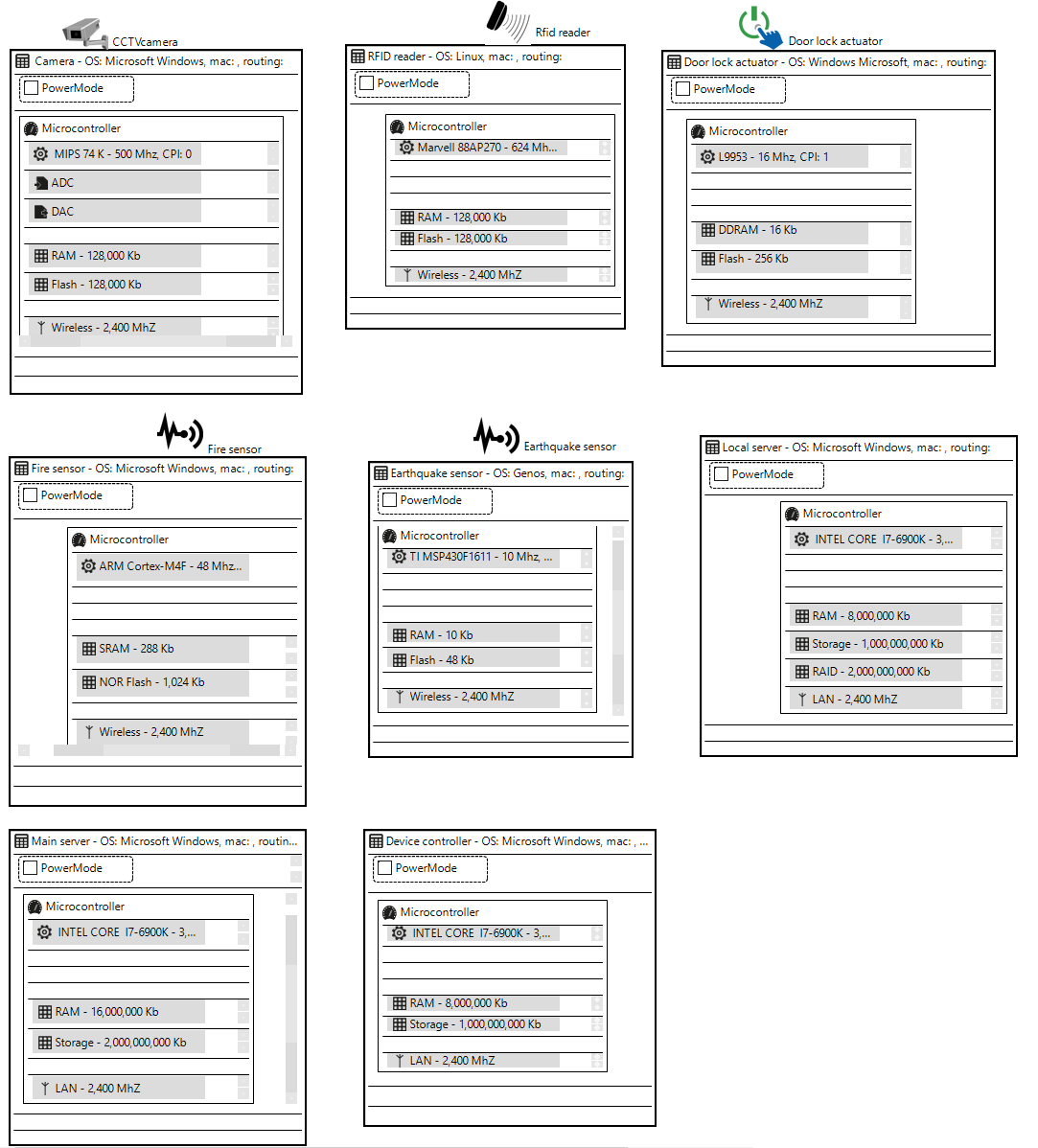
Access service waits for data from RFID readers. When data is recieved, it checks DB if user is authorized and in that case it sends gate open message to gate associated with given RFID reader. Higher quality image can be seen in project files.

1. **Alarm service**



Alarm service recieves alarms from sensors (fire, earthquake) and users. It notifies operators, parents in case of alarms as well as calling appropriate emergency services.

CAPS HWML



CAPS SPML

Here we describe the system of a school through SweetHome 3D.

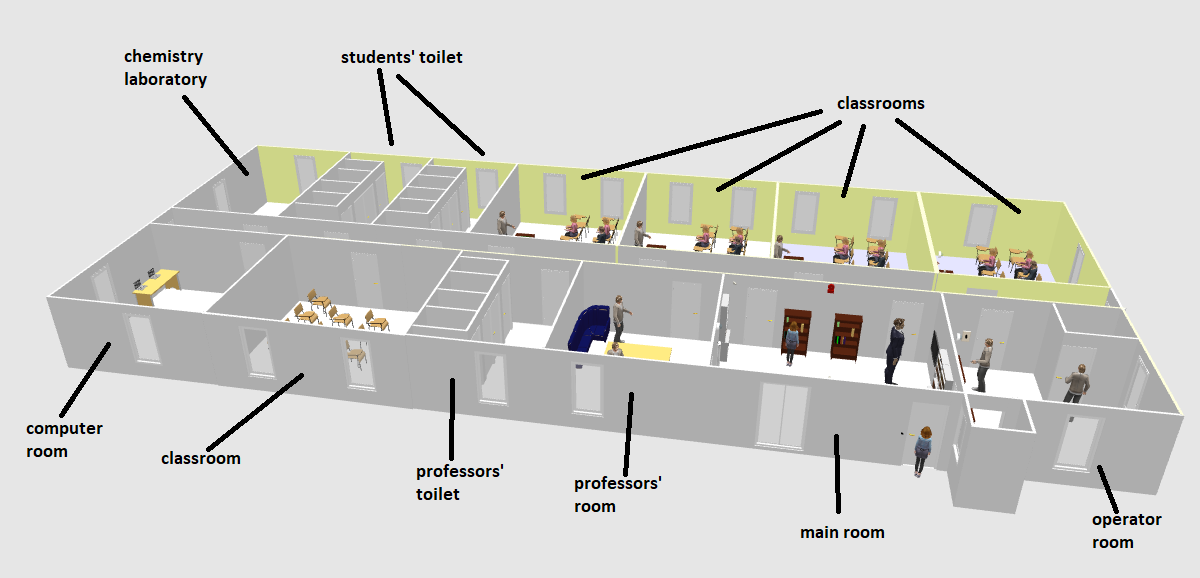
# 

# This is a general view of the school.

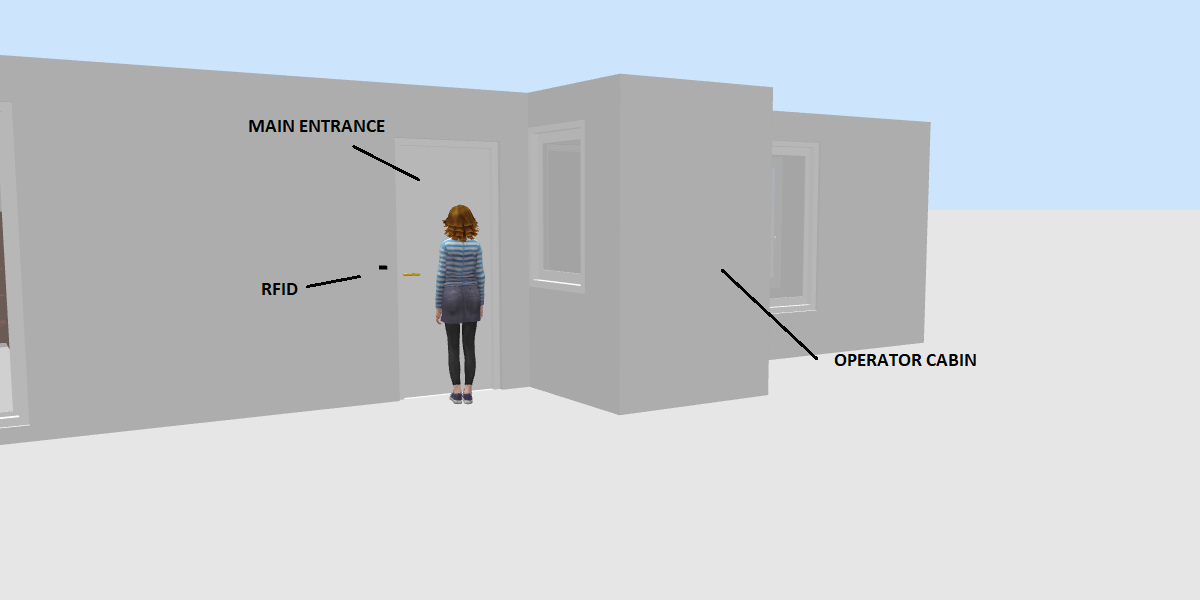
There are 5 classrooms, 3 toilet room (2 for students, 1 for staff), 1 computer laboratory, 1 chemistry laboratory, 1 operator room, 1 professor room and 1 main room.

2 emergency exits are positioned on the two sides of the corridor.

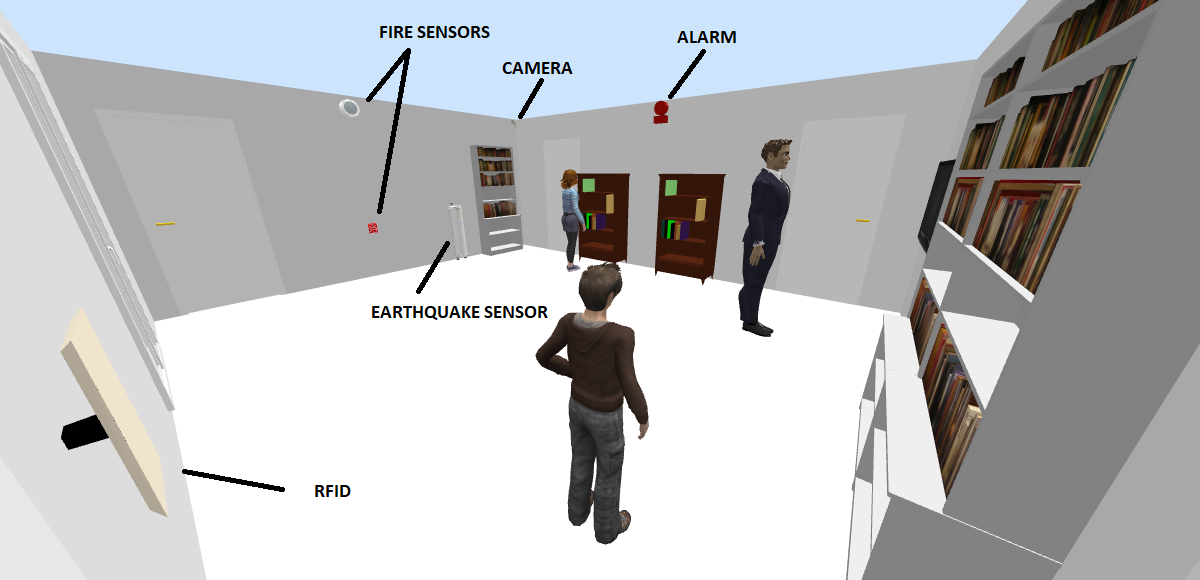
Main room, computer room, professor room, chemistry laboratory and operator room have RFID reader that check permissions and work also as people counter.



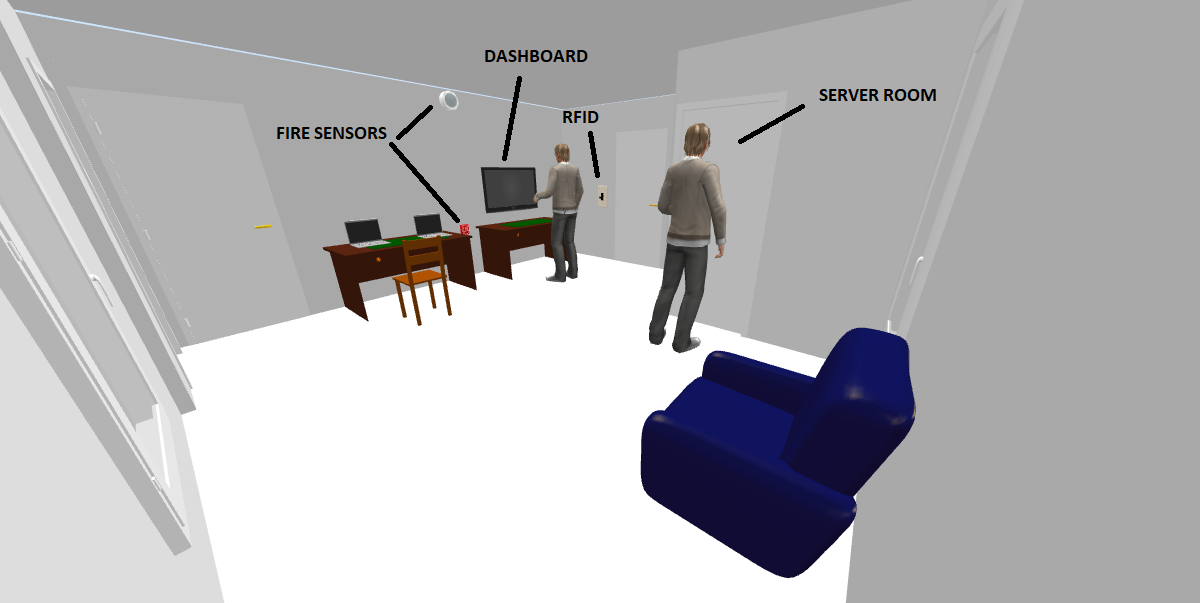
The main entrance (that has RFID reader) is positioned near an operator cabin, that gives help in case of necessity (ex. Operator can add manually entry/exit).



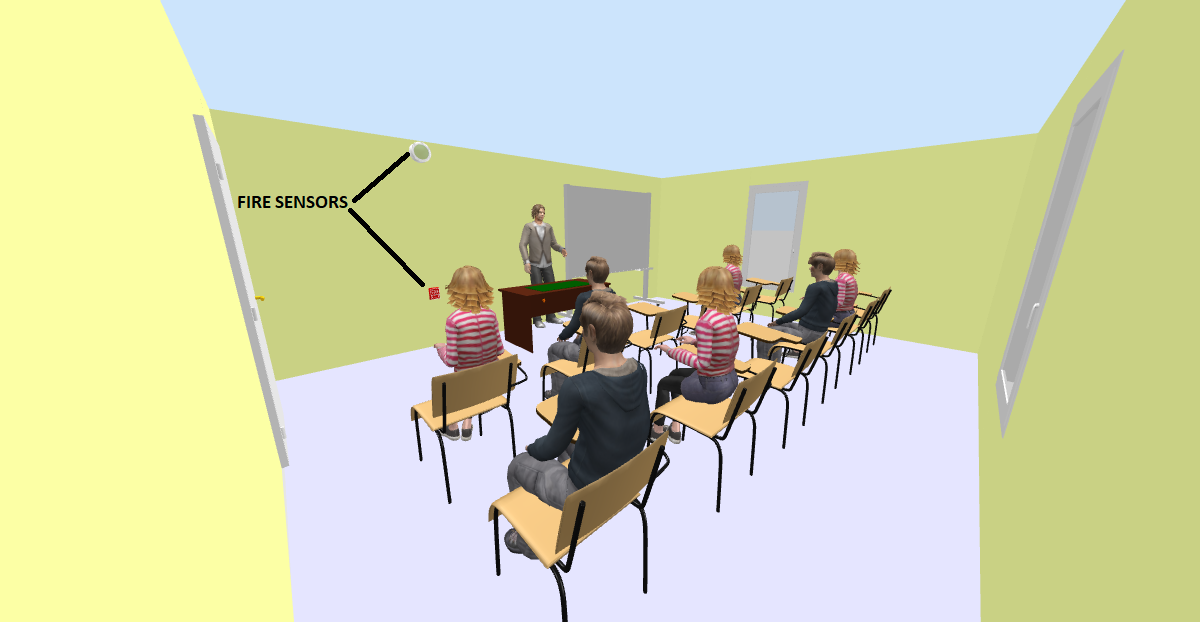
The first room after the entrance is the main room.



# On the right there is a door (with RFID reader) that gives access to operator room.



Inside the operator room there is another room that contains the local server of the school.



Classrooms have just fire sensors. No camera, no RFID reader.

# Summary

General architecture of system has been shown in *Informal Description of your system and its Software/System Architecture* section. More detailed behavior can be seen in *UML Static and Dynamic Architecture View.* Component diagrams show detailed view of static parts of system focusing on each of potential services, while sequence diagrams show general control flow key user stories of our safety system. Also we present CAPS diagrams, both HWML and SAML which provide different view on control flow of system and hardware components used.

In the last section we described implemented services. Access right service, its parts and external libraries used, and its implementation. Messages are passed using RabbitMQ message broker. Security of service will be achieved by keeping sensors on dedicated network to disable interaction of potential rogue sensors and systems with our code. We also described Parent information service which provides parents with ability to monitor arrival/departures times of their children from school.