PROJECT PLAN

Group name

First Airbenders

Project name

COMFORT HOME



Version 1.0

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Student number | Email | Role |
| Stefan Teeuwen | 3654900 | 415849@student.fontys.nl | Scrum Master/Project Leader |
| Viktor Ivanov | 3774147 | 425404@student.fontys.nl | Engineer |
| Ahmad Alzarkaoui | 3741834 | 441396@student.fontys.nl | Engineer |
| Ivaylo Ivanov  Valentin Vasilev | 3853764  3860140 | [432022@student.fontys.nl](mailto:432022@student.fontys.nl)  432543@student.fontys.nl | Engineer  Engineer |

**Table of contents**

[Document history 2](#_Toc35614057)

[Description 3](#_Toc35614058)

[Functional requirements 5](#_Toc35614059)

[Non-functional requirements 6](#_Toc35614060)

[**Performance and scalability** 6](#_Toc35614061)

[**Portability and compatibility** 6](#_Toc35614062)

[**Reliability, availability and maintainability** 6](#_Toc35614063)

[**Security** 6](#_Toc35614064)

[**Usability** 7](#_Toc35614065)

[Project way of working 7](#_Toc35614066)

[Communication plan 7](#_Toc35614067)

[**Internal communication** 8](#_Toc35614068)

[**Customer communication** 8](#_Toc35614069)

[**Teacher communication** 8](#_Toc35614070)

[Milestones 10](#_Toc35614071)

[Deliverables 12](#_Toc35614072)

[Risk analysis 13](#_Toc35614073)

[Configuration management 15](#_Toc35614074)

[Reference list 16](#_Toc35614075)

# **Document history**

|  |  |  |
| --- | --- | --- |
| **Version №** | **Description** | **Date** |
| 1.0 | Project description, project organization, project way of working, risks analysis, configuration management, deliverables were added | 21/02/2020 |
| 2.0 | Project description, project way of working and project organization were updated after feedback | 13/03/2020 |
| 3.0 | Removed minor errors from the document and made improvements after feedback, i.e. added this history table and added references. | 20/03/2020 |

# **Description**

In today’s world, houses get more and more insulated. That brings up problems such as condensation damages to the building, high risk of asthma or allergies triggering, high concentration of volatile organic compounds and Radon Gas, which can be fatal even if small amounts are inhaled on daily basis. To oppose those problems, we need to find a way to control the air quality in our homes. We at First Airbenders aim to provide our clients with an automated air-conditioning system like no other. Our goal is to achieve an automatic climate environment that does not have the need of constant human regulation. That goal will be made possible by implementing wireless sensors that analyze different measurements in the air such as humidity, temperature, carbon dioxide and particle matter. Based on the measurements a ventilation box connected with a fan will decide if and how much it should ventilate the room to improve the indoor climate and air quality.

# **Functional requirements**

**Картина, която съдържа карта

Описанието е генерирано автоматично**Our system will be based around the various sensors we are going to implement. Every sensor will measure continuously, but it will send its value every 15 minutes or when a sudden spike in the measures occurs. Those sensors will be connected to a simulated ventilation box in C# on a laptop application, where an algorithm will calculate the state of the air quality and the needed actions to take depending on that state. Furthermore, a green and red LEDs will show if the value of the sensor is being used by the algorithm. The C# system will have the ability to increase or decrease the number of sensors connected to it. Additionally, the red LED will also indicate by blinking when the bi-directional communication between a sensor and the app is lost. All the values sent by the sensors will be stored in a log where they can be accessed by the user. Visit the use cases document in the appendix for further explanation [3].

*visual representation of the functional requirements*

# **Non-functional requirements**

## **Performance and scalability**

**The functionality of the system should works in optimal status with discard how many components connected to the system.** The systems must work without interruptions. sending and receiving data through the Wi-Fi network between the sensors will support the idea of connecting many components which means the can be extended Easley.

## **Portability and compatibility**

**By using** the heterogeneity of components ( sensors, actuators, microcontroller,….) interacting, communicating with each other, we should ensure that the system perform without any problem. The system will be able to be compatible with other system component which has not identified, by using a well-structured protocol.

**The systems will be functional on every operating system and will not conflict any other applications and processes within the environment.**

## **Reliability, availability and maintainability**

**Since the system is fully automated, also the issues are taken into the account to be solved as much as possible in in automated way, self-healing is a protocol to repot the system and restart all sensors. The issue will sent as report to the support team automatically. We expect the system to work in natural conditions, so if unexpected action occur ( natural disasters ), in this case maintenance team should come to replace the damaged components.**

## **Security**

**To establish a secure application, only the customer and an assigned maintainer will have access to the system files.**

## **Usability**

**The goal of the system to provide a healthy and fresh air by using a completely automated system, depending on the values that come from the sensors, the ventilation box will decide the speed of the fan, on the other hand, in some cases the user wants to control some features like the temperature, so we leave this feature to him to control, through changing the speed of the fan manually, taking care of the quality of the air, therefor the system will give the user the advised speed of the fan.**

# **Project way of working**

The project group wants to transmit information between project members in such a way that there is no information loss and/or miscommunication. This is done in multiple manners which is discussed in the upcoming section. The project group invests time in project management, because setting up “correct” deadlines is crucial to achieve the highest customer satisfactory.

People often think that delivering a product before the scheduled deadline in itself is “the only benefit” and therefore forget to see the broader aspect of project management. When delivering a product before the deadline, the customer has the chance to observe and evaluate the delivered product and is therefore able to request the addition of essential extensions or removal of unnecessary functions. Then, the project group has the chance to implement those changes. This results in the highest customer satisfactory because the product is delivered on time and has all requested functions implemented.

# **Communication plan**

The project group expects to have certain setbacks which are inevitable, for example absence due to illness, defective parts or data loss due to corrupted files. The project group will therefore set “internal” deadlines and “external” deadlines. The internal deadlines are set by the project team without customer co-operation, which ensures enough time to live up to customer expectation and evades tasks snowballing. The external deadline(s) are predominantly set by the customer and are non-negotiable. The customer expects the desired result and the project group must deliver on time.

## **Internal communication**

An initial meeting is set at the project kick-off to ensure that all members have a say in the dates on which the internal deadlines are set. The primary ways of communication have been agreed to among group members in order to promote a healthy group process. These primary ways of communication are divided in two categories: digital and non-digital communication. The digital conversation is led via group chat on WhatsApp in which all members are able to express their opinions and ideas. The non-digital conversation is led via group discussions, which will be held weekly on Mondays. During the group discussions the project group will gauge at the internal deadlines, bring ideas to the table and criticize the current release. There will be stand up meetings as well in which (individual) project members will briefly communicate between each other about the current state of affairs. These can be held before/after school, during breaks or in-between lectures.

## **Customer communication**

The project group seeks feedback on the product in order to live up to the expectation of the customer. The customer will criticize the current release of the product during sprint demos in order to improve the final product. Sprint demos will be held once every three weeks.

## **Teacher communication**

The current release consists of multiple files, e.g. code structures or wiring diagrams. The completed files will be reviewed by group members (other than the writer) in order to ensure that the code is as optimal as possible. The mentor will be reviewing the current release (once every week) on Mondays in order to give feedback and thus to improve the end result.

**Roles & responsibilities**

The project itself is straight forward and the tasks are limited in difference.

Consequently, there are no major differences in roles and the roles are as follows:

|  |  |
| --- | --- |
| **Name** | **Role** |
| Stefan Teeuwen | Scrum Master / Project leader |
| Ahmad Alzarkaoui | Engineer |
| Viktor Ivanov | Engineer |
| Ivaylo Ivanov | Engineer |

The following is expected from all group members:

* Contribute and deliver conform the deadlines
* Taking responsibility over delivered components and expected contributions
* Participate in team activities
* Being able to perform individually
* Help out group members and express expertise
* Taking decisions independently without authorization

In addition to the previously mentioned statements, the following is expected from

the scrum master:

* Organize meetings
* Manage project plan activities
* Maintain documentation about feedback
* Correct execution of an Agile way of working

Team dynamics are important and we expect all members to be able to express their

ideas and opinions and allow open communication in order to avoid conflicts

between group members. Everyone should be able to show their specialties and

personality types in order to complement each other’s work. The optimal result for

the customer is achieved if all requirements from all group members are met and if

open communication is paramount.

# **Milestones**

|  |  |  |
| --- | --- | --- |
| **Activity name:** Minutes of Meeting | | **Delivery date:** Weekly |
| **Input:** Previous Minutes of Meeting, Agenda | | |
| **#1** | **Activities:** | |
| For every meeting the team has a Minutes of Meeting shall be published. The document shall describe how the meeting went, what topics were discussed and what decisions were made, as well as who participated in the meeting, where it took place and at what time. The minutes of meeting shall be updated after every meeting so that members who could not participate in the meeting for one reason or another have the opportunity to catch up. | | |
| **Output:** This week’s Minutes of Meeting | | |

|  |  |  |
| --- | --- | --- |
| **Activity name:** Project plan | | **Delivery date:** 21.02.2020 |
| **Input:** Use cases, deliverables, risk analysis, project structure and organization | | |
| **#2** | **Activities:** | |
| The project plan includes everything regarding the work of the team, including how the team will function, how the project will be organized, what the team is expected to deliver, how communication with the client takes place and an analysis of the risks. | | |
| **Output:** A general plan of how the project will proceed | | |

|  |  |  |
| --- | --- | --- |
| **Activity name:** Sensor calibration | | **Delivery date:** N/A |
| **Input:** Sensors provided by customer | | |
| **#3** | **Activities:** | |
| After the humidity, temperature and particulate matter sensors are provided by the customer the team shall set up the mentioned sensors for testing and calibration. | | |
| **Output:** Working sensors | | |

|  |  |  |
| --- | --- | --- |
| **Activity name:** Building the network | | **Delivery date:** N/A |
| **Input:** ZigBee module | | |
| **#4** | **Activities:** | |
| The team shall use the ZigBee module provided by the customer combined with an embedded board to build a simple network used for communication between the sensors and the C# application. | | |
| **Output:** Functioning network | | |

|  |  |  |
| --- | --- | --- |
| **Activity name:** C# application | | **Delivery date:** N/A |
| **Input:** Functioning network, sensors | | |
| **#5** | **Activities:** | |
| A C# application will be built to simulate a ventilation box. The application will take the readings from the sensors over the network and communicate with the sensors bi-directionally. Additionally, the final assembly of the system will take place. | | |
| **Output:** Working application | | |

# **Deliverables**

|  |  |  |
| --- | --- | --- |
| **Activity name:** Documentation | | **Delivery date:** N/A |
| **Input:** Project code | | |
| **#1** | **Activities:** | |
| The project team will deliver all documentation (system design document, UML diagrams, project plan, state machine diagrams etc.). | | |
| **Output:** Detailed documentation | | |

|  |  |  |
| --- | --- | --- |
| **Activity name:** Sprint demo | | **Delivery date:** every three weeks |
| **Input:** Word file and PowerPoint presentation | | |
| **#2** | **Activities:** | |
| The project team will hand in a Word and PowerPoint file to deliver a presentation that meets the needs for information of the customer | | |
| **Output:** Presentation | | |

|  |  |  |
| --- | --- | --- |
| **Activity name:** System | | **Delivery date:** N/A |
| **Input:** Given hardware | | |
| **#3** | **Activities:** | |
| The project team will deliver all required sensors connected as a single system, properly calibrated and ready to function as required by the customer. | | |
| **Output:** Assembled and configured hardware | | |

|  |  |  |
| --- | --- | --- |
| **Activity name:** C# application | | **Delivery date:** N/A |
| **Input:** N/A | | |
| **#4** | **Activities:** | |
| The project team will deliver a C# application that is used to monitor the sensor output. The application will mimic the behavior of a ventilation box, which is used to control the whole system. | | |
| **Output:** Ventilation box (simulated in C#) | | |

# **Risk analysis**

In this project we have defined the following risks:

1. Bad time management. Even though everything is scheduled and always updated in git, the possibility that someone forgets to check their scheduled times for meeting is still there.
2. Not meeting due dates for deliverables. Programming and embedded systems are not always straightforward and things almost never go according to plan. Additionally, it could be the case that someone is having troubles with their assignment, which could further slowdown the process.
3. Sprint demo failing. Sprint demos are always botched together solution just for demonstration purposes and prototyping. Thus there always exists the possibility that the demo fails, which will leave the client dissatisfied.
4. A project team member fails to deliver their part. Perhaps someone will have struggles with their task, maybe personal problems or lack of organization, anything can go wrong even on a personal level. This could lead to significant delays.
5. Documentation is incomplete. The team members are already quite familiar with the documentation process, however there could still be some holes in the process. Mentor’s input should be considered at all times.
6. Provided hardware malfunctions. The solutions the team is going to be working with are well-tested and stable products, however hardware is hardware and something could always go wrong.
7. The needed hardware is not provided on time by customer. This is always a possibility when the team relies on the customer to provide something.

|  |  |  |  |
| --- | --- | --- | --- |
| **Risk** | **Probability** | **Impact** | **Mitigation** |
| 1. Bad time management | Medium | Medium | Every meeting is scheduled in advance and everyone’s preferences are taken into account |
| 1. Not meeting due dates for deliverables | Low | High | Due dates are carefully tracked and scrum master makes sure everything is delivered on time |
| 1. Sprint demo failing | Medium | High | Sprint demo prototypes are carefully tested a day before the event |
| 1. A project team member fails to deliver their part | High | Medium | Scrum master keeps track of how everyone’s handling their tasks and offers help if needed. |
| 1. Documentation is incomplete | Low | Medium | We make sure to keep the client and the mentor updated to make sure everything is delivered correctly. |
| 1. Provided hardware malfunctions | Low | High | In case of any issues the client is to be contacted and new hardware is requested as soon as possible. |
| 1. The needed hardware is not provided on time by customer | Low | High | In case of this event the team is responsible for supplying needed hardware to finish on time. |

# **Configuration management**

As established, all of the code written by the team members should be submitted to the git repository to the /test directory. From there, a member of the team should review the code and give feedback. The program should then be tested and possibly a new iteration of the program will emerge, which also has to be reviewed, tested etc. If a member wants their code to be reviewed and tested, they can assign a task in “Boards” in GitLab. When an application is assigned to more than 1 person, the members are responsible for creating their own branch to work on in git instead of using the master branch. After each of their codes are finished, reviewed and tested, they are both responsible for merging their branches to the master branch, which contains a stable version of the project, which is suitable for sprint demos. Team members are also obligated to follow the directory structure and commit their work to the appropriate branch/directory.

Merging to the master branch should only be handled by one person and only after careful examination and review of the code.

# **Reference list**

1. Altexsoft. (18th October 2019). Non-functional Requirements: Examples, Types, How to Approach. Retrieved from <https://www.altexsoft.com/blog/non-functional-requirements/>
2. Lucidchart. (7th February 2018). *UML Use Case Diagram*[Video file]. Retrieved from <https://www.youtube.com/watch?v=zid-MVo7M-E>
3. Use cases.docx in Git