Module 5- Computer Systems (2021-22)

UNIVERSITY OF TWENTE.

Software Design Document

Project Name: ***Raspberry Home***	Team ID: ***31***
Team Members: ***Damon Kaewborisut*** ***Niek Zieverink*** ***Masis Zovikoglu*** ***Shuhang Tian*** ***Yifan Sun*** ***Jelke Schröder***	Mentor(s): ***Filip Ivanov*** ***Ricco Pratama Halim***

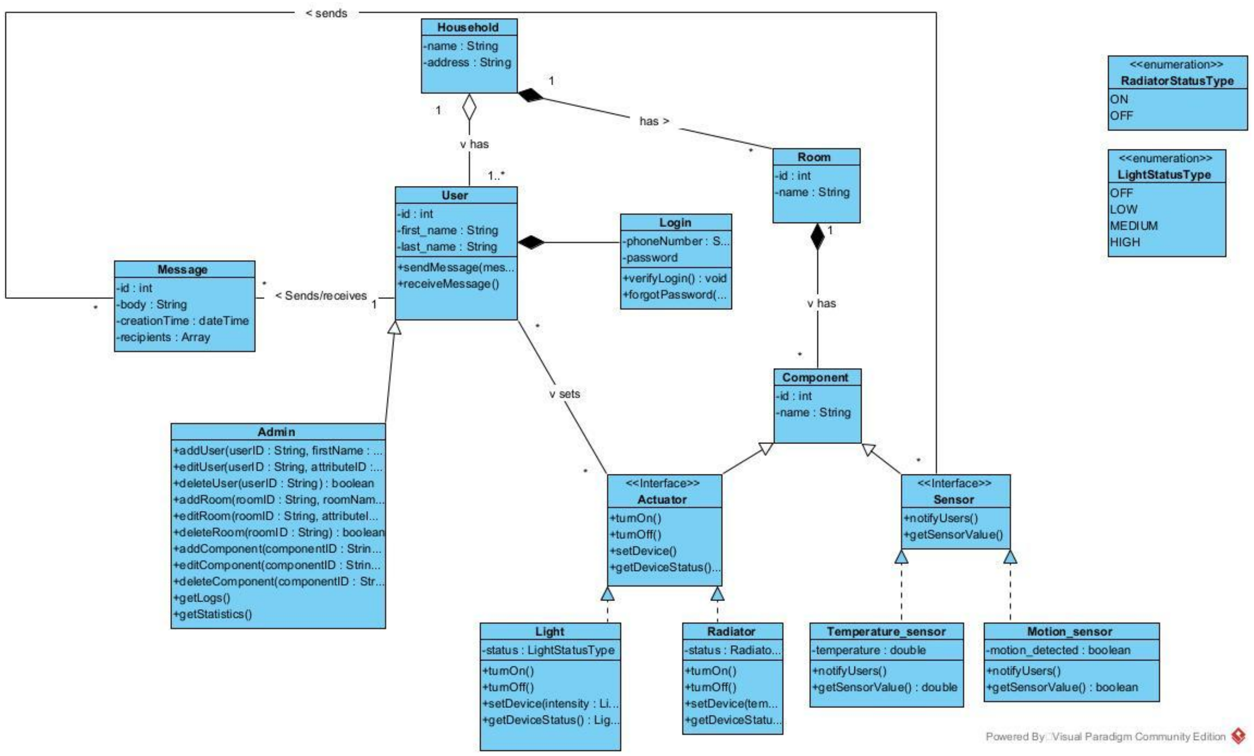
1. Introduction

Raspberry Home is a Home Automation project centered around easier house management and maintaining a positive effect on the environment. The system consists of a web application that can be connected to a raspberry pi 4 in a household and is able to send a message to users connected to this household. Multiple users can connect to the same web application at the same time and manage their home from there. On the web application, the users can turn on and off the lights and temperature in different rooms. They can also see the status of the room. The system will use different types of sensors to get awareness of the temperature and the amount of light in the house. An algorithm will be used to try and maintain the least amount of energy used to give the user the feel of a comfortable home and will evaluate the user with a score on how environmentally friendly the household is.

All data will be protected using user authentication in a way that only the user of a household can access information concerning the household. All personal information will be encrypted and stored in the database.

3. Architectural Design

Class diagram



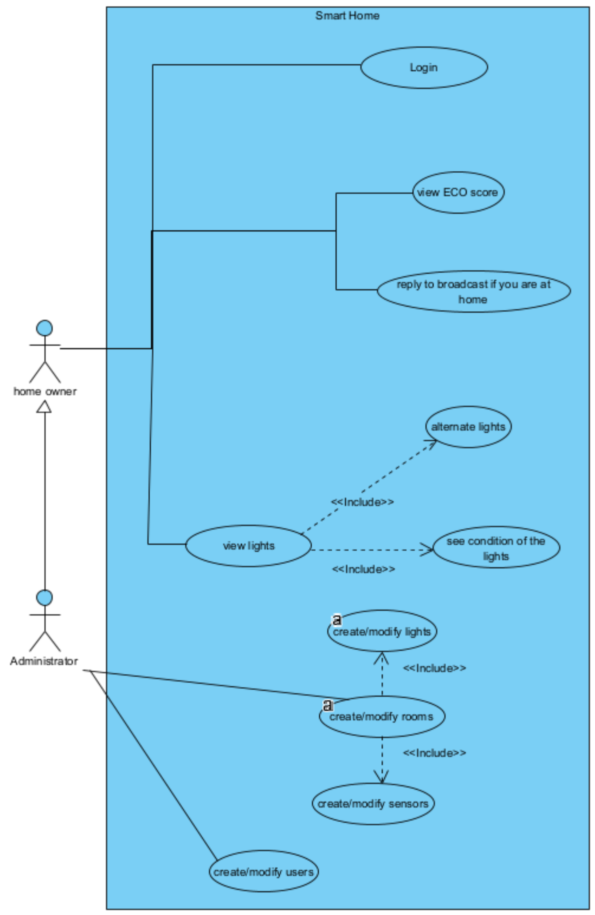
We created a class diagram in order to get a good overview of the entire system. As becomes clear when looking at the class diagram, the ‘top’ class is the Household class. Every household can have multiple rooms and multiple users (members of the household).

Users can either be regular users or admins. Admins can do the same things as regular users but they also get to use some management functions like adding, editing and deleting users, rooms and components to and from the household.

Every room can contain multiple components as well. Components are abstract classes and can be either actuators or sensors. Actuators and sensors are implemented as interfaces as they only describe which functionalities the actual components should have. The actual components (temperature sensors, motion sensors, lights and radiators) should implement these functionalities so they can be used by the users. Actuators can be set while sensors can notify users through messages.

Not only can messages be sent by sensors in order to notify users, messages can also be exchanged between users.

Use case diagram (normal user)

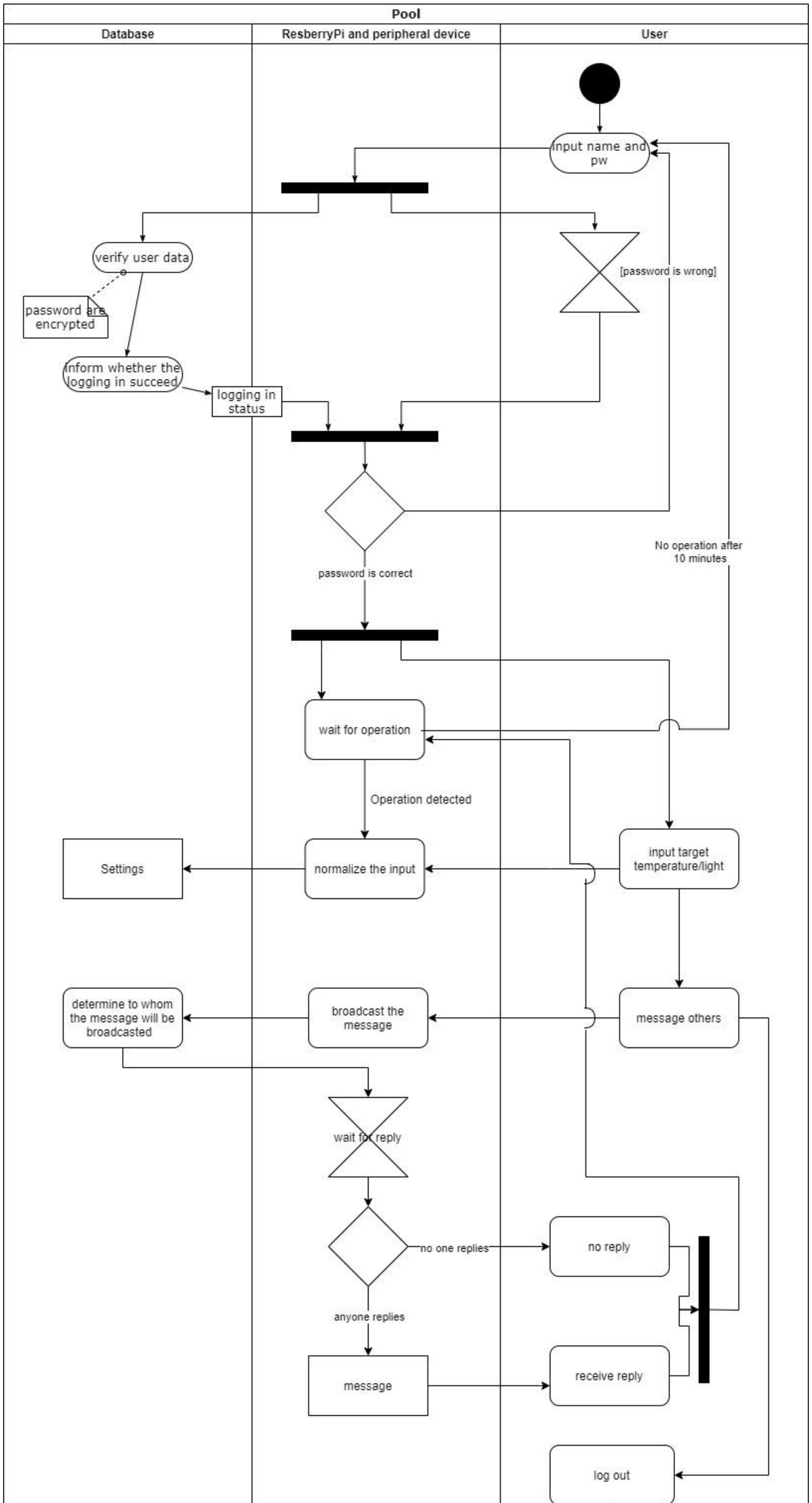


- View the lights that includes
  - Alternate lights
  - See conditions of the lights, to see the lights the users need to check into the database that stores the data of the conditions of the lights.
- View the ECO score, but the ECO score must first be calculated by the System.
- Login, the following things must happen:
- Reply to the message if you are at home or not)

Use case diagram (administrator)

- Create/modify Users, where the created/modified user will be stored in the database.
- Create/modify rooms with the followings things that could be in a room:
  - With Sensors
  - With lights
    - For all the components, they will be saved in the database.

Activity diagram


Pool
Database
ResberryPi and peripheral device
User
input name and pw
verify user data
password are encrypted
[password is wrong]
inform whether the logging in succeed
logging in status
password is correct
No operation after 10 minutes
wait for operation
Operation detected
Settings
normalize the input
input target temperature/light
determine to whom the message will be broadcasted
broadcast the message
message others
wait for reply
no one replies
no reply
anyone replies
message
receive reply
log out

1. Input the username and password: if we use sql to manage the database then the prepared statements should be used, or we at least check if the input is legal

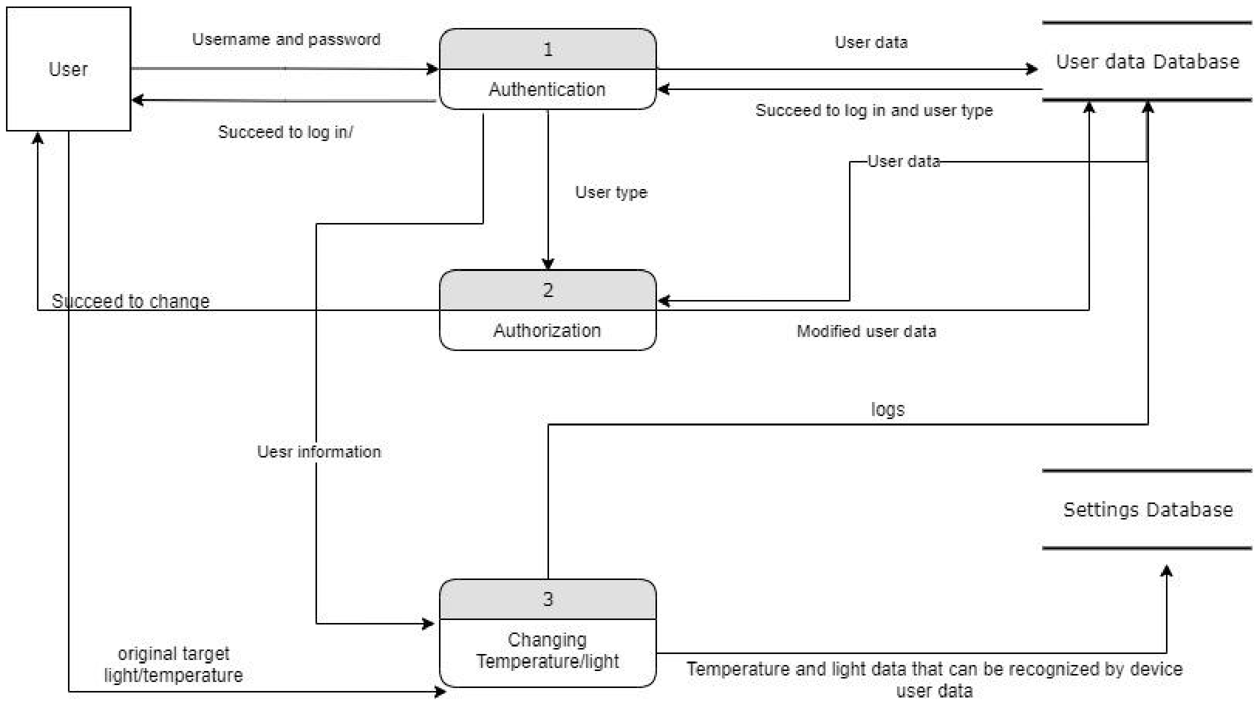
2. [password is wrong]: user can only try 5 times per 30 minutes

3. Wait for operation: if the user does not do an operation in 10 minutes, then it will be logged off.

4. Normalize the input: the input should be transfer introductions that can be used directly by device, also we should prevent any illegal input to protect the devices

5. Wait for reply: we should set a time limit for waiting to save resource

Dataflow diagram



1. Authentication: For convenience, user may use O Auth to log in

2. User data and settings (light/ temperature) should be stored separately, and devices should only access the settings data.

4. Product User Interface

The Product User Interface can be found in GitLab

5. Prevention/Mitigation Criteria (Security Controls)

Possible scenario: You enter the correct username but fail to enter the right password.

- The password doesn’t match / Forgot password?/ send a message to user(Was that you?)

Possible scenario: The user enters a phone number which is not the specified format.

- Enter your number in the following format (ex. 6217876543)

Possible scenario: You enter a wrong password more than 5 times.

- Block the system for this user for 30 minutes

Possible scenario: You enter a default password.

- Choose another password

Possible scenario: The length of the password is less than 8 digits.

- The password is not secure enough, set another password

Possible scenario: The token is too simple, such as ‘0000’, ‘1234’ etc.

- It must include a letter and cannot contain repetitive numbers, use random number generation

Possible scenario: You enter a wrong SMS token more than three times.

- Resend the SMS token and warn the user

Possible scenario: You enter wrong biometric data more than 3 times.

- Make sure that your finger is centred on the sensor/ Use a different method to login

Possible scenario: The biometric data are blurred.

- Can not identify / reread the biometric data/ use another method

Possible scenario: You try to perform an operation to which you’re not authorized twice in a row.

- Warning to user, block the access for a certain amount of time

Possible scenario: You try to open a log file without the proper permissions.

- access denied

Possible scenario: You request sensitive data from the server to which you do not have access.

- Shut down the server

6. The cost involved (if any):

To make our application secure, we need to handle the situations where our application could be hacked or unable to work. Here are the certain situations with their solutions and involved costs:

Sensitive Data exposure

Without any encryption, the sensitive data could be read by an unauthorized user or a man-in-the-middle that could manipulate the data without any notice of the 2 entities communicating. To solve this, we need to encrypt the sensitive Data with a strong encryption implementation. This will cost about 1 or 2 hours to implement this and finding a library of the encryption method.

Insecure Direct Object References

A direct object reference means that an internal object such as a file or database key is exposed to the user. The problem with this is that the attacker can provide this reference and, if authorization is either not enforced (or is broken), the attacker can access or do things that they should be precluded from. Such as the user gaining access to a file that downloads a file where you need to fill in an input of the file name. This can easily be misused by downloading other things around the server. With proper and consistent user authorization, and storing data internally and not relying on user inputs, could the problem be solved. It will cost about 5 or 6 hours to implement this.

broken authentication & session management

without proper authentication and session management, any user can break into the website and gain access to certain privileged actions that only authorized users can do. In this project, changing the lights or modifying the setup of the lights, without any notice if they are actually a homeowner that can do that. It is possible to gain access by a not proper session management, where the session ID could be leaked or might be predictable to gain access. With the use of already known secure frameworks or by implementing a proper framework, it is possible to solve this. It will cost about 5 or 6 hours to implement this.

SQL injection

Without any filter, a user could inject SQL coding into a database query to alternate the result of the query (SQL injection). This can be easily implemented by Validating User input, blacklist special characters or whitelist only normal characters and use of prepared Statements with Parameterized Queries. This will cost about 3 or 4 hours to *implement this with establishing a Database.*

7. Conclusion:

Design decisions

We decided that the system will contain roles for different users in a household. This will ensure safety in the house. This is done because it is not always convenient if for example children can make the temperature higher. This will create the challenge that we need to add an extra layer of security inside the application.

Since we make an automated home system, we need to be able to control lights and heaters with an electronic device such as a smartphone or a pc. If we only use one raspberry pi for a household, it can be a challenge to connect lights and heaters to this raspberry pi using wires, if the devices are in different rooms. Therefore we can use wireless light bulbs, and connecting the raspberry pi with these lightbulbs might be a challenge as well.

Reference:

We did not use any references for this document.