Requirement Documentation

By Group 2

Anton Geneser Matzen
202008936@post.au.dk
Std.nr: 202008936
Au-id: 683185

Emil Hilligsøe Lauritsen
202004154@post.au.dk
Std.nr: 202004154
Au-id: 668867

Martin Michaelsen
202007433@post.au.dk
Std.nr: 202007433
Au-id: 672598

Kevin Vollesen Schønberg
202007282@post.au.dk
Std.nr: 202007282
Au-id: 674059

Table of contents

1. Version History

Test Specification Versioning Table			
Version	Date	Editor	Modifications
0.1	15-02-2022	All	First iteration of the requirements
0.2	22-02-2022	All	Second iteration of the requirements
0.3	29-03-2022	All	Third iteration of the requirements
0.4	22-05-2022	All	Final iteration of the requirements

Table 1: Requirements Specification - Version History

2. Introduction

Safety is very important to maintain and when cognitively challenged people live on their own, they can easily increase the risk of accidents without knowing it. They can simply forget to lock the door or leave candles on when leaving the house or going to sleep. These situations are difficult to prevent without having a caretaker or family members in the house to help. And when saving money becomes a priority, some people are not able to get the help they need.

But with technology progressing, the possibility to place monitoring systems in the house can help maintain a safe home without the presence of a third party. In this document we will describe the system and the requirements for a monitoring system that will help the user to turn off the stove if left on and allow a third party to monitor this from outside the house[1].

This document contains a description of the system, the functional requirements and non-functional requirements, which all have been derived from the project specification given in lecture 1[2].

2.1 Abbreviations

Abbreviation	Description
PHP	PHP: Hypertext Preprocessor
UC (#)	Use case (number)
UML	Unified Modeling Language

Table 2: Requirements specification - Version History

[1] Wagner, 2022: p.17
[2] Wagner, 2022: p.17

3. System description

The system is to be used by cognitively challenged people who have trouble or potential trouble to turn off the stove after use or forgetting it was on. The system's functionality can be split into two, the main part of the system alerts the user if the stove is left on and turns it off if the user does not return. The other part of the system is the monitoring part, where the user or a third party can monitor when and how long the stove is left on.

The main part will use sensors to detect if the user leaves the kitchen while the stove is on. It will then alert the user with lights that indicate the stove is still on and no one is in the kitchen. If the user does not return in time, the system will cut the power to the stove and will alert the user the next time they use the stove that it was left on the last time they used it.

The monitoring part will use data from the main part on when and for how long the user leaves the kitchen with the stove on. It will do this by saving the time for when the user leaves the kitchen and if they return within 20 minutes it will save the time they returned to the kitchen. This can be used to monitor habits of the user and increase safety in the home.

The data for the monitoring part will be saved on a database and will be available via a web page. This means the user's habits with the stove can be monitored from outside the home, which will save time for caretakers or family members and make it easier to monitor the user.

4. Functional requirements

In this section we will describe the functionality of the system. We are using use case based requirements as described by Sommerville[3] and have made each use case so the result is measurable. To each use case we have described the main actor and stakeholders. We also stated pre- and postconditions to indicate what is expected before and after the use case happens. Finally we stated the main scenario and possible alternate scenarios if relevant.

[3] Sommerville, 2016: p.125

4.1 Use case overview

Here is a list of the use cases that we created from the project specification[4]. Each has been given a number, a name and a description.

UC ID	UC name	Description
UC1	Monitor Stove	The sensor detects if the user is in the kitchen and sends events to the controller
UC2	Start Timer	A timer is started and after 5 minutes the user is alerted in the room he occupies if no movement is detected. If the user returns before 5 minutes, the timer is not started. And no lights are effected
UC3	Alert user	The system tries to find the user and alert him/her by turning on the light in the room the user is in. If the user re enters the kitchen or the user is not found, then no lights turn on.
UC4	Search for User	Detect if the user is in one of the monitored rooms and informs the controller
UC5	Detect User Return	An event is received informing of the user presence in the kitchen after an absence of more than 10 seconds with the stove turned on.
UC6	Turn stove off	The user has been absent from the kitchen with the stove turned on for more than 20 minutes, the stove and alerting light is turned off, while an indicating light is turned on in the kitchen.
UC7	View Data	Allows the user to see the data from the database

Table 1 - Usecase overview.

[4] Wagner, 2022: p.17

4.2 UML Use case diagram

The use cases from 4.1 have been connected with the actors, dependencies and relations with other use cases and visualized in the diagram below.

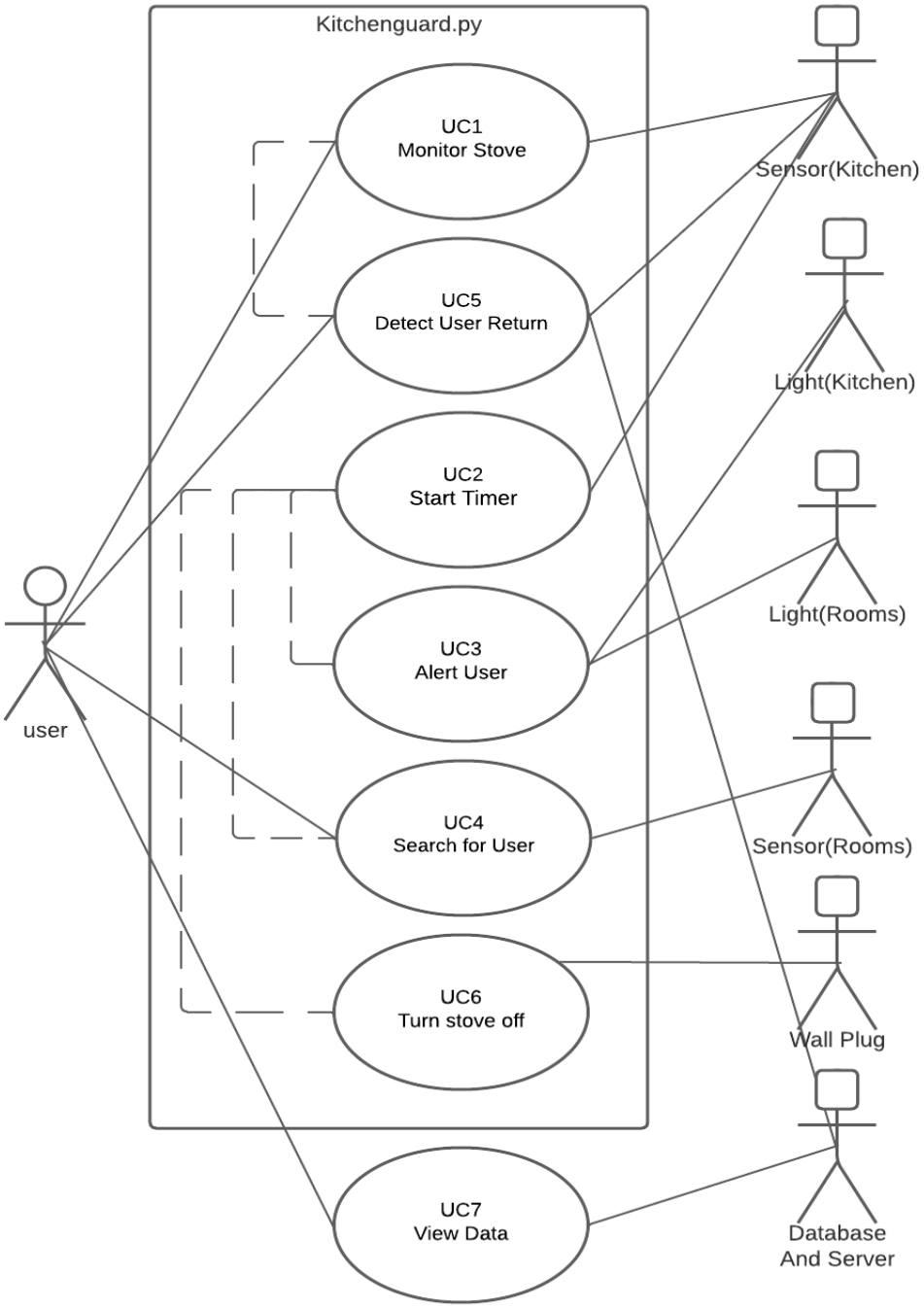

Figure 1. UML Use-Case diagram of Kitchenguard.

4.3 Use cases

Each use case will be presented in detail, with important information such as pre- and postconditions and main scenarios .

Name	Monitor Stove
Use case	UC1
Primary actor	User
Stakeholders	User & sensor(kitchen)
Precondition	Stove is turned on
Postcondition	Controller receives the event from the sensor
Main scenario	1. Sensor in the kitchen scans for movement 2. The sensor sends an event to the controller
Extension	If the Sensor in the kitchen is broken and fails to send an event, the controller will assume that there is no movement in the kitchen.

Name	Start Timer
Use case	UC2
Primary actor	Sensor(Kitchen)
Stakeholders	Sensor(Kitchen)
Precondition	The controller has not received events confirming that the user is still in the kitchen for more than 20 seconds.
Postcondition	The timer is started
Main scenario	1. Start a 20 minute timer
Extension	None

Name	Alert User
Use case	UC3
Primary actor	Alerting lights
Stakeholders	Alerting lights
Precondition	UC2 + 5 minutes has passed
Postcondition	Light is turned on where the user is
Main scenario	1. UC4 to find the room the user is located in 2. Turn on the light in that room 3. Turn off the light in other rooms
Extension	None

Name	Search for User
Use case	UC4
Primary actor	Sensors
Stakeholders	User, sensors
Precondition	UC3
Postcondition	The controller is told where the user is
Main scenario	1. Search for movement in one of the monitored rooms 2. Tell the controller where and if movement is detected
Extension	None

Name	Detect User Return
Use case	UC5
Primary actor	Sensor
Stakeholders	User, sensor(kitchen) and database
Precondition	The user is not in the kitchen and the stove is on
Postcondition	The timer is reset and the event is saved on the database
Main scenario	1. Controller receives event “User reenters the kitchen” from UC1 2. Sensor(kitchen) senses movement 3. Timer is stopped 4. All alerting lights are turned off 5. Report event to server and database
Extension	None

Name	Turn stove off
Use case	UC6
Primary actor	Powerplug
Stakeholders	Powerplug, lights
Precondition	UC2 + 20 minutes has passed
Postcondition	The stove is off and the light in the kitchen is showing a colored light indicating the stove was left on for too long
Main scenario	1. Turn off the stove 2. Turn off the alerting lights 3. Turn on light in kitchen indicating the stove was left on 4. Report to server and database
Extension	None

Name	View Data
Use case	UC7
Primary actor	User
Stakeholders	User & database
Precondition	None
Postcondition	The user has access to the information from the database
Main scenario	1. The user opens a browser 2. The user enters the web page 3. The user can watch the information from the database
Extension	None

Table 2. Use-case table.

5. Non-functional requirements

In this section we have listed the non-functional requirements of the system. These are what the system should contain and not what it should be able to do. These are derived from the project description[5].

NFR ID	NFR name	Description
NFR1	Must support at least 5 rooms including kitchen	The system must be able to support identification of user presence in at least 5 rooms including the kitchen.
NFR2	Database	The system should have a database for logging information.
NFR3	Distributed user interface server and client	A distributed user interface server and client must be developed for personal computer, tablet, web, and/or smartphone, for accessing the “monitoring” data. The server should be based on web technologies, either NodeJS, PHP, ASP.NET or Blazor (.NET)

Table 3. Non-functional requirements.

[5] Wagner, 2022: p.17

6. Requirements validation

Requirements validation is an important part of requirements engineering. In his book Sommerville describes how validation of the requirements can help avoid extensive rework costs caused by bad requirements[6].

In the validation process we performed several requirement reviews with the peer group and the customer. We then used this feedback to make sure the requirements were always in line with the customers/peer group's needs. The feedback can be seen in the appendix. In addition to this we also did verifiability checks that made sure that each of our requirements always were testable. This was the process than led us to the final use case-based requirements that can be seen in section 4.3

[6] Somerville, 2016: section 4.5, page 130

7. Sources

1. “Week 1 - Introduction”, Wagner, Stefan Rahr, 2022
2. “Software engineering 10th edition”, Sommerville, Pearson, 2016